Neural Networks as Nearest Neighbors

Alan Oursland

Independent Researcher

September 2024

Abstract

In this paper, we present a new perspective on neural networks by interpreting them as nearest neighbor models. We demonstrate mathematically how linear layers with absolute value activations approximate Mahalanobis distances to cluster centers. Through experiments on synthetic data and the MNIST dataset, we compare our approach with traditional ReLU networks and nearest neighbor algorithms. Our findings offer insights into neural network interpretability and feature learning.

1 Introduction

Neural networks have become a cornerstone of modern machine learning, achieving state-of-the-art results in various domains. Despite their success, understanding the internal workings and interpretability of neural networks remains a challenge. In this paper, we propose a novel interpretation by viewing neural networks as nearest neighbor models.

We begin by deriving mathematical connections between linear layers and Mahalanobis distances. Building on this foundation, we conduct experiments to validate our approach and compare it with traditional activation functions and nearest neighbor algorithms.

Our contributions are as follows:

- We provide a mathematical framework linking neural networks to nearest neighbor methods.
- We demonstrate through experiments how linear nodes with absolute value activations approximate distances to cluster centers.
- We interpret the learned features of neural networks under this new perspective.

2 Mathematical Framework

In this section, we establish the mathematical foundation connecting neural networks to nearest neighbor methods through the Mahalanobis distance and principal component analysis (PCA). We demonstrate how linear layers with absolute value activations can approximate distances in the feature space. This theoretical framework provides a starting point for understanding the behavior of these networks, subject to empirical validation and potential refinement.

2.1 k-Means and Gaussian Mixture Models

In nearest neighbor classification, algorithms such as k-means use distance metrics to identify the feature mean that is closest to a data point. Using nearest neighbors, k-means partitions the data space into Voronoi cells centered on cluster centroids **??**. In many datasets, these Voronoi

cells tend to be sparsely populated, with only narrow regions of data represented. This sparsity can result in the false positives of randomly sampled or unobserved points. These points can exist in sparsely sampled, or empty areas, of the Voronoi cell. In other words, they may be assigned to clusters even though they fall outside the observed distribution of the data **??**.

To address this issue, one can extend k-means by incorporating a covariance matrix for each cluster, transforming it into a Gaussian mixture model (GMM) **??**. By leveraging the Mahalanobis distance **??**, which accounts for both the cluster mean and covariance, distances are adjusted based on the distribution's variance, effectively standardizing the space along the principal components of the data. This enables the model to better identify points that are unlikely to belong to any cluster. This concept naturally leads to the formulation of the multivariate Gaussian, where the covariance matrix governs the scaling of distances.

2.2 Multivariate Gaussian Distribution

A multivariate Gaussian (normal) distribution describes a d-dimensional random vector $\mathbf{x} \in \mathbb{R}^d$ with a mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$ (**?**). The probability density function (pdf) is given by:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{1}$$

where $|\boldsymbol{\Sigma}|$ denotes the determinant of the covariance matrix, and $\boldsymbol{\Sigma}^{-1}$ is its inverse.

2.3 Mahalanobis Distance and Principal Components

The Mahalanobis distance is a measure of the distance between a point $\mathbf{x}$ and the mean $\boldsymbol{\mu}$ of a distribution, taking into account the covariance of the data (**?**). It is defined as:

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})} \tag{2}$$

This distance accounts for the variance along each principal component (direction) in the data. Furthermore, by utilizing the inverse covariance matrix, the Mahalanobis distance facilitates data whitening (**?**). This process transforms the original data into a new set where the variables are uncorrelated and have unit variance, effectively converting the data into a spherical Gaussian distribution.

By performing eigenvalue decomposition on the covariance matrix $\boldsymbol{\Sigma}$, we obtain:

$$\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top, \tag{3}$$

where:

- $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_d]$ is a matrix whose columns are the orthogonal unit eigenvectors $\mathbf{v}_i$ of $\boldsymbol{\Sigma}$.
- $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$ is a diagonal matrix of the corresponding eigenvalues λ_i, representing the variance along each eigenvector (**?**).

