Insights on the Developmental Trajectory of the Posterior Ventral Pathway (PVP) from a Feed-Forward Neural Network (FFNN)

1. Background & Motivation

1.1 Tracts connecting ventral and dorsal streams - posterior vertical pathway

The posterior vertical pathway (PVP) is a group of at least four vertical white matter tracts anterior to the occipital cortex that directly link cortical areas associated with the ventral and dorsal streams (Figure 1.1). It is a crucial pathway for communication between the temporal and parietal cortices. Two of the PVP tracts connect the posterior ventral temporal cortex to either the inferior parietal cortex (i.e., the Posterior Arcuate, pArc) or the superior parietal cortex (i.e., the Temporal Parietal Connection to the Superior Parietal Lobe, TP-SPL). The remaining two tracts connect the anterior ventral temporal cortex to either the inferior parietal cortex (i.e., the Middle Longitudinal Fasciculus to the Angular Gyrus, MDLFang) or the superior parietal cortex (i.e., the Middle Longitudinal Fasciculus to the Superior Parietal Lobe, MDLFspl). Tracts within the PVP have largely remained unexamined, particularly in terms of their development across the lifespan.

An ongoing study focuses on the development of these white matter tracts. This insight may reveal how perception-action communication within the visual information processing streams evolves throughout development. Understanding the interplay between the dorsal-ventral visual processing streams is important for studying activities such as handwriting and tool use where the action depends on perceptual guidance.

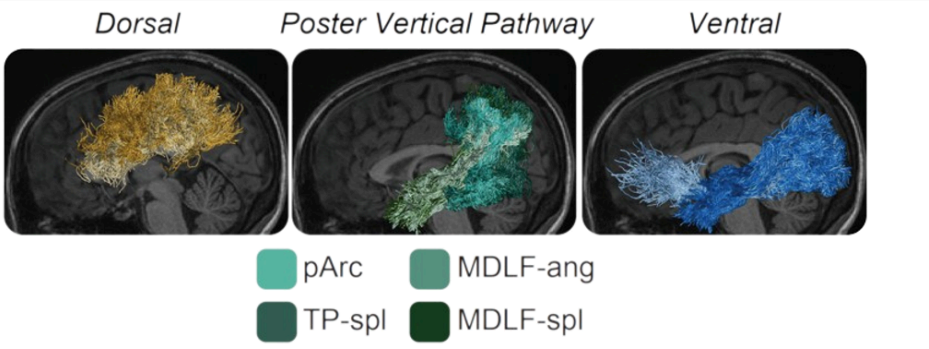

Figure 1.1: The dorsal and ventral visual streams are associated with brain function occurring in the gray matter areas of the brain (cortex), but they are connected through the bundles of axons that comprise the white matter areas of the brain. The white matter tracts that directly connect cortical regions associated with ventral and dorsal stream processing are referred to as the Posterior Vertical Pathway (PVP). The PVP can be segmented into at least four white matter tracts: (1) Posterior Arcuate, pArc, (2) Temporal Parietal Connection to the Superior Parietal Lobe, TP-spl, (3) Middle Longitudinal Fasciculus to the Angular Gyrus, MDLFang, and (4) Middle Longitudinal Fasciculus to the Superior Parietal Lobe, MDLFspl.

1.2 Tract Profiles Analysis

Tract profiles were generated using the diffusion metric fractional anisotropy (FA) along the length of each segmented white matter fascicle (Figure 1.2.1). FA is a catch-all, summary measure of white matter tissue microstructure that is related to white matter integrity. It is correlated with myelin in addition to other tissue properties such as fiber density and axonal diameter. FA indicates when a microstructural feature along a tract has changed and is used commonly in studies targeting the development of white matter microstructure.

Tract profiles were generated for each of the four PVP tracts. The VOF was also included in this analysis for a comparative study of the development and microstructural properties between the PVP and another major vertical tract. This can help establish whether observed patterns in the PVP are unique or part of a broader trend among vertical tracts.

Using a service on the cloud-computing neuroimaging platform Brainlife.io, 1.5 million streamlines were generated per subject, represented in Figure 1.2.1 by gray lines. Each streamline is one piece of evidence that a major white matter tract exists at particular coordinates. The highest likelihood of a major tract's presence is represented by the centroid of these aggregated streamlines. Along 200 equally-spaced nodes on this centroid, FA measurements were calculated (in the final results, the first and last 20 nodes were excluded to avoid partial volumizing effects). To obtain a tract group mean, FA values for each tract were averaged across the core values.

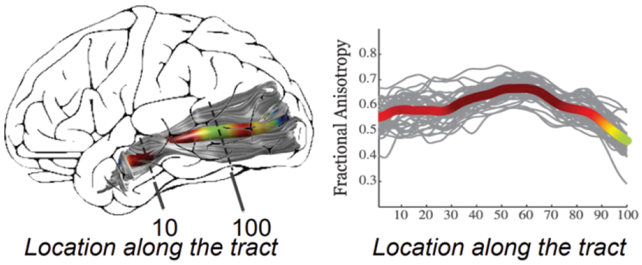

Figure 1.2.1: Tract profiles analysis of the posterior ventral pathway (PVP) white matter tracts. The left image shows the location of tract nodes along the length of the segmented tract, while the right plot illustrates the fractional anisotropy (FA) values measured at each node. Gray lines represent individual streamlines generated to approximate the major white matter tracts, with the central red line indicating the centroid, where FA measurements are most likely to reflect the core tract structure.

In Figure 1.2.2, FA is displayed on the y-axis, and location along the tract or the tract 'core'/node of a specific FA measurement is displayed on the x-axis.

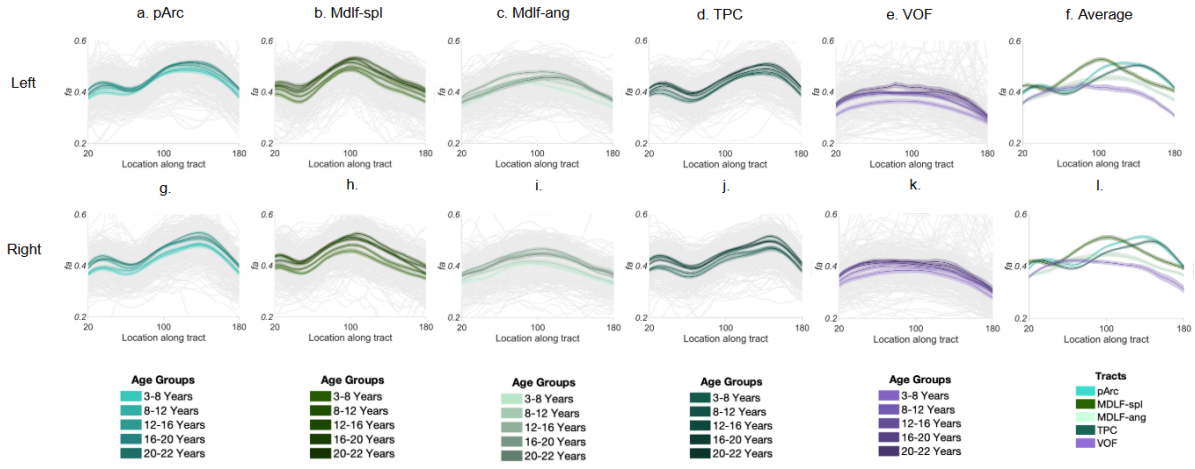

Figure 1.2.2: Tract profiles analysis. (a-h) FA is displayed on the y-axis; the location along the tract 'core' is displayed on the x-axis. The solid line corresponds to the mean tract profile and the shaded areas correspond to 95% confidence intervals. Each gray line in the background represents an individual subject.

