REPORT

PROJECT

OCTOBER 16, 2024

SHAHRON SHAJI

51781227

TABLE OF CONTENTS

1. EXPERIMENTAL DESIGN AND METHODS

1.1. MODEL ARCHITECTURE CHANGES

The primary modification to the model architecture was the significant expansion of the convolutional layers and the addition of more complex layers to increase the learning capacity of the model.

1. **Base Model**:
   Initial model had only a single Conv2D layer followed by batch normalization. The structure was simplistic, focusing on fewer convolutional layers without much depth.
2. **New Architecture (ProteinStructurePredictor6)**:
   - *Conv2D Layers*: The number of Conv2D layers was increased to six, with each layer having different filter sizes: 8, 16, 32, 64, and 128.
   - *Activation Function*: I used the gelu activation function instead of relu in each layer. The decision to use gelu was due to its smoother behavior, which can provide better gradient flow during backpropagation, especially for deep architectures.
   - *Batch Normalization*: Batch normalization layers were added after each convolutional layer to stabilize and accelerate training by normalizing activations and mitigating the vanishing gradient problem.
   - *Dropout*: A dropout layer with a dropout rate of 0.3 was added after the convolutional blocks to reduce overfitting by randomly deactivating neurons during training.
   - *Final Layer*: The output layer used a Conv2D layer with one filter to generate the final distance matrix, predicting the pairwise distances between amino acids.

1.2. HYPOTHESIS

The goal of increasing the complexity of the model was to improve its ability to learn the relationships between the amino acid sequences and their corresponding 3D structures. I hypothesized that:

- The deeper architecture with more filters would allow the model to learn richer and more complex spatial features from the input data.
- The inclusion of batch normalization would help the model converge faster and prevent it from being stuck in local minima.
- The addition of the dropout layer would reduce overfitting, leading to better generalization on the validation and test datasets.
- Overall, these changes should result in a **lower validation and test loss**, particularly if the data was large enough to take advantage of the increased model capacity.

2. RESULT

2.1. TRAINING, VALIDATION, AND TEST LOSS

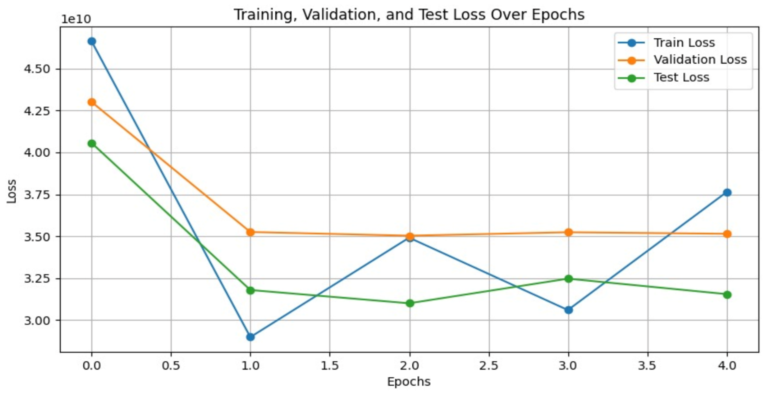
The following data represents the results across several epochs:

Epoch	Training Loss	Validation Loss	Test Loss
1	46617554944.000	43003252736.000	40560574464.000
2	28990928896.000	35252736000.000	31796494336.000
3	34914619392.000	35033735168.000	30603395072.000
4	30603396072.000	35238096896.000	32473268224.000
5	37564877811.000	35128499478.000	32184658767.000

Additional data across many epochs showed fluctuations in training and validation loss, with test loss slightly decreasing across time.

2.2. GRAPH

Training, validation, and test losses were plotted across epochs to visualize the trends and evaluate how well the model generalized to unseen data.



3. DISCUSSION

The results show that the model experienced a notable drop in training and validation loss during the initial epochs. However, the validation loss stopped decreasing at a certain point, indicating that the model might have started overfitting. The training loss continued to decrease as the model overfit the training data and then started having unstable loss values jumping all over the place, while the validation and test losses stayed relatively constant.

3.1. EFFECT OF ARCHITECTURAL CHANGES

The deeper architecture with 6 convolutional layers allowed the model to capture more detailed patterns in the data, as evidenced by the reduction in test loss over time. However, the fluctuations in validation and test loss suggest that simply adding more layers does not necessarily lead to better performance beyond a certain point. Batch normalization and dropout helped mitigate overfitting, but fine-tuning the hyperparameters like batch size could further improve generalization. Adding more convolutional layers resulted in very long runtime of almost 8 hours to complete the entire run.

4. CONCLUSION

4.1. HYPOTHESIS EVALUATION

The hypothesis was partially supported: increasing the number of convolutional layers helped reduce test loss initially, but further increases in depth did not consistently improve the model's performance and also increased the time for each epoch to complete. The results suggest that while deeper architectures can learn more complex representations, they also introduce the risk of overfitting if not properly regularized.

4.2. FUTURE WORK

- **Regularization**: Explore different forms of regularization, such as L2 weight decay or varying dropout rates, to improve the model's generalization.
- **Fine-tuning**: Experiment with other architectural changes like adding residual connections.
- **Data Augmentation**: Since the model is overfitting, applying data augmentation techniques might help expose the model to a more diverse set of input data and improve its robustness.