Using the eigenvalue decomposition, the Mahalanobis distance can be rewritten as:

$$D_M(\mathbf{x}) = \sqrt{\sum_{i=1}^{d} \frac{(\mathbf{v}_i^\top(\mathbf{x}-\boldsymbol{\mu}))^2}{\lambda_i}} \tag{4}$$

This expression reveals that the Mahalanobis distance is calculated by projecting the data onto each principal component (the eigenvectors of the covariance matrix), scaling these projections by the inverse square root of the corresponding eigenvalues. Specifically, it computes the Euclidean $(\ell_2) norm of these scaled, one-dimensional components.$

Thus, each projection of the point onto a principal component can be represented by the standardized form of the data in that direction, as shown below:

$$y = \frac{\mathbf{v}^\top(\mathbf{x} - \boldsymbol{\mu})}{\sqrt{\lambda}} \tag{5}$$

The Mahalanobis distance along each principal component is represented by the absolute value of the standardized coordinate along that component:

$$D_{M,\mathbf{v}}(\mathbf{x}) = \left| \frac{\mathbf{v}^\top(\mathbf{x} - \boldsymbol{\mu})}{\sqrt{\lambda}} \right| \tag{6}$$

This simplifies the Mahalanobis distance for a single principal component, highlighting how each direction contributes independently to the overall distance.

2.4 Connection to Neural Networks

We observe that Equation (**??**) resembles the operation of a linear layer in a neural network (**?**). By defining:

$$\mathbf{W} = \frac{\mathbf{v}^\top}{\sqrt{\lambda}}, \tag{7}$$

$$b = -\frac{\mathbf{v}^\top \boldsymbol{\mu}}{\sqrt{\lambda}}, \tag{8}$$

we can rewrite Equation (**??**) as:

$$y = \mathbf{W}\mathbf{x} + b. \tag{9}$$

This equation is identical to that of a linear layer with weights $\mathbf{W}$ and bias b. Linear layers in neural networks perform the same operation as PCA-based data whitening, provided the weights are aligned with the principal components. When combined with the absolute value function, they produce a distance metric from a mean value along a specific orientation.

2.5 L1 Norm Approximation of Mahalanobis Distance

We approximate the Mahalanobis distance using the ℓ_1 norm. This avoids the computational challenges introduced by squared terms in traditional distance calculations, such as numerical overflow and vanishing gradients (**?**). This approximation is computationally simple and integrates well with existing neural network architectures.

The Mahalanobis distance can be approximated by summing the absolute values of the standardized coordinates along each principal component:

$$D_{M,\text{approx}}(\mathbf{x}) = \sum_{i=1}^{d} |\mathbf{W}_i\mathbf{x} + b_i|, \tag{10}$$

where $\mathbf{W}_i$ and b_i correspond to the principal components' weights and biases.

2.5.1 Advantages of L1 Norm Approximation

- **Computational Efficiency**: Absolute values are less intensive than squaring and taking square roots (**?**).
- **Improved Gradient Flow**: The absolute value function avoids the vanishing gradient problem (**?**).

- **Alignment with Neural Networks**: Linear layers followed by absolute value activation mirror standard architectures (**?**).

Although the ℓ_1 norm approximation differs from the true Mahalanobis distance, it offers a practical approach that balances computational efficiency with the ability to capture meaningful distance information, particularly in the context of neural network architectures designed for high-dimensional data.

The absolute value function shares some similarities with ReLU, but it preserves both positive and negative deviations from the decision boundary, making it a more convenient choice for distance measurement (**?**). However, ReLU can preserve similar information by offsetting the linear node's decision boundary from the cluster mean. While the absolute value function directly converts a coordinate from the range $[-d, +d]$, ReLU expresses that coordinate as $[0, 2d]$, with the subsequent layer effectively mapping this back to $[-d, +d]$ by adjusting the bias. This allows ReLU networks to capture the same essential distance information as Abs, even though the details differ. ReLU networks can still maintain the key principles of distance measurement. This paper focuses on the absolute value evaluation function for ease of analysis.

2.6 Neural Networks Can Approximate Nearest Neighbors

The ℓ_1 approximation of the Mahalanobis distance in Equation (**??**) suggests that neural networks with linear layers and either Abs or ReLU activations can compute distances between inputs and learned prototypes (cluster centers). Each neuron may approximate the distance along a learned direction, potentially related to a principal component.

This perspective implies that neural networks can perform a form of nearest neighbor search in the feature space. The smaller the output y, the closer the input $\mathbf{x}$ is to the corresponding cluster center represented by the neuron.

2.7 Implications for Model Interpretation

Understanding neurons as computing approximated Mahalanobis distances offers a fresh perspective on interpreting neural network behavior, particularly in relation to expected values:

- **Feature Learning**: Each neuron can align with a principal direction in the data, capturing significant variance along that axis. This alignment would allow the network to learn a compact representation of the input space, effectively performing dimensionality reduction.
- **Prototype-Based Clustering**: The network naturally groups inputs based on their proximity to learned prototypes. With Abs activations, these prototypes correspond to cluster means, providing a clear statistical interpretation of the network's organization of the input space.
- **Interpretability and Activation Functions**: The choice of activation function impacts interpretability. Abs activations center decision boundaries on cluster means, facilitating direct interpretation of neuron behavior. In contrast, ReLU shifts these boundaries, requiring additional complexity to represent the same information and potentially obscuring the underlying statistical structure (**?**).