1.3 Developmental Trajectory Analysis

From the tract profiles, to obtain a tract group mean, the mean FA values for tracts included in that pathway were averaged across the core FA values. For example, for subject A's tract profile for pathway A, the FA measurements for all 160 equally-spaced nodes were averaged to obtain one mean FA value. Then, for all subjects ($n \approx 400$), for each of 4 PVP tracts on the left and right hemispheres, the subject's mean FA value for each tract was plotted against the age of the subject on the x-axis. This generated plots that displayed the relationship between FA and age.

We modeled the relationship between age and FA as: $FA = C + Ae^{\frac{-age}{t}}$; this model was used successfully with a similar age range, i.e., 5-30 years of age (Lebel et al., 2008).

We completed an asymptotic analysis in which the maximum development was identified as the FA values where the exponential curves leveled off (i.e. the horizontal asymptote represented mathematically in the model by the parameter C). For each tract, we calculated the time it took to reach 90% of this maximum starting from the minimum age of the fitted model, as in Lebel et al. (2008).

To identify the age at 90% maximum FA for each tract of interest, we calculated the range of FA by subtracting the maximum FA value from the minimum FA value of the fitted model: $yrange = max(y) - min(y)$. We then multiplied yrange by 0.90 and added that value to min(y) to identify the FA measurement that corresponded to 90% of the maximum FA for that tract based on the fitted model : $yat90 = 0.90(yrange) + min(y)$. Finally, we evaluated the age at which the FA value at 90% of maximum development was reached: $xat90 = eval(x)\ at\ yat90$. This final value, $xat90$, was used as a summary measure of the relationship between microstructure and age for each tract of interest. Smaller $xat90$ values indicate tracts that reach adult-like FA values earlier while larger $xat90$ values indicate tracts that reach adult-like FA values later.

Current results indicate that the PVP tracts have different developmental trajectories (Figure 1.3). Most tracts showed significant differences; almost all comparisons yielded significant p-values (<0.01), indicating that the age of peak FA differs significantly between most of these tracts and suggesting that these tracts follow unique trajectories.

We found no significant differences between the left TPC vs. right TPC; this suggests that the TPC in both hemispheres mature around the same age and has bilateral symmetry in its development. Similarly, the left MDLF-spl vs. left pArc (p = 0.0015), right MDLF-spl vs. right pArc (p = 0.3772), left TPC vs. right pArc (p = 0.0054), left TPC vs. right MDLF-spl (p = 0.019), and right MDLF-spl vs. right TPC (p = 0.0051) reach adult-like FA values at similar ages, indicating similar relationships with age in the timing of their peak FA.

The left VOF appears to have the earliest age at peak FA (11.296 years), while the right pArc and right MDLF-ang have relatively later peak ages at around 18 years.

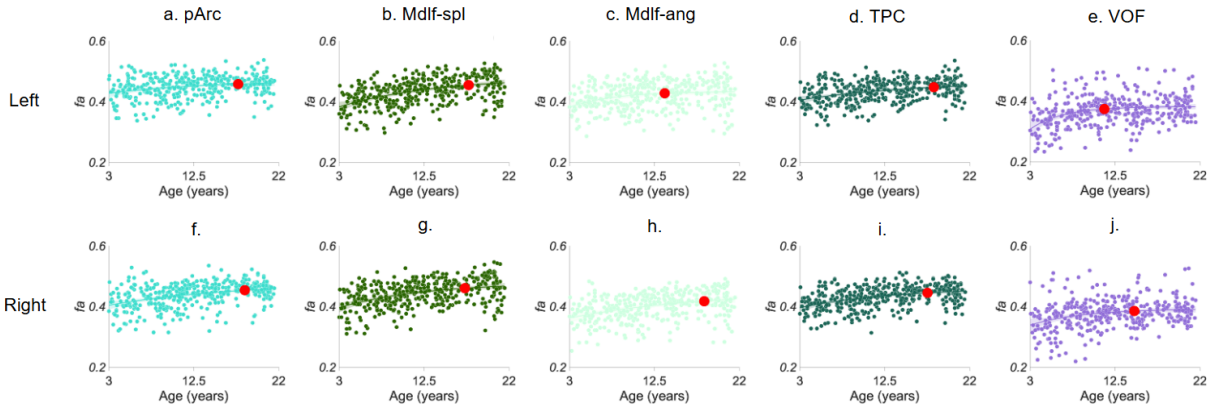

Figure 1.3: Developmental trajectories. (a-j) ***To illustrate the growth trajectories of each tract, mean***

FA values for each subject for each tract of interest showing increases in FA growth rate were plotted against age. Mean FA is displayed on the y-axis; age of the participant is displayed on the x-axis. Each dot corresponds to one participant. The exponential model from Lebel (2008) was added to illustrate the developmental trajectories of FA in each tract.

1.4 Motivation for Computational Model

Traditional neuroimaging studies have provided valuable data on tract development across age groups, but predictive models using neural networks are underexplored in this area. Unlike traditional curve-fitting methods (e.g., exponential functions), neural networks can capture more complex patterns and outliers in the data; they have greater flexibility in capturing nonlinear growth trajectories or individual differences across subjects and exploring dependencies and interactions between tracts. If successful, a neural network could also classify tracts based on their developmental trajectory profiles by automatically learning features that distinguish tracts instead of relying solely on predefined comparison metrics like the age at 90% of maximum FA ($xat90$). Also, for cross-sectional data, as used in this study, the network could predict developmental stage or maturity based on current FA profiles.

Although not pursued in this project, a neural network could analyze node-level FA values to identify which parts of each tract contribute most to developmental changes or functional outcomes. Techniques like saliency maps or feature importance analysis on the neural network can highlight critical tract regions.

This project involves designing and training a feed-forward neural network (FFNN) that can work with cross-sectional diffusion MRI data to predict FA in the PVP tracts based on age, sex, and node position. This problem is a regression task where the goal is to predict a continuous numerical value (FA) based on input data by accurately modeling the non-linear relationships between the input features and FA values and by minimizing prediction errors during training and validation.

2. Methods

2.1 Dataset Characteristics

We accessed available data from two large-scale, open-source, cross-sectional neuroimaging datasets: the Healthy Brain Network (HBN, aged 5–21 years; Alexander et al., 2017) and the Pediatric Imaging, Neurocognition, and Genetics (PING, aged 3 to 20 years; Jernigan et al., 2016) datasets. In this work, the data was grouped into smaller bins based on the participant's age. These bins were: [3-8] years; (8-12] years, (12-16] years, and (16-22] years.

HBN: We downloaded Release 1.1 through Release 11 of the HBN dataset in January 2023 (n = 100) from the website of the Child Mind Institute. Recruitment included flyers, posters, and outreach to local schools, community organizations, pediatric healthcare centers, and mental health services in the New York City metropolitan area. The HBN cohort consists of both neurotypical individuals and those with mental health or neurodevelopmental conditions like ADHD, anxiety, and autism. For more information on initial participant screening, inclusion/exclusion criteria, and assessments, refer to Alexander et al. (2017). In the current work, we excluded participants with no diagnosis of psychiatric disorders so that the sample accurately represents a neurotypical population. From HBN, 84 participants were used as input for the subsequent processing after excluding data for quality concerns.

PING: We accessed the PING dataset through the NIMH Data Archive (NDA) in Month Year (n = 401). The PING dataset is part of a collaborative project led by multiple institutions with support from the National Institutes of Health (NIH), primarily funded by the National Institute on Drug Abuse (NIDA) and the Eunice Kennedy Shriver National Institute of Child Health and Human Development (NICHD). The PING study recruited participants through schools, community outreach events, and partnerships with local organizations and clinics in the greater

metropolitan areas of Baltimore, Boston, Honolulu, Los Angeles, New Haven, New York, Sacramento, and San Diego. The dataset includes neurotypical children and those with neurodevelopmental disorders such as ADHD who were all fluent in English. From PING, 387 participants were used as input for the subsequent processing after excluding data quality concerns.

