Enhancing Brain Tumor Classification: VGG16 with Low-Rank Adaptation (LoRA)

Justin Zeng
Columbia University Data Science Institute
jz3886@columbia.edu

December 16, 2024

Abstract

This paper investigates the application of Low-Rank Adaptation (LoRA) to the VGG16 convolutional neural network for brain tumor classification. By integrating LoRA, I aim to improve model performance and parameter efficiency on medical imaging datasets. Hyperparameter tuning experiments highlight the challenges and initial results of this approach. Further optimization and exploration of parameter-efficient fine-tuning techniques are required to maximize the potential of LoRA in medical image analysis.

1 Introduction

Brain tumor classification is a critical task in medical image analysis with significant implications for patient diagnosis and treatment. Accurate classification enables early intervention and optimized therapeutic strategies, ultimately improving patient outcomes. The challenge lies in the variability of tumor appearances in MRI scans, requiring advanced computational techniques to achieve reliable classification.

Gliomas, meningiomas, and pituitary tumors are distinct types of brain tumors that significantly impact patients' lives. Gliomas, which include the aggressive glioblastoma multiforme, are among the most common and deadliest brain tumors, with an estimated 20,000 cases diagnosed annually in the United States alone (American Cancer Society, 2021). These tumors often lead to severe neurological impairment, and despite treatment, prognosis is generally poor. Meningiomas, typically benign but potentially life-threatening depending on their size and location, account for approximately 36% of all primary brain tumors, affecting over 100,000 patients each year in the U.S. (American Brain Tumor Association, 2021). Pituitary tumors, though often benign, can cause significant hormonal imbalances that disrupt normal bodily functions, impacting around 15,000-20,000 patients annually in the U.S. (Endocrine Society, 2020). These tumors require precise treatment strategies to prevent long-term complications and enhance patients' quality of life. Early and accurate detection of these tumors is critical for improving patient outcomes, emphasizing the need for advanced medical imaging and classification techniques.

Convolutional Neural Networks (CNNs), particularly the VGG16 architecture, have shown promising results in medical image classification. However, traditional fine-

tuning approaches often suffer from computational inefficiency and potential overfitting, especially when applied to large-scale datasets. These limitations are particularly evident in medical imaging, where datasets may be imbalanced or scarce.

Low-Rank Adaptation (LoRA) emerges as an innovative technique to address these challenges. By introducing low-rank matrix decompositions during fine-tuning, LoRA reduces the number of trainable parameters while maintaining model performance. This paper explores the integration of LoRA with VGG16 for brain tumor classification, aiming to improve parameter efficiency and overall classification accuracy.

1.1 Background

Deep learning models have become pivotal in medical imaging, but their computational demands pose challenges for efficient deployment. The VGG16 architecture is widely recognized for its ability to capture hierarchical image features, making it suitable for brain tumor classification. LoRA, as a parameter-efficient fine-tuning method, offers a potential solution to enhance VGG16's adaptability without requiring full model retraining. By reducing the computational overhead, LoRA facilitates quicker iterations and scalability to larger datasets or new tasks.

1.2 Objectives

The primary objectives of this study are:

- To implement a VGG16 model with LoRA for brain tumor classification.
- To evaluate the performance gains and parameter efficiency of LoRA.
- To demonstrate the applicability of parameter-efficient fine-tuning techniques in medical image analysis.
- To explore the potential of LoRA in improving classification outcomes with limited training data.

2 Architecture Design

The proposed VGG16-LoRA architecture adapts the original VGG16 design by applying LoRA to the fully connected layers. LoRA is applied to the fully connected layers to reduce their computational complexity. The architecture is detailed below:

Layer	Configuration
Input	Shape: $(224, 224, 3)$
Conv2D	Filters: 64, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 64, Kernel: (3×3), Padding: 1
ReLU	Activation
MaxPool2D	Pool Size: (2×2), Stride: 2
Conv2D	Filters: 128, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 128, Kernel: (3×3), Padding: 1
ReLU	Activation
MaxPool2D	Pool Size: (2×2), Stride: 2
Conv2D	Filters: 256, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 256, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 256, Kernel: (3×3), Padding: 1
ReLU	Activation
MaxPool2D	Pool Size: (2×2), Stride: 2
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
MaxPool2D	Pool Size: (2×2), Stride: 2
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
Conv2D	Filters: 512, Kernel: (3×3), Padding: 1
ReLU	Activation
MaxPool2D	Pool Size: (2×2), Stride: 2
Flatten	-
LoRA Fully Connected	Units: 4096, Rank: 8
ReLU	Activation
LoRA Fully Connected	Units: 4096, Rank: 8
ReLU	Activation
Fully Connected	Units: 2