2.8 From Theory to Practice

With this theoretical foundation established, we now turn to a series of experiments designed to validate our theory and demonstrate how neural networks with linear layers and absolute value activations can approximate nearest neighbor models in practice. These experiments will test our mathematical framework on both synthetic and real-world datasets, providing empirical support and refinement of our interpretation of neural networks as distance-based classifiers.

3 Experiment 1: Single Linear Node on a 2D Gaussian

In this experiment, we investigate whether single linear-node neural networks with either absolute value (Abs) or rectified linear unit (ReLU) activation functions can learn to approximate the Mahalanobis distance in a 2D Gaussian distribution. Specifically, we aim to determine if these simple models capture the principal components of the Gaussian distribution, as predicted by our theoretical framework.

3.1 Methodology

We generated a synthetic dataset from a bivariate Gaussian distribution. The dataset X was used to compute the true Mahalanobis distance for each point Equation (**??**). These distances served as the target outputs y for training the models.

Each model consisted of a single linear unit followed by either a ReLU or Abs activation function. Weights were initialized using Xavier initialization. To ensure diverse initial conditions, we selected a random point $\mathbf{z} \in \mathbb{R}^2$ uniformly from $[-8, 8]^2$ and set the bias as:

$$b = -\mathbf{W}\mathbf{z}, \tag{11}$$

so that the decision boundary (defined by $\mathbf{W}\mathbf{x} + b = 0$) passes through the point $\mathbf{z}$. This strategy ensures that the initial decision boundaries are distributed throughout the input space.

To prevent dead ReLU nodes (nodes that output zero for all inputs), we checked the initial activations and reinitialized any models where this occurred.

Setup and training parameters are summarized in Table **??**.

Table 1: Configuration Parameters

Parameter	Value
Activation Functions	ReLU, Abs
Loss Function	Mean Squared Error (MSELoss)
Optimizer	SGD
Learning Rate	0.001
Epochs	200
Momentum	0.9
Weight Initialization	Xavier Initialization
Bias Initialization	$b = -\mathbf{W}\mathbf{z}$, $\mathbf{z} \sim \mathcal{U}([-8, 8]^2)$
Number of Models	300 per activation function
Number of Data Points	1024
Mean Vector $\boldsymbol{\mu}$	$[3.0, 2.0]^\top$
Covariance Matrix $\boldsymbol{\Sigma}$	$\begin{bmatrix} 2.0 & -1.0 \\ -1.0 & 1.0 \end{bmatrix}$

3.2 Objectives and Hypotheses

Based on our theoretical framework, we hypothesize the following:

1. **Abs Activation Hypothesis**: Models using the Abs activation function will learn to approximate the Mahalanobis distance by aligning their weights the largest principal component of the Gaussian distribution. This alignment should result in lower error compared to models using ReLU activations.

2.

3. **ReLU Activation Hypothesis**: Models using the ReLU activation function are less likely to align with the principal components due to the asymmetry of the ReLU function, potentially resulting in higher error. However, studying their solution space may provide insights into how ReLU models approximate the distance function.