2.1 Implementation & Model Environment

The neural network model was implemented in MATLAB's Deep Learning Toolbox, specifically with the Deep Network Designer GUI. This toolbox provides a comprehensive environment for designing, training, and evaluating neural networks, with the flexibility to integrate custom preprocessing pipelines and advanced visualization tools. The architecture was defined interactively within the Deep Network Designer. All code for this project can be found at:
https://github.com/Zoha-Arif/PVPDevelopmentProject

3. Computational Model Design

Since the data is cross-sectional, a recurrent neural network designed to handle sequential data or data with a temporal dependence is not an optimal model for this process. Each participant is scanned only once at a particular age, so we do not have the data for the network to learn dependencies across multiple time steps for each individual. Age is simply a feature as opposed to a sequence that the model needs to track over time. Therefore, an FFNN with multiple hidden layers and non-linear activations will likely better directly learn the relationships between input features and FA.

3.1 Input Layer

The model takes a 3-dimensional input vector where the features are as follows:

1. *Age:* a discrete, normalized value representing the age of the participant.
2. *Sex:* a categorical variable (one-hot encoded) to account for sex-based differences in FA development.
3. *Node position:* a discrete value representing the position along the tract's centroid (see Section 1.2)
    a. This is needed as input because FA values vary at different points along the length of the white matter tract. The model needs to learn location-specific FA patterns along each tract as well.

3.2. Hidden Layers

The model will include three fully connected hidden layers as follows:

1. Layer 1: 64 units, ReLU activation.
    a. This layer will learn high-level relationships (such as non-linear age effects, localized developmental differences, and sex-based variations) between input features and FA.
2. Layer 2: 32 units, ReLU activation.
    a. This layer reduces the dimensionality by having fewer units.
    b. The network is encouraged to filter out redundant or less significant information; it is forced to represent the data in a more compact, abstract form, where each neuron in Layer 2 holds information distilled from a broader set of interactions learned in Layer 1.
3. Layer 3: 16 units, ReLU activation.
    a. This layer further reduces the dimensionality of the learned patterns.

3.3 Output Layer

The output layer produces a single FA prediction for the given input (age, sex, and node position). A ReLU activation function will be used since FA is a continuous variable and since sigmoidal units present the vanishing and exploding gradient problem.

3.4 Hyperparameters

A Levenberg-Marquardt backpropagation training algorithm was used. This is an advanced variation of the standard backpropagation algorithm that switches from using the gradient descent approach when the solution is far from the optimum and Gauss-Newton optimization when the solution is near the optimum.

Mean Squared Error (MSE) was used as a performance metric because FA is a continuous variable and this metric is standard for regression tasks.

The learning rate was initially set to 0.001 and adjusted later after experimentation to optimize convergence.

The initial trial ran 100 epochs. After experimentation, early stopping criteria were enabled based on validation loss.

The initial batch size was set to 32 to balance computational efficiency and gradient stability.

4. Results

4a. Trial 1

In Trial 1, we initialized and trained the model architecture detailed in Section 3. However, after approximately 1 hour, the training process had only completed 2 epochs. Due to the slow convergence rate, we stopped training prematurely.

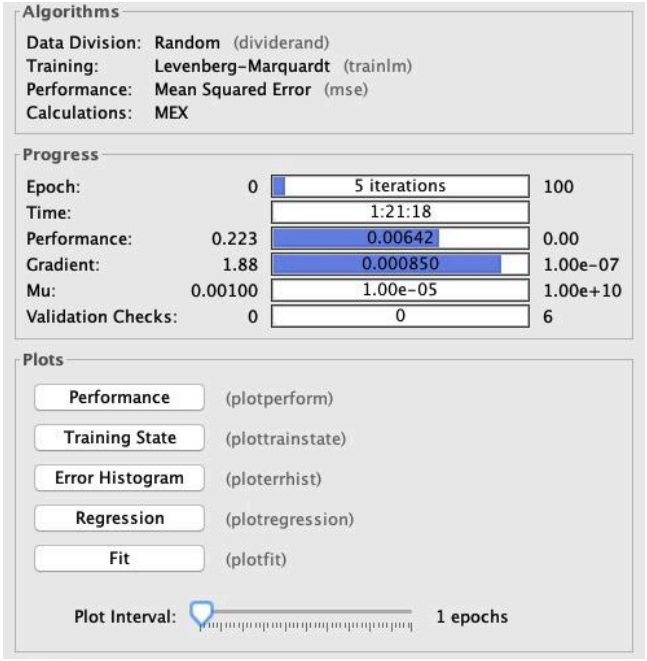

Figure 4.1.1: ***Results from the GUI provided by MATLAB's Deep Learning Toolbox for training and evaluating neural networks. Mu (μ) indicates the level of damping in the Levenberg-Marquardt algorithm—a smaller μ value suggests the optimization is moving toward convergence. There were no validation checks as training stopped prematurely.***

A small gradient (~0.00085) suggests that the network is converging effectively (Figure 4.1.1). The mean squared error (MSE) on the training data was 0.006452, which is close to zero and indicates a

promising level of accuracy. However, based on predefined performance criteria, an MSE of 0.005 or lower was considered an optimal threshold for this model.

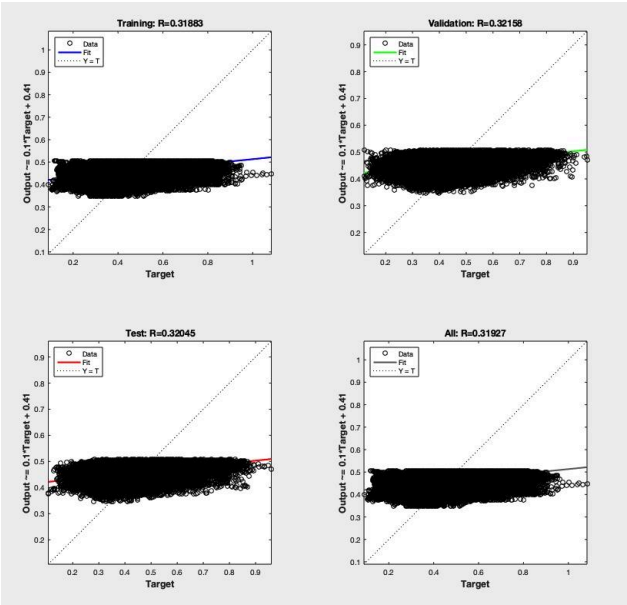

Figure 4.1.2: Regression Plot ***The regression plot evaluates the alignment between the predicted values (outputs) and the actual target values (FA) by visualizing their correlation. The R-values quantify the strength and direction of this relationship, ranging from -1 (perfect negative correlation) to +1 (perfect positive correlation). These values represent the correlation coefficient between the network's predictions and the true target values. The x-axis (true FA values) represents the actual FA values from the dataset for each input, i.e., the ground truth values of the dependent variable being predicted. The y-axis (predicted FA values) represents the corresponding FA values predicted by the neural network after processing the inputs through its architecture. The diagonal dotted line represents the ideal case, where the network's predictions match the true values perfectly; the points along this line indicate that the prediction equals the target for a particular data point. A perfect R-value of 1.0 would mean all points lie exactly on this line. Training data (top-left) shows the fit of the data used to train the network. This fit typically has the best r-value since the network has directly learned from this data. Validation data (top-right) shows the fit on a hold-out set used to evaluate the model during training. This fit indicates how well the model generalizes to unseen data during training. Even though there were no validation checks due to training being topped prematurely for Trial 1, the validation set was still used to generate performance metrics/to see how well the current model generalizes to the validation set. Test data (bottom-left) shows the fit on completely unseen data used to evaluate the final model. This is the most important panel for assessing generalization performance. All data (bottom-right) combines all subsets (training, validation, and test) to give an overall picture of the network's performance.***

From Figure 4.1.2, we see that the regression plots show low correlation (R ~ 0.36), suggesting that the model has not captured strong relationships in the data. We also see evidence that the network is not overfitting as overfitting would typically show a tight fit in the training plot (high R) but poor fit in validation and test plots (low R). The low R-values are evidence that the network is underfitting possibly because (1) the data does not have strong linear or non-linear patterns or (2) features and the network architecture need more complexity to capture more complex patterns.