2.1 Technical Framework

The model is implemented using PyTorch. LoRA is integrated by decomposing the fully connected layers into two low-rank matrices, initialized with small random values. The architecture leverages GPU resources for accelerated training, with hyperparameters fine-tuned to achieve an optimal balance between learning rate, regularization, and convergence.

2.2 Low-Rank Adaptation Details

The low-rank adaptation mechanism involves decomposing the weight matrices of the fully connected layers. Given a weight matrix W of dimensions $d \times k$, LoRA represents W as the product of two matrices U and V, where U has dimensions $d \times r$ and V has dimensions $r \times k$. The rank r is chosen to balance parameter reduction with model expressiveness. During the forward pass, the low-rank components are used to approximate the original transformation, ensuring efficient adaptation.

The methodology for this study involves training the VGG16-LoRA model using a standard set of hyperparameters, with initial baseline training to establish a reference performance. The baseline model was trained for 10 epochs with the following hyperparameters: a learning rate of 0.001, a batch size of 32, and the Adam optimizer.

3 Training and Hyperparameter Tuning

The methodology for this study involves training the VGG16-LoRA model using a standard set of hyperparameters, with initial baseline training to establish a reference performance. The baseline model was trained for 10 epochs with the following hyperparameters: a learning rate of 0.001, a batch size of 32, and the Adam optimizer.

3.1 Dataset

The Kaggle Brain Tumor Classification (MRI) dataset is utilized in this study. This dataset contains MRI scans categorized into different tumor types, with characteristics such as varying image resolutions and tumor distributions. MRI images are preprocessed to standardize dimensions and enhance contrast for improved feature detection.

The dataset used in this study comprises high-dimensional data represented by both training and test dataframes. The training dataframe contains 2,870 rows and 150,529 columns, while the test dataframe consists of 394 rows and 150,529 columns. This high-dimensional representation poses challenges for feature selection and model optimization, necessitating the use of techniques like LoRA to manage computational overhead effectively.

Train Data Class Distribution:

- `glioma_tumor` (encoded as 3): 827 samples (28.82%)
- `meningioma_tumor` (encoded as 0): 826 samples (28.78%)
- `no_tumor` (encoded as 1): 822 samples (28.64%)
- `pituitary_tumor` (encoded as 2): 395 samples (13.76%)

Test Data Class Distribution:

- `no_tumor` (encoded as 1): 115 samples (29.19%)
- `pituitary_tumor` (encoded as 2): 105 samples (26.65%)
- `meningioma_tumor` (encoded as 0): 100 samples (25.38%)
- `glioma_tumor` (encoded as 3): 74 samples (18.78%)

The class distribution in both the training and test datasets shows a slight imbalance, especially in the test data, where the class `glioma_tumor` (encoded as 3) is underrepresented. This imbalance may require careful handling during model evaluation to ensure the model's performance is not biased toward the more frequent classes. Techniques such as weighted loss functions or resampling can be considered in future work to address these issues.

3.2 Preprocessing and Data Augmentation

Preprocessing steps include image normalization and resizing to ensure uniform input dimensions. Data augmentation techniques, such as rotation, flipping, and contrast adjustments, are employed to enhance model generalization. Cross-validation is implemented to evaluate the model's robustness across different subsets of the dataset. Images are resized to 224x224 pixels to align with the VGG16 input requirements.