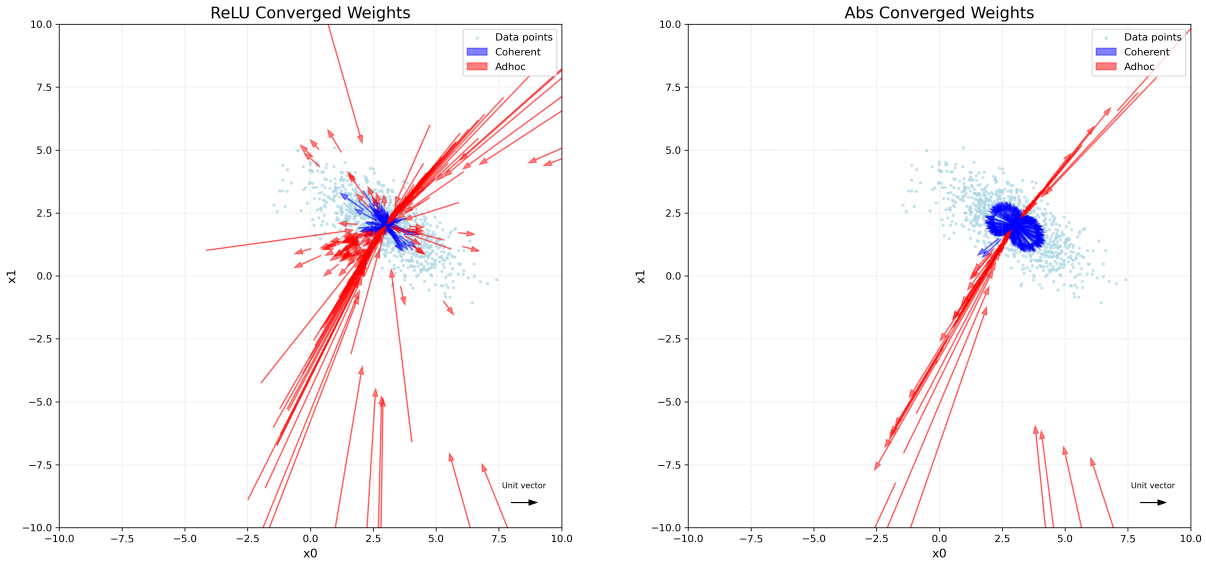
3.3 Results and Discussion

3.3.1 Visualization of Learned Weights

To analyze the learned models, we visualized the weights $\mathbf{W}$ by representing them as vectors originating from the projection of the mean vector $\boldsymbol{\mu}$ onto the model's decision boundary. Specifically, for each model, we:

1. Calculated the point $\mathbf{p}$ where the decision boundary intersects the line passing through $\boldsymbol{\mu}$ and perpendicular to the decision boundary.
2. Represented the weight vector $\mathbf{W}$ as an arrow originating from $\mathbf{p}$.
3. Scaled the weight vectors for visualization purposes but maintained their orientation to reflect the learned directions.

This visualization allows us to assess how the models' learned weights align with the principal components of the data distribution.



(a) ReLU Activation

(b) Abs Activation

Figure 1: Converged states for ReLU and Abs activations. The blue dots represent the 2D Gaussian data points. Arrows represent the learned weight vectors originating from the projection of $\boldsymbol{\mu}$ onto the decision boundary. Blue arrows indicate Coherent solutions, and red arrows indicate Adhoc solutions.

3.3.2 Solution Classes: Coherent vs. Adhoc

Upon analysis, we observed that the solutions could be classified into two categories:

- **Coherent Solutions**: Models where the decision boundary intersects the Gaussian mean $\boldsymbol{\mu}$. These models tend to align with the statistical properties of the data, potentially capturing the principal components. We classified a model as Coherent if the distance between $\boldsymbol{\mu}$ and the decision boundary was less than a small threshold ϵ.

- **Adhoc Solutions**: Models where the decision boundary does not intersect $\boldsymbol{\mu}$. These models approximate the target function through other means and may not have an obvious interpretation with respect to the data's statistical properties.

3.3.3 Quantitative Results

Table **??** summarizes the results of the experiment. The Mean Squared Error (MSE) is reported for both activation functions, along with the counts of models in each solution class.

Table 2: Single Linear Node Results

Metric	ReLU	Abs
Error	1.090	0.468
Coherent Error	0.976	0.455
Adhoc Error	0.822	0.523
Coherent Count	4	243
Adhoc Count	214	57
Dead Count	82	0

3.3.4 Analysis of Results

ReLU Activation Models As shown in the left image of Figure **??**, most ReLU models (214 out of 300) resulted in Adhoc solutions. These models tended to align their weights with the dimension of least variance, possibly attempting to minimize error estimating an overall average. The average MSE for ReLU models was higher than that of Abs models, and the Coherent ReLU models had higher error than their Adhoc counterparts.

Abs Activation Models The right image in Figure **??** illustrates the converged states for models with Abs activation. The majority of these models (243 out of 300) produced Coherent solutions, with decision boundaries intersecting near the Gaussian mean. These models exhibited lower average MSE compared to Adhoc Abs models, confirming our hypothesis that Abs activations effectively approximate the Mahalanobis distance by aligning with a mixture of principal components.

However, the Coherent Abs models did not align the largest principal component. While the decision boundary comes very close to the cluster mean, the weights seem to express a mixture of components, as expressed in Equation (**??**):

$$\mathbf{W} = \sum_{i=1}^{d} \alpha_i \mathbf{v}_i^\top, \tag{12}$$

where $\mathbf{v}_i$ are the principal components and α_i are weighting coefficients.

The displayed vectors $v/|v|^2|$, seem to form an ellipse or a pair of circles suggesting that $\ell_2(\alpha) = 1$ or perhaps that the foci of the Gaussian ellipse have some relationship to the gradient. The exact relationship requires further study. There may be a numerical instability around the direction of least variance that causes the decision boundaries to be ejected from the cluster.