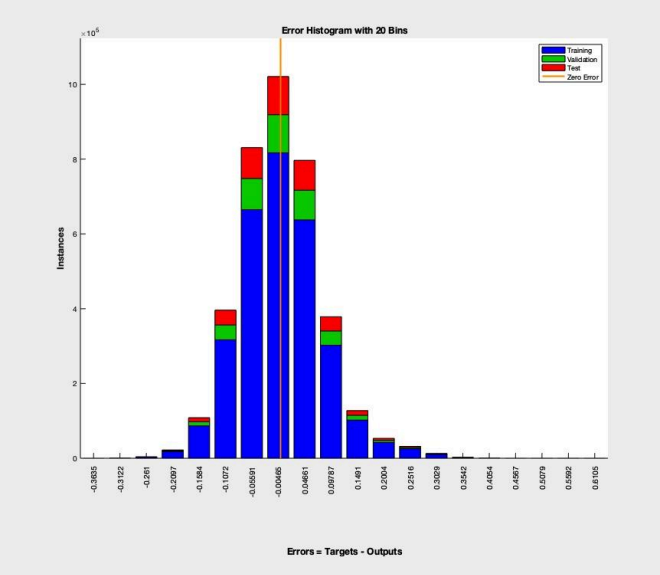

Figure 4.1.3: Error Histogram ***This histogram shows the distribution of errors, where errors are calculated as the difference between the target values and the predicted outputs of the neural network, e.g.,*** $error = target - output$***. The x-axis represents the error values (e.g., positive, negative, or zero), while the y-axis shows the frequency (intensity) of those errors in the training, validation, and test datasets. The bars are color-coded to distinguish the datasets (e.g. training, validation, or test).***

The histogram is approximately symmetric around 0 on the x-axis, suggesting that the network's predictions are not biased toward systematically overestimating or underestimating target values. The error histogram provides a limited view of model performance and fails to reveal the underfitting highlighted in the regression plots since the errors are uniformly small.

Most of the errors being concentrated near zero also indicates that the model's predictions are generally accurate for the majority of data points across all datasets (e.g. training, validation, and testing).

The presence of wider error bars farther from zero suggests a small number of outliers where the network's predictions deviate more significantly from the target values. The prominent bar at 0 indicates that many data points across all datasets have very low or no prediction error.

The color-coded bars show that errors are consistently distributed across the training, validation, and test sets. This indicates that the model generalizes well and has not overfitted to the training data.

The majority of errors are small, but there are a few larger errors (longer tails on the left and right). Training, validation, and test errors overlap significantly, which, again, indicates no severe overfitting. If overfitting were present, the training error distribution would be tightly clustered near 0, but validation/test errors would be more spread out. Here, the error distributions for all subsets are comparable.

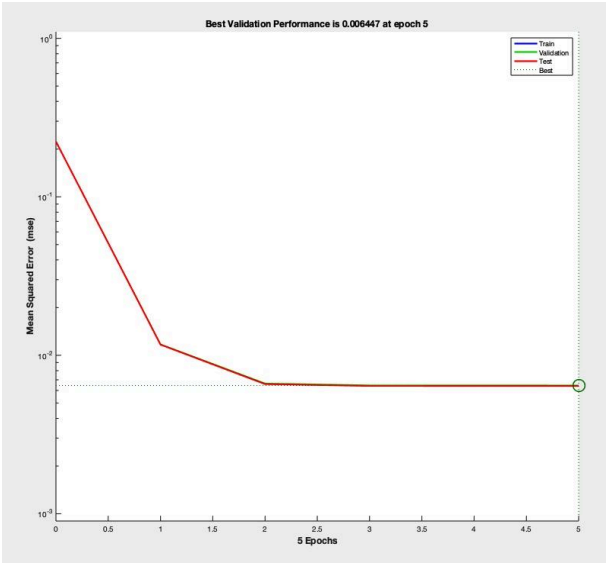

Figure 4.1.4: Validation Plot ***This plot shows the MSE as a function of the number of training epochs for a neural network. The x-axis indicates the number of epochs the training algorithm has made through the entire dataset. The y-axis represents the MSE values, plotted on a logarithmic scale to visualize both large and small changes in the error. A circle marks the epoch with the lowest validation error, which is recorded as ~0.006447 at epoch 5.***

The training and validation errors decreased rapidly during the initial epochs, indicating that the model is learning and fitting to the data. After epoch 3, the errors plateau, suggesting that further training does not yield significant improvements. The validation error closely follows the training error, indicating that the model generalizes well to unseen data. There are no signs of overfitting (e.g., where the validation error starts to increase while training error continues to decrease). The test error is consistent with the training and validation errors, further confirming that the model generalizes well to unseen data.

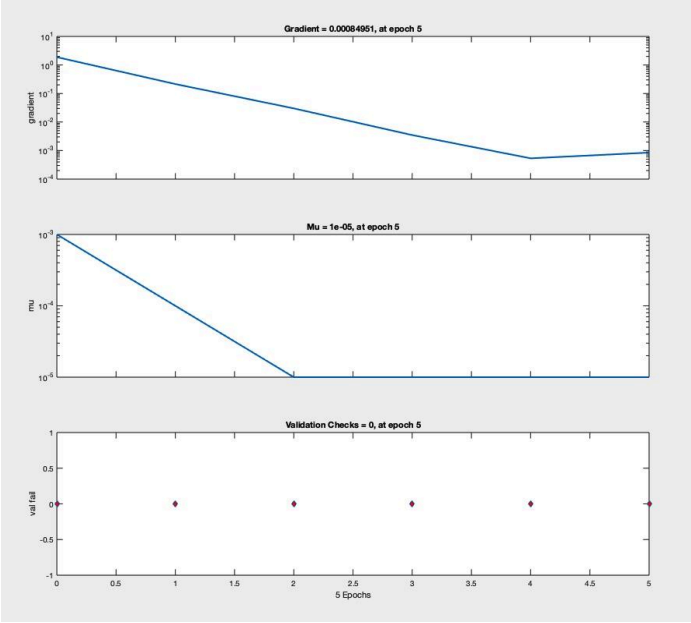

Figure 4.1.5: Gradient, Mu, and Validation Checks ***This plot consists of three subplots, which provide insights into the training process of the neural network by analyzing the gradient, MSE, and validation checks over the epochs. (Top) The x-axis represents the number of epochs, and the y-axis represents the magnitude of the gradient. (Middle) The x-axis represents the number of epochs, and the y-axis represents the value of μ, the adaptive learning parameter. (Bottom) The x-axis represents the number of epochs, and the y-axis represents the number of failed validation checks.***

The gradient decreases steadily over the first few epochs, reaching a very small value (~0.00034831 at epoch 5). A small gradient magnitude indicates that the model is converging and that the optimization process is stabilizing. After epoch 3, there is no significant gradient update, suggesting that the model has reached a near-optimal solution.

μ is the damping parameter of the optimization algorithm. It decreases over the first few epochs which means that the algorithm has switched from "global search" to "local convergence." When μ is high, the algorithm is in the global search phase where it takes smaller, safer steps to avoid overshooting the minimum in the early training stages when the loss surface might be highly irregular or far from the optimal solution. When μ is low, the algorithm transitions to Gauss-Newton approximation, which assumes a smoother loss surface around the local minima. In this stage, the algorithm takes larger, more confident steps, focusing on local convergence. At the start, μ is high because the model needs to explore the loss surface and identify promising regions to optimize. Over the first few epochs, as the model moves closer to a good region (where the loss decreases significantly), μ decreases. This therefore indicates that the algorithm is transitioning from exploring the global landscape to refining the solution within a smaller, localized area. The decrease in μ indicates successful progress in the optimization process. If μ stayed high throughout training, it might indicate that the algorithm is struggling to find a good region and is stuck in the global search phase.

