**Medical Image Classification**

**Using Deep Learning with PyTorch.**

**Smit Lila**

**CECS 456 – Machine Learning**

**GitHub Repository:** [Your GitHub Link]

**Introduction:**

Medical image classification is an important use of deep learning in healthcare, as it aims to automate diagnostic processes with high accuracy. This project uses the Medical MNIST dataset using transfer learning with a customized ResNet18 model to categorize photos into six distinct medical categories. Advanced techniques including data augmentation, adaptive learning rate scheduling, and model checkpointing are used to ensure effective learning and generalization.

**Dataset and Related Work:**

**Dataset Description:**

The Medical MNIST dataset is well-structured for this challenge, with separate training, validation, and test sets. The key preprocessing processes are:

* To ensure compatibility with deep learning models, all photos are resized to 224×224 pixels.
* Normalizing grayscale images to a consistent range.
* To improve model generalization, add random transformations to the dataset.

**Related Work:**

Recent research underscores the role of deep learning in medical imaging:

1. **Transfer Learning:** Pretrained architectures like ResNet have demonstrated superior performance in medical image analysis (He et al., 2016).
2. **Data Augmentation:** Techniques such as rotation, flipping, and scaling improve model generalization for small datasets (Zhang et al., 2020).
3. **ResNet Architectures**: Their skip connections effectively address vanishing gradient problems in deep networks, making them ideal for medical classification tasks (Wang et al., 2019).

**Methodology:**

**Data Processing Pipeline:**

The data processing pipeline starts with scaling images to 224×224 pixels and normalizing grayscale values to [-1,1]. Data augmentation techniques, such as random rotations (up to 30°), horizontal flips, affine transformations, and scale variations (0.8-1.2), are used to increase dataset diversity and improve model generalization. The dataset is organized using a custom PyTorch Dataset class and effectively loaded with the DataLoader module, resulting in smooth training with batch processing.

**Model Architecture**

The suggested solution employs a modified ResNet18 architecture specifically designed for medical image classification:

1. To process single-channel grayscale images, the first convolutional layer is modified to accept a 1×64 filter.
2. The feature extraction backbone uses a pre-trained ResNet18 model with weights from ImageNet for initial feature extraction.
3. The classification head consists of 512 neurons in fully connected layers, 0.5 dropouts for regularization, and a final layer that generates six outputs based on the dataset classes. This design strikes a compromise between feature extraction and task-specific classification.

**Training Strategy:**

The training pipeline employs recent deep learning techniques such as adaptive learning rate scheduling, early halting, and model checkpointing. When the learning rate scheduler detects stagnation in validation loss, it cuts the learning rate to improve convergence. Early stopping guarantees that training stops when no significant improvement is detected, preventing overfitting. Checkpointing saves the best model depending on validation accuracy to ensure optimal performance.

**Experimental Setup**

**Implementation Details:**

The model was built using PyTorch, a popular deep-learning tool. The training procedure used a batch size of 32, a learning rate of 0.001, and the Adam optimizer. The loss function utilized was Cross-Entropy Loss, which is appropriate for multiclass classification issues. The training was carried out across ten epochs, with the device set to GPU for faster computing. In settings lacking GPU access, the code easily switches to CPU execution.

**Training Process**

The training process involved multiple steps:

1. Data pretreatment and augmentation improved the model's capacity to generalize across different input photos.
2. To evaluate generalization and direct learning rate modifications, validation metrics were tracked following each epoch of the model's training on the training set.
3. To assess the model's ultimate performance, testing was carried out on an independent test set.

**Measurement:**

**Performance Metrics**

The following indicators were monitored in order to assess the model:

* **Accuracy**: The percentage of correctly categorized photos during the testing, validation, and training stages was measured.
* **Precision and Recall**: The model's precision and recall were evaluated to determine how well it prevented false positives and false negatives, respectively.
* **F1-Score:** Especially helpful for unbalanced datasets, it offered a balanced measure of precision and recall.
* **Confusion Matrix:** Showcased the model's predictions, pointing out misclassifications and emphasizing accuracy by class.

**Method of Evaluation:**

To examine convergence, training and validation loss were plotted. Class-wise precision, recall, and F1-scores were also used to benchmark the model's performance, offering detailed information on the classification efficacy for every medical category.

**Results Analysis, Intuitions, and Comparison:**

**Performance:**

The following outcomes were attained by the model:

Accuracy of Training: 93.673%

The accuracy of validation: 94.389%

**Confusion Matrix:**

Accurate predictions for classes with high representation, such as chest X-rays and abdominal CT scans, are highlighted in the confusion matrix. Nevertheless, some misclassifications between breast ultrasound and head CT images happened because of overlapping features, highlighting the need for better feature extraction for these categories.

A graph with blue squares and numbers

Description automatically generated

**Insights**:  
1. **Data Augmentation**: Significantly improved the model’s generalization ability, reducing overfitting despite the dataset’s limited size.

2. **Transfer Learning:** By utilizing pre-trained weights, less intensive training was required, enabling the model to converge rapidly with little processing power.

3. **Batch Normalization**: Enhanced convergence and stability during training, especially in the ResNet architecture's deeper layers.

**Comparison Analysis:**  
The improved ResNet18 achieved faster convergence and higher accuracy than baseline models trained without transfer learning. The model's performance was further improved via hyperparameter tuning and augmentation techniques, which showed a successful trade-off between accuracy and training complexity.

**Conclusion:**

Using PyTorch and the Medical MNIST dataset, this research effectively illustrated a deep-learning approach for medical image categorization. To attain excellent classification accuracy across six medical categories, the implementation made use of transfer learning, data augmentation, and sophisticated training techniques. The findings highlight deep learning's potential for automating medical diagnosis and optimizing healthcare processes. To improve clinical acceptance, future directions include investigating deeper structures, utilizing domain-specific augmentation strategies, and incorporating explainability frameworks.

**Contributions:**

This project was finished by me alone. Important contributions consist of:

1. Putting the process for dataset preparation and augmentation into action.
2. Creating and adjusting the ResNet18 architecture for the classification of grayscale images.
3. Incorporating cutting-edge methods like checkpointing and learning rate scheduling into the training process.
4. Evaluating outcomes using comparative research, confusion matrix visualizations, and performance measures.
5. Dividing the source up into modular parts and offering thorough documentation to ensure reproducibility.