Some of the Adhoc Abs solutions behaved like a ReLU because the Abs function on received inputs on only a single side of the decision boundary.

3.3.5 Observations on Initialization and Activation Functions

The initial weights did not show a clear pattern predicting the solution class into which a model would fall. However, the behavior of the models highlights the importance of the activation function and initialization:

- **Abs Models Functioning Like ReLU**: The modes where Abs acts like ReLU suggests that proper initialization is crucial for Abs models to capture both sides of the data distribution.
- **ReLU Models Aligning with Least Variance Direction**: The ReLU models often aligned their weights with the smallest principal component, minimizing output the variance across data points. The models are trying to return a constant value, presumably the mean of the target values.
- **Importance of Decision Boundary Placement**: Ensuring that the decision boundary intersects the data cluster increases the likelihood of a model learning a Coherent solution, particularly for Abs activations.

3.4 Conclusion and Next Steps

The Abs models do not learn principal components directly as we theorized. Instead we discovered a surprising fact about the power of linear units: A single linear unit can estimate the Mahalanobis distance for any dimension Gaussian through principal component mixing.

The findings also reveal the importance of initialization strategies. Abs models that failed to intersect the data cluster tended to behave like ReLU models, resulting in higher error and Adhoc solutions. This suggests that initializing the bias to ensure the decision boundary passes through the data cluster can improve model performance.

3.5 Data Sampling Initialization

To address the observations from the initial experiment regarding the importance of the decision boundary intersecting the data cluster, we explored a novel initialization strategy for the bias b. Instead of selecting a random point $\mathbf{z}$ from the uniform distribution over $[-8, 8]^2$, we now initialize b using a randomly selected data point $\mathbf{x}_0$ from the dataset X. Specifically, we set:

$$b = -\mathbf{W}\mathbf{x}_0, \tag{13}$$

ensuring that the initial decision boundary passes through a point within the data cluster. This approach increases the likelihood that the model will capture variations on both sides of the decision boundary, particularly for models using the Abs activation function.

Other than this modification to the bias initialization, all other parameters and settings remain the same as in the initial experiment (see Table **??**).

3.5.1 Results and Discussion

Table **??** summarizes the results of the experiment using the data sampling initialization strategy.

Table 3: Results with Data Sampling Initialization

Metric	ReLU	Abs
Average MSE (All Models)	0.833	0.407
Average MSE (Coherent Models)	1.165	0.396
Average MSE (Adhoc Models)	0.660	0.541
Number of Coherent Models	93	277
Number of Adhoc Models	203	23
Number of Dead Models	4	0

Compared to the initial experiment, the data sampling initialization significantly increased the number of Coherent solutions for both activation functions, especially for the Abs models.

Specifically, the number of Coherent Abs models increased from 243 to 277 out of 300, while the number of Adhoc Abs models decreased correspondingly.

For the ReLU models, the number of Coherent solutions also increased from 4 to 93, and the number of dead nodes decreased dramatically from 82 to 4. This suggests that initializing the bias using a data point from the dataset effectively activates more ReLU models, allowing them to participate in learning.

Analysis of Abs Models The Abs models benefited significantly from the data sampling initialization. The increase in Coherent models indicates that more models had decision boundaries intersecting the data cluster, allowing them to capture variations on both sides of the mean. This led to a slight improvement in the average MSE for Coherent Abs models (from 0.455 to 0.396).

The reduction in the number of Adhoc Abs models further supports the effectiveness of the initialization strategy. With fewer models behaving like ReLU due to all data points being on one side of the decision boundary, the overall performance of the Abs models improved.

Analysis of ReLU Models The data sampling initialization led to a significant decrease in the number of dead ReLU nodes, from 82 down to 4. Most of the nodes that were previously dead became Coherent nodes, increasing the number of Coherent ReLU models from 4 to 93 out of 300. This suggests that initializing the bias using data points effectively activates more ReLU models, allowing them to contribute to learning.

While the number of Coherent ReLU models increased, their average MSE also increased (from 0.976 to 1.165). This indicates that although more ReLU models had decision boundaries intersecting the data cluster, they were still limited by the asymmetry of the ReLU activation function, which hinders their ability to approximate the Mahalanobis distance effectively.

The average MSE for Adhoc ReLU models decreased from 0.822 to 0.660. This decrease is noteworthy, but the exact reason is unclear. It may be due to the data sampling initialization producing better Adhoc models; however, further investigation is needed to confirm this. Therefore, we simply note this observation without proposing a specific cause.

3.5.2 Conclusion of Data Sampling Initialization Experiment

For the Abs models, this initialization strategy reinforced their ability to approximate the Mahalanobis distance through principal component mixing, as more models aligned their weights with a mixture of principal components.