The validation check value remains at 0 throughout all epochs. Validation checks are triggered when the validation performance fails to improve for a predefined number of epochs. No validation checks occurred since training was forced to stop prematurely.

Taken together, the model isn't capturing strong patterns in the data and the network appears to stabilize quickly by epoch 3.

4b. Trial 2

Based on the performance metrics of Trial 1, the following adjustments were made:

1. **Increased Model Complexity:** The performance metrics from Trial 1 suggests that the network is not overfitting, but underfitting/struggling to model the underlying relationship between inputs and outputs. The features might not fully represent the complexity of FA.
    a. We added an interaction term (e.g., Age × Node Position) to model complex dependencies between features.
        i. The interaction term was computed directly from the input features before feeding the data into the neural network. The product of the Age and NodePosition variables was calculated:
        $InteractionTerm = Age .* NodePosition$ where .* performs element-wise multiplication for every sample in the dataset.
        ii. This new feature (InteractionTerm) was added to the matrix of input features.
    b. We added 2 more hidden layers and increased the number of units in each layer to allow the network to capture complex relationships and learn more higher-order representations of the data.

2. **Added Dropout Layers:** To prevent overfitting, we introduced dropout layers after each hidden layer. A dropout layer is a regularization technique used to improve generalization. It works by randomly "dropping out" (i.e., setting to zero) a fraction of the neurons in the layer during training, along with their corresponding connections.
3. **Decreased Number of Epochs to 10:** The validation plot showed that, after epoch 3, the errors plateau, indicating that further training does not yield significant improvements. So, we lowered the number of epochs from 100 to 10.
4. **Enabled Validation Stopping:** To avoid unnecessary epochs, we allowed the training to stop early once the validation performance stopped improving.

The new architecture was as follows:

a. Input Layer
The model takes a 4-dimensional input vector where the features are as follows:
1. *Age:* a discrete, normalized value representing the age of the participant.
2. *Sex:* a categorical variable (one-hot encoded) to account for sex-based differences in FA development.
3. *Node position:* a discrete value representing the position along the tract's centroid (see Section 1.2)
4. *Age × Node Position (Interaction Term):* A continuous feature that models the combined effect of age and node position on FA development to capture location-specific, age-related patterns along the tract.

b. Hidden Layers
The model for Trial 2 included 5 fully connected hidden layers as follows:
1. Layer 1: 128 units, ReLU activation
2. Layer 2: 64 units, ReLU activation
3. Layer 3: 32 units, ReLU activation
4. Layer 4: 32 units, ReLU activation
5. Layer 5: 16 units, ReLU activation

c. Output Layer
The output layer produces a single FA prediction for the given input (age, sex, node position, interaction); it uses the ReLU activation function.

After 11 hours of training, the training process had only completed 2 epochs. Due to the slow convergence rate, we stopped training prematurely. Unfortunately, the GUI then crashed, so we were unable to obtain performance metrics from this trial.

Figure 4.2.1: ***Results from the GUI provided by MATLAB's Deep Learning Toolbox for Trial 2.***

4c. Trial 3

To optimize the computational efficiency of the network architecture described in Trial 2 (Section 4b), we used the fitnet (shallow neural network) instead of trainNetwork (deep learning) function in Matlab. The fitnet function is designed for smaller, shallower neural networks and is computationally less expensive compared to deep learning networks that use trainNetwork. This reduced training time significantly.

We employed the Adam optimizer, a computationally efficient optimization algorithm that adjusts the learning rate dynamically based on first-order and second-order moments. The use of the Adam optimizer enabled faster and more stable convergence during training.

We also used a higher initial learning rate (InitialLearnRate = 0.01) for quicker convergence and enabled GPU acceleration.

Instead of using the default fitnet training with full datasets, we switched to mini-batch training (a technique commonly used in stochastic gradient descent) in which the training dataset is divided into small, manageable subsets, instead of using the entire dataset at once. Each mini-batch is used to update the model's weights during one iteration of training.

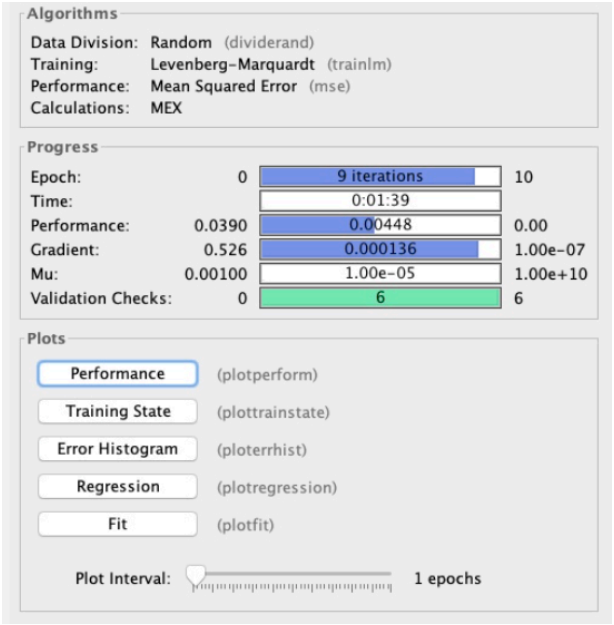

Figure 4.3.1: ***Results from the GUI provided by MATLAB's Deep Learning Toolbox for Trial 3.***

Training ran for 9 epochs out of the maximum of 10 and completed in 1 minute and 39 seconds. The validation MSE decreased significantly from 0.0390 at epoch 0 to a final value of 0.00448 at epoch 9. The gradient magnitude reduced from 0.526 to 0.000136, indicating that the optimizer made progressively smaller weight updates as the model converged. Simultaneously, the μ parameter, which governs the step size in the Levenberg-Marquardt algorithm, decreased from 0.001 to 10^{-5}, reflecting a transition from coarse to fine-tuned adjustments. However, six validation checks were triggered, suggesting that the validation error failed to improve for six consecutive epochs, leading to early stopping at epoch 9.

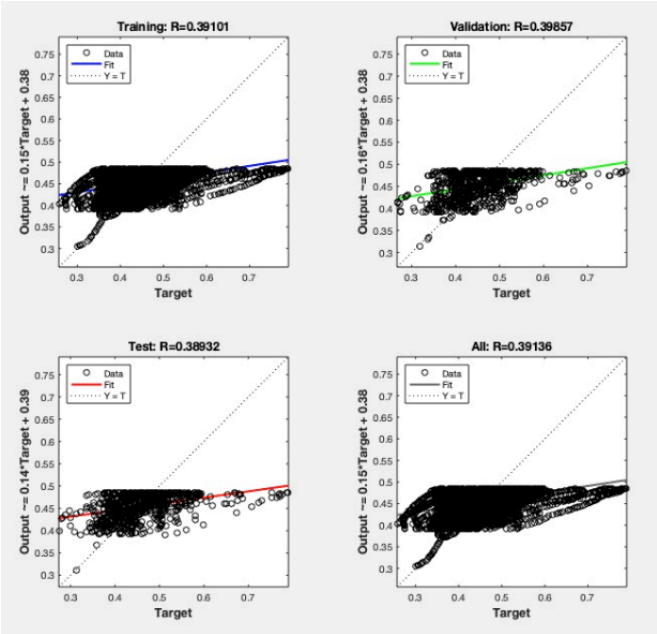

Figure 4.3.2: Regression Plot for Trial 3.

The R-values across all datasets are relatively low (~0.39), indicating a weak correlation between the predicted outputs and the actual target values. This suggests that the model is still underfitting the data and has not captured the underlying relationships effectively. The dotted diagonal line represents the ideal case where predictions match the true values exactly. Most points deviate significantly from this line, particularly in the lower and upper ranges of the target values, indicating poor predictive performance. While the model has attempted to fit the data, the low correlation reflects a lack of complexity in the model architecture or insufficient training. The validation performance is similar to the training performance, with a weak fit to the validation data. This suggests that the model generalizes poorly and struggles to learn meaningful patterns from unseen data. The test performance mirrors the training and validation performance, confirming the underfitting issue. Poor generalization is further evident as the model fails to perform well on the independent test set. The combined performance across all datasets reinforces the conclusion that the model does not effectively capture the relationship between the inputs and outputs. Overall, the low R-values indicate that the model is unable to capture the complexity of the data.