3.3 Baseline Training Results

During the baseline training, the model achieved the following results:

- Accuracy: 0.25 - Recall: 0.25 - Precision: 0.25 - AUROC: 0.49

The training process for the baseline model was conducted over 10 epochs, and the loss at the end of each epoch is shown below:

Epoch [1/10], Loss: 1.3765

Epoch [2/10], Loss: 1.3538

Epoch [3/10], Loss: 1.3506

Epoch [4/10], Loss: 1.3495

Epoch [5/10], Loss: 1.3476

Epoch [6/10], Loss: 1.3512

Epoch [7/10], Loss: 1.3503

Epoch [8/10], Loss: 1.3503

Epoch [9/10], Loss: 1.3493

Epoch [10/10], Loss: 1.349

The loss steadily decreased during the training, indicating some improvement in the model's ability to fit the data, but the results for accuracy and other evaluation metrics remained relatively low, with a 25% recall, precision, and accuracy.

3.4 Hyperparameter Tuning

Extensive hyperparameter tuning experiments were conducted using grid search method to identify the optimal settings for learning rate, batch size, optimizer, and epochs. Table 1 summarizes the results:

Learning Rate	Batch Size	Optimizer	Epochs	Accuracy (%)
0.001	16	Adam	3	29.19
0.001	**32**	**Adam**	**3**	**29.19**
0.005	64	Adam	3	25.38
0.01	32	Adam	3	25.38
0.001	16	Adam	10	0.25
0.001	32	Adam	10	0.25
0.001	16	AdamW	3	25.38
0.001	16	SGD	3	25.38

Table 1: Hyperparameter Tuning Results

Initial experiments focused on determining the best optimizer at a learning rate of 0.001 and batch size of 16, testing epochs 2 and 3. The results showed that the Adam optimizer with 3 epochs provided the best performance, achieving an accuracy of 29.19%. For comparison, using AdamW or SGD as optimizers resulted in slightly lower accuracy of 25.38% at 2 epochs, and 29.19% at 3 epochs, matching Adam's performance.

The learning rate of 0.001 and batch size 32, when paired with the Adam optimizer and trained for 3 epochs, resulted in the highest accuracy of 29.19%, with precision and recall both at 0.2919. The model with a batch size of 64, learning rate 0.001, and the Adam optimizer performed similarly with 29.19% accuracy.

Notably, higher learning rates of 0.005 and 0.01 resulted in poorer performance. Specifically, a learning rate of 0.005 with a batch size of 64 and Adam optimizer caused the model to have a significantly higher loss, achieving an accuracy of 25.38%. The highest learning rate of 0.01 resulted in similarly poor performance, with accuracy hovering around 25.38% across different batch sizes and optimizers.

4 Experiments and Results

4.1 Best Hyperparameters

The best results were achieved with the following parameters: - Learning Rate: 0.001 - Batch Size: 32 - Optimizer: Adam - Epochs: 3 - Accuracy: 29.19

The overall best configuration demonstrates the potential for balancing performance and training time with the right set of hyperparameters.

4.2 Performance Metrics

The model's performance was evaluated using accuracy, precision, recall, and AUROC. In medical applications, particularly in contexts like disease diagnosis, precision and recall are more critical than accuracy alone. While accuracy gives an overall indication of performance, it doesn't account for the class imbalance or the potential consequences of false positives and false negatives.

In medical contexts, it is often preferable to have more false positives than false negatives. This is because false negatives, where the model fails to identify a positive case, can have serious consequences, such as missed diagnoses or untreated conditions. False positives, although they may lead to unnecessary treatments or tests, generally pose less of a risk than missed diagnoses.

In this model, the best hyperparameter combination achieved an accuracy of 29.19%, with precision and recall both at 0.2919. These values indicate that the model is performing fairly evenly in terms of both precision and recall. The relatively low AUROC score suggests the model's ability to differentiate between classes is modest, indicating that there is still room for improvement in the model's discriminatory power. However, the performance metrics suggest that the model is more conservative with false negatives, aligning with the preference for minimizing missed positive cases in medical applications.

5 Results and Discussion

The VGG16-LoRA model demonstrated competitive performance with reduced computational demands. The model's performance, evaluated on the validation set, showed that the best hyperparameters were a learning rate of 0.001 and a batch size of 32, which led to the highest accuracy of 29.19%. This result highlights the model's potential for effective image classification tasks, achieving a reasonable balance between performance and computational efficiency.

Notably, the model was able to achieve these results with only 3 epochs, demonstrating a significant reduction in training time compared to traditional approaches. The use of the Adam optimizer further contributed to the efficiency, allowing for faster convergence and optimized performance. This reduced runtime makes the VGG16-LoRA model particularly appealing for environments where computational resources are limited or when quick experimentation is required.