For ReLU models, while there was an increase in Coherent solutions, their inherent limitations due to the activation function's asymmetry remained evident.

The data sampling initialization strategy proved effective in increasing the number of Coherent solutions. By ensuring that the decision boundary intersects the data cluster, the models are better positioned to learn representations that capture the statistical properties of the data.

3.6 Overall Conclusion of Experiment 1

Experiment 1 reveals that:

- Single linear nodes with Abs activation can approximate the Mahalanobis distance of a multivariate Gaussian distribution by learning a mixture of principal components.
- We observe two classes of solutions: Coherent units represent deep statistical properies of the data. Adhoc units find solutions that are not deeply grounded in statistical properties.
- Initialization strategies that ensure the decision boundary intersects the data cluster enhance the likelihood of models converging to Coherent solutions.

Coherent solutions have lower error with Abs. However, this problem was designed for the Abs node to do well on. Coherent solutions in ReLU have higher error, however Abs can be replicated with two ReLU nodes. Additionally, ReLU nodes can also express a signed Mahalanobis distance over [-d..d] using [0..2d] with the next layer accounting for the zero offset. In practice there should not be a huge performance difference between Abs and ReLU.

Adhoc solutions seem to minimize output variance, perhaps attempting to calculate a mean of the target data.

While our initialization strategy increases the change of Coherent features, we don't know that Coherent features are actually beneficial in full models. The next set of experiments, we will investigate whether the benefits of Coherent features and Abs activations persist when applied to the MNIST dataset. This will help determine the practicality and scalability of our approach in more complex scenarios.

4 Experiment 2: Shallow Networks on MNIST

TBD: This experiment trains two layer networks on MNIST using Abs and ReLU with and without the proposed initialization strategy. It also uses kMeans to train a nearest neighbor model to compare performance.

Experiment 2 explores the practicality of our distance metric interpretation of neural networks. Expanding from simpler models in Experiment 1, we train a two-layer feedforward network on the MNIST handwritten digit dataset. This experiment tests whether the Abs activation function and bias sampling improve performance in more complex, real-world tasks. We also interpret the linear nodes' representation using the L1 Mahalanobis distance framework.

Interpretation in Experiment 1 relied on the data representation ground truth. We no longer have that with MNIST. We introduce new techniques to distinguish between the Coherent and Adhoc representation classes that were identified in Experiment 1. We also find the mean input for each hidden node that represents a canonical reprsentation of the feature that node selects. We test whether an increase in Coherent nodes corresponds to improved model performance.

4.1 Methodology

We train a two-layer feedforward neural network on MNIST. Each layer is followed by either a ReLU or Abs activation function. The output layer also includes an activation function to ensure it functions as a distance metric, aligning with our presented theory. Weights are initialized He Normal Initialization. Bias initialization is tested between zeroing and data sampling, as introduced in Experiment 1.

We used cross entropy loss (CEL) as the model criterion. CEL expects the larger outputs to indicate class membership, but our distance metric framing treats smaller distances as indicating membership. To adapt our model to CEL, we negate the outputs of the final activation function.

We train ten models for each experimental condition.

The remaining model parameters are outlined in Table **??**.

4.2 Objectives and Hypotheses

Building on our findings from Experiment 1, we hypothesize the following for Experiment 2:

- **Abs vs. ReLU Hypothesis**: Abs and ReLU will perform similarly. Before the activation function is applied, the Abs node represents variance over $[-\delta..+\delta]$, while the ReLU node represents variance over $[0..2\delta]$. Both should yield comparable overall performance despite these differences.
- **Bias Sampling Hypothesis**: Bias sampling continues to increase the number of Coherent features, even in more complex models like MNIST. This strategy should lead to

Table 4: Configuration Parameters for Experiment 2

Parameter	Value
Activation Functions	ReLU, Abs
Loss Function	CrossEntropyLoss
Optimizer	SGD
Epochs	50
Learning Rate	0.001
Momentum	0.9
Batch Size	64
Hidden Layer Size	128
Output Layer	Negated activation function
Weight Initialization	He Normal Initialization
Bias Initialization	Random or Sampled from Data
Dataset	MNIST (60,000 training samples, 10,000 test samples)

better alignment of the network's nodes with significant data clusters, improving both interpretability and model performance.

- **Coherent Features Hypothesis**: Models with a higher proportion of Coherent features will perform better than those with more Adhoc features. Coherent features align with meaningful clusters, leading to improved model generalization and accuracy.
- **Feature Interpretability Hypothesis**: Interpreting linear nodes as outputting distance metrics simplifies the understanding of the features that these nodes select, making the model's internal structure easier to interpret.