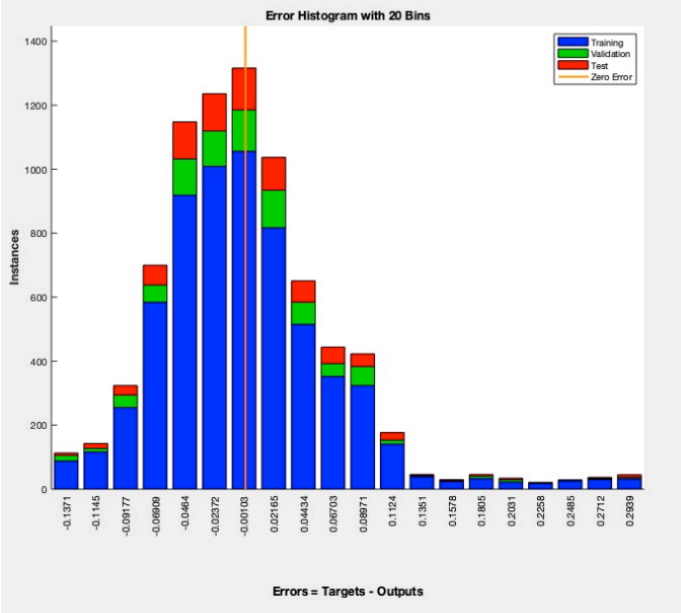

Figure 4.3.3: Error Histogram for Trial 3

Most errors are concentrated between −0.05 and +0.05, indicating that the majority of predictions are relatively close to the target values. However, the presence of larger errors in the tails (e.g., ≥ 0.15 or ≤ −0.15) suggests the model struggles with certain data points. The similarity in error distribution for the training, validation, and test sets suggests that the model generalizes reasonably well across unseen data and is not overfitting. Peaks near zero indicate a good number of predictions are very accurate, as they fall close to the ideal zero error line. The histogram shows a notable spread, which aligns with the earlier regression plots indicating that the model underfits the data.

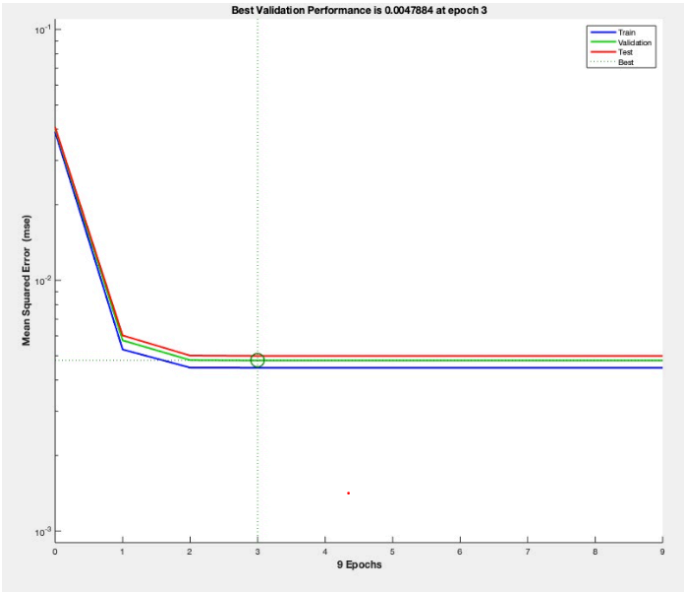

Figure 4.3.4: Validation Plot for Trial 3

The model converges quickly, with no significant improvement in performance after epoch 3. This suggests that the model requires relatively few epochs to achieve its best performance. The validation error does not increase after the training error plateaus, indicating that the model generalizes well to unseen data. The alignment of training, validation, and test errors reflects that the model is not overfitting. However, the earlier regression and error histogram plots, along with weak correlation values, suggest that the model may be underfitting. While the MSE values are relatively small, indicating the model produces predictions close to the target values on average, these predictions do not strongly capture the underlying data relationships. This points to a case where the model fits the data with low error magnitudes but lacks sufficient complexity to fully learn the patterns in the data.

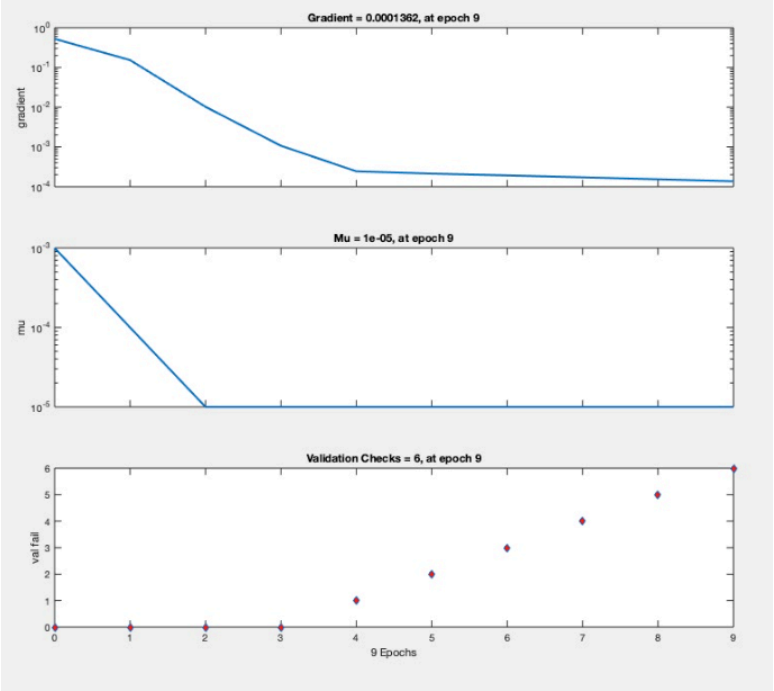

Figure 4.3.5: Gradient, Mu, and Validation Checks

In the top panel, the gradient magnitude decreases steadily, reaching a value of 0.0001362 by epoch 9, which indicates that the optimization process is stabilizing, and the model's weights are no longer undergoing significant updates. In the middle panel, the adaptive learning rate parameter μ decreases rapidly by epoch 3, after which it remains constant. This behavior suggests that the optimizer quickly transitioned from a global search phase to a local refinement phase, focusing on fine-tuning the model weights. In the bottom panel, the validation checks, which are triggered when validation performance fails to improve, begin accumulating after epoch 3 and reach a total of 6 checks by epoch 9. This indicates that the model stopped improving on the validation set early in the training process, signifying a plateau in validation performance. Taken together, these results suggest that the model converged quickly, with no further improvement after epoch 3. While this reflects stability in the training process and no overfitting, the early stagnation of validation performance and rapid decrease of μ suggests the model may be underfitting and struggling to fully capture the underlying data relationships.

4d. Trial 4

Based on the performance metrics of Trial 3, the following adjustments were made:

1. **Increased Model Complexity**
    a. We added polynomial terms (e.g., NodePositionSquared and AgeSquared) to capture nonlinear relationships as FA values vary non-linearly across a tract. The aim is that, with this polynomial term, the model can better represent and learn relationships between input variables and FA that may not be purely linear or additive.
        i. The squared version of the NodePosition feature was computed using: $NodePositionSquared = NodePosition^2$.
        ii. The new feature NodePositionSquared was added as a column in the input matrix alongside other features and treated as an additional input to the network.
        iii. AgeSquared was calculated similarly using the Age feature.
    b. We increased the number of nodes in the hidden layers, adjusting the configuration to 128 → 64 → 32, thus further enhancing the model's capacity to capture complex data relationships.
2. **Set Moderate Rates for Dropout Layers:** We kept dropout layers with moderate rates of 0.3 and 0.2 after each hidden layer. As the model becomes more complex, there is a risk that it might start overfitting the training data. Incorporating dropout layers with moderate rates ensure that the model does not overfit.
3. **Applied Batch Normalization:** We applied batch normalization after each fully connected layer to improve convergence stability and address internal covariate shift. Normalizing the input to each layer within a mini-batch of data ensures that the input values to the layer have a mean of 0 and a standard deviation of 1. This helps stabilize the training process, especially with the interaction terms and non-linear features added to the network. Batch normalization also indirectly regularizes the network by adding noise to each mini-batch during training.
4. **Increased Batch Size:** We increased the batch size to 256, thus allowing the model to process more samples simultaneously, resulting in smoother gradient updates.