While the VGG16-LoRA model performed well on image classification tasks, there are many opportunities for future work. One avenue for improvement is to extend this approach to other neural network architectures, such as ResNet or DenseNet, to assess whether the computational efficiency gains are consistent across different models. Additionally, exploring the applicability of LoRA to other domains, such as natural language processing (NLP) or reinforcement learning, could open up new possibilities for using reduced parameter models in a variety of AI applications.

Overall, the VGG16-LoRA model represents a promising direction for reducing the computational load of deep learning models while maintaining competitive performance, and it lays the groundwork for further research and experimentation in this area.

6 Challenges

While the VGG16-LoRA model provides a promising approach for brain tumor classification, several challenges persist in terms of runtime efficiency and data preprocessing.

6.1 Time Constraint

Training deep learning models, especially in medical image analysis, can be computationally expensive and time-consuming. The high-dimensional nature of MRI data, coupled with the complex architecture of VGG16, results in significant runtime requirements. Even with the introduction of LoRA to reduce the number of trainable parameters, the model still faces challenges related to memory usage and processing power. Training the model on large datasets with high-resolution images demands substantial GPU resources and can lead to extended training times, especially when experimenting with different hyperparameters or running multiple iterations for fine-tuning.

6.2 Preprocessing

Data preprocessing plays a critical role in ensuring that the model receives consistent and high-quality input. In the case of MRI scans, preprocessing typically involves normalization, resizing, and enhancement of image features. However, challenges arise due to the variability in image resolution and the presence of noise in medical images. Standardizing the input dimensions of images (e.g., resizing to 224x224 pixels) can lead to loss of important details and may impact model performance.

Moreover, MRI images often suffer from inhomogeneities, artifacts, and poor contrast, which can affect feature extraction. Preprocessing steps such as contrast enhancement and artifact removal are necessary but can be time-consuming and require domain-specific expertise to avoid introducing bias or overfitting. Data augmentation techniques, such as rotations, flipping, and intensity adjustments, are essential to increase model generalization but also increase the overall preprocessing time and complexity.

Additionally, managing high-dimensional data, particularly in the context of multi-channel MRI scans, further complicates preprocessing efforts. Feature selection techniques and dimensionality reduction methods are often employed to mitigate the impact of high-dimensionality, but these methods introduce additional computational overhead and can complicate the pipeline.

7 Next Steps

1. **Hyperparameter Tuning:** Experiment with different ranks for LoRA and learning rates to identify optimal configurations.

2. **PCA:** Explore the use of PCA or other methods to reduce dimensionality.

3. **Comparative Analysis:** Benchmark VGG16_LoRA against other state-of-the-art models to validate its efficiency and accuracy.

8 Conclusion

The integration of LoRA with VGG16 demonstrates modest initial results, achieving an accuracy of 29.19%. While the results suggest potential for parameter-efficient fine-tuning, further hyperparameter optimization is necessary to enhance model performance. Future work will focus on advanced LoRA configurations, longer training durations, and exploring alternative datasets to fully leverage the potential of LoRA in medical image analysis.

9 Code

My implementation is available in this Github repository.

To train or capture same results with hyper-parameter tuning using the model, refer to the How to Run section of the README.

10 Data

The dataset used in my project can be found here.

Acknowledgements

Acknowledgements are extended to dataset providers, computational resource sponsors, and supporting institutions for their invaluable contributions to this research. Special thanks to Simonyan and Zisserman for their foundational work on the VGG16 architecture, which has significantly advanced the field of image classification. Their contributions have laid the groundwork for the methods explored in this study.

References

1. Sartaj Bhuvaji, et al. (2020). Brain Tumor Classification (MRI) [Data set]. Kaggle.

2. Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556.

3. Hu, E., et al. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv.

4. American Cancer Society. (2021). Glioma and Glioblastoma. Retrieved from `https://www.cancer.org`

5. American Brain Tumor Association. (2021). Meningiomas. Retrieved from `https://www.abta.org`

6. Endocrine Society. (2020). Pituitary Tumors. Retrieved from `https://www.endocrine.org`