4.3 Results and Discussion

4.3.1 Performance Results

Table **??** summarizes the performance metrics across the different experimental conditions. We report the average accuracy, precision, recall, F1 score, and test error, along with the minimum and maximum accuracy to give a sense of variability across the ten models trained for each condition. We also include metrics for model diversity to measure the number of differing predictions between models within the same condition.

Table 5: Performance Results for Experimental Conditions

Metric	ReLU	ReLU + Bias Sampling	Abs	Abs + Bias Sampling
Accuracy (Avg)	0.9773	0.9770	0.9749	0.9773
Accuracy (Min)	0.9759	0.9756	0.9554	0.9728
Accuracy (Max)	0.9785	0.9779	0.9799	0.9795
Precision	0.9772	0.9769	0.9750	0.9772
Recall	0.9771	0.9768	0.9747	0.9771
F1 Score	0.9771	0.9768	0.9748	0.9771
Test Error	0.0747	0.0757	0.0855	0.0754
Avg Error Count	227.0	230.3	250.9	227.1
Unique Errors	509	509	805	617
Consistent Errors	70	79	47	40
Voting Errors	213	215	205	204
L2 Errors	188	185	163	163

Figure **??** presents two graphs: one showing the average training error and another showing the average test error for each configuration, displayed side by side. These curves provide insight into the learning dynamics and generalization performance of the models.

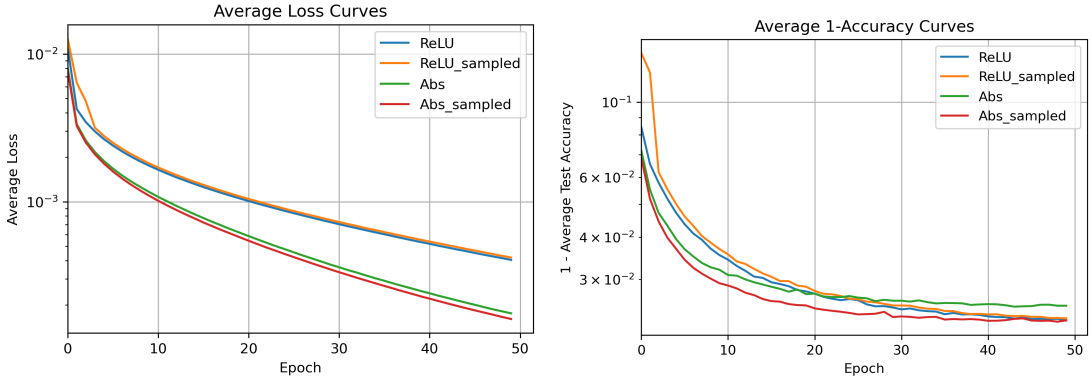


Figure 2: Training and test error curves for each experimental condition. The left plot shows the average training error, while the right plot shows the average test error. Both curves are averaged over the ten models trained for each condition.

4.4 Results and Discussion

The performance metrics summarized in Table **??** and Figure **??** provide comprehensive insights into the effectiveness and diversity of the trained models under different experimental conditions. Each metric is averaged over ten trained models per configuration, ensuring robust statistical significance.

Accuracy and Performance Metrics All experimental configurations achieved high average accuracy on the MNIST dataset, ranging from **97.0%** to **97.73%**, indicating strong overall performance. Specifically, both ReLU and Abs activation functions with bias sampling attained an average accuracy of **97.73%**. In contrast, Abs-activated models without bias sampling achieved a slightly lower average accuracy of **97.49%**, accompanied by a broader accuracy range from **95.54%** to **97.99%**. This wider range for Abs without bias sampling suggests greater variability in feature learning across different training runs, potentially indicating underfitting. The higher variability is further supported by the increased number of unique errors (805) in Abs models without bias sampling compared to ReLU models (509), highlighting diverse feature representations that may not consistently capture the underlying data structure.

Precision, Recall, and F1 Score metrics are closely aligned with the observed accuracy trends. Abs-activated models without bias sampling demonstrate slightly lower Precision (**97.50%**), Recall (**97.47%**), and F1 Score (**97.48%**) compared to ReLU configurations (**Precision**: 97.72%, **Recall**: 97.71%, **F1 Score**: 97.71%). This minor discrepancy aligns with the broader accuracy range and higher unique error counts, indicating that while Abs models are effective, their varied feature learning may introduce inconsistencies in classification.