The new architecture was as follows:

a. Input Layer
The model takes a 6-dimensional input vector where the features are as follows:
1. *Age:* a discrete, normalized value representing the age of the participant.

2. *Sex:* a categorical variable (one-hot encoded) to account for sex-based differences in FA development.
3. *Node position:* a discrete value representing the position along the tract's centroid (see Section 1.2)
4. *Age × Node Position (Interaction Term):* A continuous feature that models the combined effect of age and node position on FA development to capture location-specific, age-related patterns along the tract.
5. Age² (Polynomial Term): A higher-order feature created by squaring the normalized age value to allow the model to capture nonlinear age effects on FA development, such as accelerating or decelerating growth patterns.
6. Node Position² (Polynomial Term): A higher-order feature created by squaring the node position to model quadratic relationships, such as curvature or positional effects on FA along the tract.

b. Hidden Layers

The model for Trial 4 included 7 fully connected hidden layers as follows:

1. Layer 1: 128 units, ReLU activation
2. Layer 2: 64 units, ReLU activation
3. Layer 3: 64 units, ReLU activation
4. Layer 4: 32 units, ReLU activation
5. Layer 5: 32 units, ReLU activation
6. Layer 6: 16 units, ReLU activation
7. Layer 7: 16 units, ReLU activation

c. Output Layer

The output layer produces a single FA prediction for the given input (age, sex, node position, interaction, node squared, age squared); it uses the ReLU activation function.

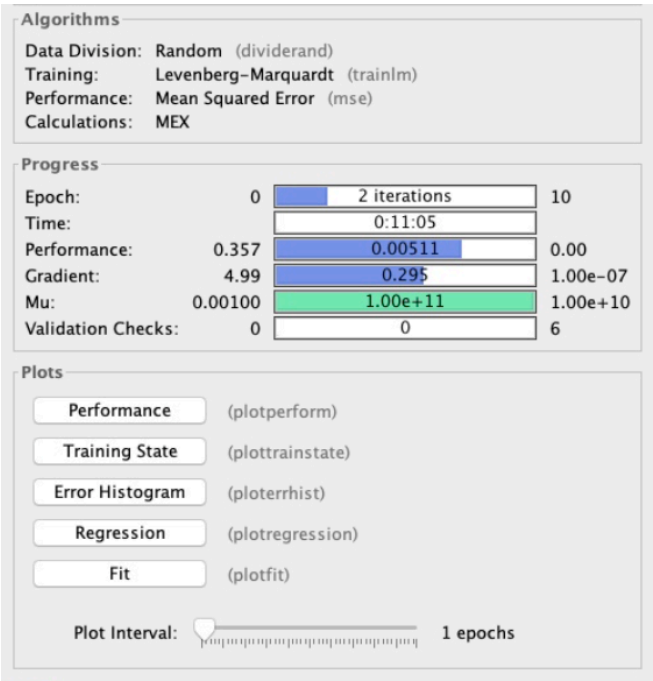

Figure 4.4.1: ***Results from the GUI provided by MATLAB's Deep Learning Toolbox for Trial 4.***

In this trial, the training has completed 2 iterations (epochs) out of the maximum of 10, and the

elapsed training time was 11 minutes and 5 seconds. The validation performance (MSE) decreased significantly from an initial value of ~0.357 to ~0.00511, showing that the model made substantial progress in reducing error. However, the gradient magnitude, which decreased from ~4.99 to ~0.295, indicates that the network is not yet fully converged as it is still above the stopping threshold of 10^{-7}. The parameter μ increased significantly to 10^{-11} which suggests that the optimizer is attempting to escape a local minimum and adjust the search space. The exceptionally high μ indicates potential instability and challenges in convergence.

No validation checks were triggered, meaning the stopping criteria were likely not because of the validation set performance, but because the model reached a performance plateau (no significant improvement in the loss metric).

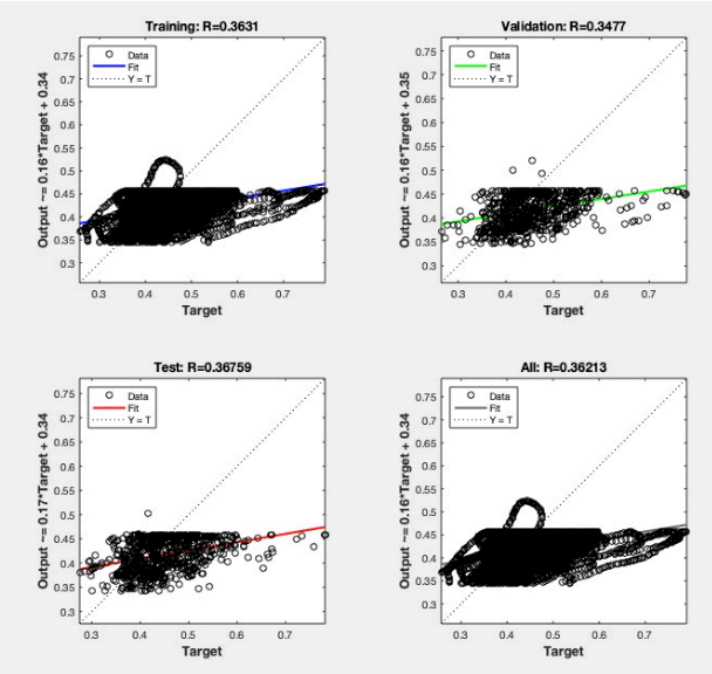

Figure 4.4.2: Regression Plot for Trial 4.

The regression plot for Trial 4 reveals weak predictive performance across the training, validation, test, and combined datasets, with consistently low R-values (R ~ 0.36) indicating a poor correlation between the model's predictions and the true target values. In the training data, R ~ 0.3631 shows that the model fails to adequately capture the underlying patterns, as evidenced by significant deviations of points from the ideal diagonal line and a poorly aligned regression fit. Similarly, the validation and test data exhibit R-values of ~0.3477 and ~0.36759, respectively, with comparable patterns of poor alignment between predicted and true values, further highlighting the model's inability to generalize effectively to unseen data. The combined data regression (R ~ 0.36213) further reinforces the conclusion that the model is underfitting, as it consistently struggles to capture the relationships between inputs and outputs across all splits. This systematic underperformance points to limitations in the model's architecture or feature representation.

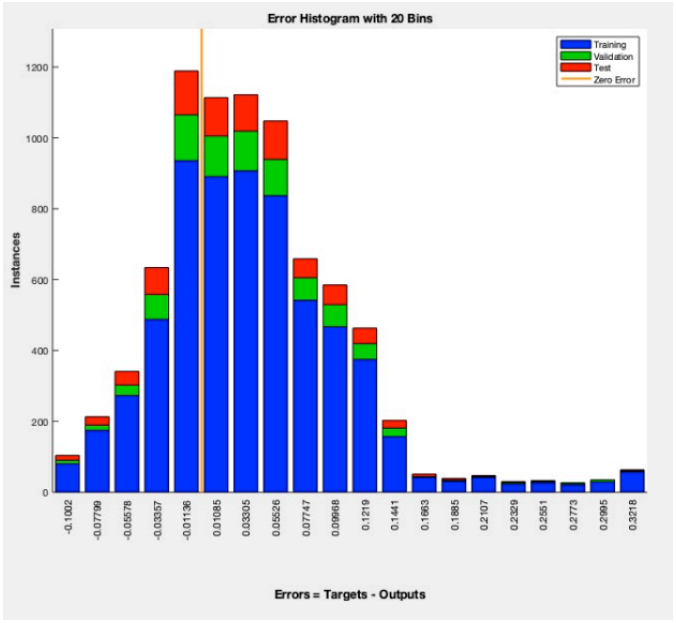

Figure 4.4.3: Error Histogram for Trial 4

The error histogram suggests that the model's predictions are reasonably centered around zero, with the majority of errors falling within the range of −0.03 to +0.03. This indicates that the model is making accurate predictions for most data points. However, the spread of errors, especially in the tails with errors exceeding ±0.1, reveals that the model struggles with certain samples or outliers. The alignment of training (blue), validation (green), and test (red) error distributions suggests consistent generalization across datasets, indicating no signs of overfitting. Considering the weak correlation values from the regression plots, however, the small error magnitudes here do not reflect the model's limited capacity to fully capture underlying relationships.

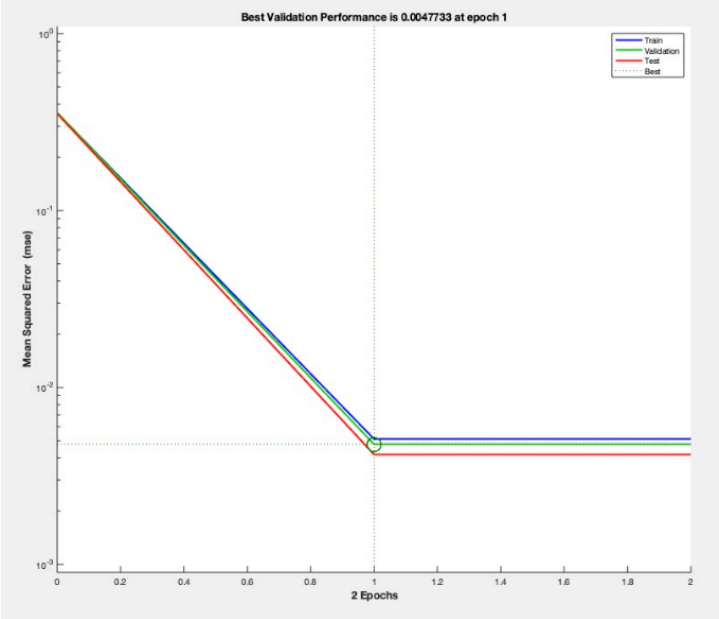

Figure 4.4.4: Validation Plot for Trial 4

This validation plot shows that the best validation performance was achieved at epoch 1, where the validation MSE was 0.0047733. After the first epoch, the training, validation, and test errors remain nearly constant, indicating that no further improvements occurred during subsequent epochs. The near-identical alignment of the training, validation, and test error lines indicates consistent generalization across the datasets, suggesting no overfitting or instability during training. However, the lack of improvement beyond epoch 1 further points to potential underfitting. This early plateau in performance suggests that while the model may achieve low error magnitudes, it might not be adequately learning complex relationships within the data.

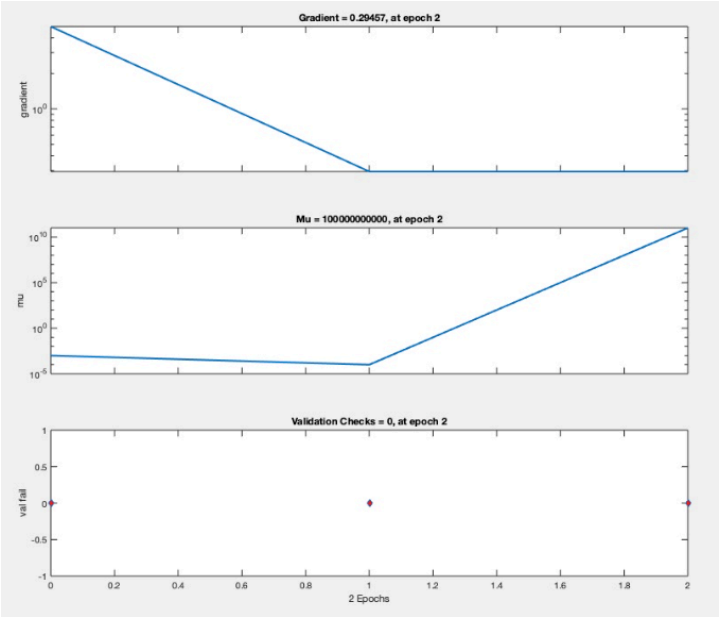

Figure 4.4.5: Gradient, Mu, and Validation Checks for Trial 4

In the top panel, the gradient starts at a relatively high value and decreases sharply to 0.29457 by epoch 2. This rapid reduction indicates that the optimization process is converging quickly. However, such a steep drop in gradient over a very short duration indicates that the model is underfitting and not leveraging the training process to learn more complex patterns in the data.

In the middle panel, the adaptive learning rate parameter μ starts at a relatively small value and then spikes dramatically by epoch 2. This sudden increase in μ typically indicates that the optimizer encountered difficulty in fine-tuning weights, potentially due to a flat or poorly defined loss landscape. This behavior suggests that the model reached a point where the optimizer struggled to make meaningful updates, further supporting evidence of underfitting and convergence issues.

In the bottom panel, the validation checks remain at 0, indicating that validation performance did not worsen during training. This is consistent with the earlier validation performance plot, which showed that the model achieved its best performance at epoch 1 and then plateaued. The absence of validation checks aligns with the lack of further improvement in performance after the initial epoch, suggesting that the model converged prematurely.

Overall, this plot suggests that while the training process was stable in terms of error reduction, the rapid convergence and spike in μ reflect underfitting and a lack of sufficient capacity to capture the complexity of the data. Improvements to the model architecture or training configuration are necessary to enhance its learning ability and avoid premature convergence.

5. Conclusion

Across all trials, the neural network consistently exhibited underfitting, indicating that the feed-forward architecture and current training configurations were insufficient to capture the complexity of the data. In Trial 3, the model converged quickly with minimal performance improvements after the first few epochs, and regression plots revealed weak correlations ($R \sim 0.39$) across all datasets, highlighting its inability to learn underlying data patterns. Trial 4 introduced adjustments, including polynomial and interaction terms, batch normalization, and increased batch size, but these changes only stabilized training without addressing the fundamental underfitting issue. The model's rapid convergence, plateauing validation performance, and a sharp spike in μ during optimization suggested premature stagnation and architectural limitations.

While interaction and polynomial terms were added, they may not fully represent the complex relationships inherent in the data. Some potentially informative features (e.g., temporal or spatial dependencies in FA) were missing, further limiting the model's ability to generalize. Simply, the shallow feed-forward network, even with added complexity (e.g., more hidden layers, larger layer sizes, interaction and polynomial terms), may not possess sufficient expressive power to capture higher-order interactions and nonlinearities in this project's data.

To improve performance, a shift to a more advanced architecture, such as convolutional neural networks (CNNs) for spatial features, recurrent networks (RNNs or LSTMs) for sequential dependencies, or transformer-based models for complex relationships, is needed. Additional strategies include refining feature engineering, incorporating transfer learning, applying alternative optimizers like RMSprop or AdaGrad, and dynamically adjusting regularization and training hyperparameters. Exploring higher-order feature transformations, such as interaction terms involving more variables or spline-based transformations for non-linear trends, and incorporating domain-specific features, such as tract-level measures or region-specific FA patterns (to add more granularity to the input data) may also result in a more optimal solution. Finally, analyzing model performance visually by generating heatmaps or saliency maps is needed toidentify underutilized features or relationships the model struggles to capture. Overall, these steps will help the model better capture the data's inherent relationships and avoid premature convergence while ensuring generalization.