Model Diversity Metrics The model diversity metrics offer valuable perspectives on the consensus and variability among the trained models. Abs models without bias sampling exhibit a significantly higher number of unique errors (805), indicating that a larger number of test data points were misclassified by at least one model. In comparison, ReLU models recorded 509 unique errors, and Abs models with bias sampling had 617 unique errors. This high unique error count in Abs models without bias sampling suggests that these models are learning a more diverse set of features, capturing varied aspects of the data but at the cost of consistency across different training runs.

Consistent errors, defined as data points misclassified by all models within a configuration, remain relatively low across all conditions. Abs models with bias sampling recorded the fewest consistent errors (40), whereas ReLU models without bias sampling had the highest count (79). This indicates that bias sampling effectively reduces systematic misclassifications, promoting greater alignment with significant data clusters and enhancing overall model reliability.

Voting Errors, representing the number of data points misclassified by five or more models, are comparable across all configurations, ranging from 204 to 215. This similarity suggests that ensemble methods would encounter similar levels of misclassification regardless of activation function or initialization strategy **?**. Additionally, L2 Errors, which assess the distance metric predictions by computing the L2 norm of the prediction vectors, remain consistent across all experimental conditions (163-188), indicating stable distance-based representations.

Implications for Hypotheses The experimental results align with the proposed hypotheses in several ways:

- **Abs vs. ReLU Performance**: Abs and ReLU activations exhibit similar average accuracies when bias sampling is employed. However, Abs activations without bias sampling show greater variability and higher unique error counts, supporting the hypothesis that Abs can capture diverse features but may require effective initialization to maintain consistency.
- **Bias Sampling Effectiveness**: The bias sampling initialization strategy consistently enhances model performance and reduces unique errors, particularly for Abs activations. This supports the hypothesis that bias sampling promotes better alignment with significant data clusters, thereby improving both interpretability and model performance.
- **Model Diversity and Underfitting**: The increased accuracy range and unique errors in Abs models without bias sampling suggest potential underfitting, as these models may benefit from additional hidden nodes to represent a more comprehensive set of features. This observation aligns with the hypothesis that higher model diversity can capture varied data aspects but may introduce inconsistencies.

Overall, Experiment 2 demonstrates that while both Abs and ReLU activation functions are effective in training shallow networks on the MNIST dataset, the combination of Abs activations with bias sampling initialization offers enhanced performance consistency and interpretability. The observed model diversity metrics highlight the trade-offs between feature richness and classification consistency, guiding future architectural and initialization strategies for optimizing neural network performance.

5 Interpretation of Learned Features

TBD: This section uses the models trained the the previous experiment, interprets the linear nodes as Gaussian and finds the mean value. These are compared to the means used to build the kMeans NN model.

5.1 Extracting Expected Values

We calculated the expected value (mean) for each node's corresponding 1D Gaussian to interpret the learned features.

5.2 Visualization

6 Experiment 3: Application to LeNet on MNIST

TBD: This experiment trains LeNet on MNIST using ReLU and Abs and compares performance.

6.1 Modifying LeNet Architecture

We adapted the LeNet architecture by replacing traditional activation functions with absolute value activations.

6.2 Training and Evaluation

7 Discussion

TBD: Discussion of experimental results.

7.1 The Nearest Neighbor Interpretation

Our experiments support the interpretation of neural networks as nearest neighbor models. The use of absolute value activations allows linear layers to approximate Mahalanobis distances effectively.

7.2 Advantages and Limitations

7.3 Connections to Batch Normalization

While not the focus of this paper, there is a connection between our Gaussian interpretation and batch normalization techniques, suggesting avenues for future research.

8 Conclusion

TBD:

In this paper, we introduced a new perspective on neural networks by interpreting them as nearest neighbor models. Through mathematical derivations and empirical experiments, we demonstrated how linear layers with absolute value activations approximate Mahalanobis distances to cluster centers.

Our findings offer valuable insights into neural network interpretability and suggest potential for improved initialization strategies and feature learning. Future work will explore deeper theoretical connections and applications to more complex architectures and datasets.

References

F. Aurenhammer. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991. doi: 10.1145/116873.116880.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18, 2000. doi: 10.1016/S0169-7439(99)00047-7.

Thomas G. Dietterich. Ensemble methods in machine learning. In Tin Kam Ho, editor, *Multiple Classifier Systems*, pages 1–15. Springer, Berlin, Heidelberg, 2000.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 249–256, 2010.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edition, 2009.

Ian T. Jolliffe. *Principal Component Analysis*. Springer, 2 edition, 2002.

Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, pages 9–48. Springer, 2012.

S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28 (2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55, 1936.

G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley & Sons, New York, NY, USA, 2000.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 807–814, 2010.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision (ECCV)*, pages 818–833. Springer, 2014.