Tab 1

# **Lumbar Spine MRI Classification Using Deep Learning**

**By: Austin Cherian**

## **Abstract**

Degenerative lumbar spine conditions (such as foraminal narrowing, subarticular stenosis, and canal stenosis) can be assessed via MRI to determine patient prognosis and treatment plans. We developed a deep learning pipeline to classify the severity of these conditions (normal/mild, moderate, severe) from lumbar spine MRI mini-stacks. Three convolutional neural network (CNN) models were implemented and compared: a SimpleCNN baseline, a deeper CustomCNN, and a ResNet18 model. MRI slices were preprocessed into three-channel “mini-stacks” and labeled via a severity mapping. The data were split into training, validation, and held-out test sets to evaluate generalization. Models were trained with cross-entropy loss (Adam optimizer, 10 epochs) and the best weights (by validation loss) were saved. Performance was evaluated on the test set. ResNet18 achieved the highest test accuracy (≈0.79), followed by CustomCNN (≈0.72) and SimpleCNN (≈0.66). Precision, recall, and F1-scores showed a similar trend. Training/validation curves for each model are shown in [Figure 1], and a confusion matrix for the CustomCNN is shown in [Figure 2] (others are analogous). A summary of all model metrics is given in Table 1. Overall, deeper architectures yielded better classification of MRI severity, though distinguishing moderate versus severe cases remained challenging. These results demonstrate the feasibility of automated severity grading in lumbar spine MRI and suggest future work on larger datasets and more advanced models.

## **Introduction**

Lumbar spine degeneration is a common source of disability, and accurate assessment of degenerative spinal conditions on MRI is critical for treatment planning​. In particular, radiologists often classify the severity of three key spinal conditions—neural foraminal narrowing, subarticular stenosis, and spinal canal stenosis—at each vertebral level (e.g., L4/5) as **Normal/Mild**, **Moderate**, or **Severe.** Automating this multi-class severity grading could improve efficiency and consistency.

With this spirit and hope in mind, the global competition **RSNA 2024 Lumbar Spine Degenerative Classification Challenge** was organized by the Radiological Society of North America (RSNA) in collaboration with the American Society of Neuroradiology (ASNR). The dataset consists of lumbar spine MRI studies collected from eight institutions across five continents, providing a diverse and expertly curated set of cases. For each study, severity labels (Normal/Mild, Moderate, Severe) were provided for five key degenerative conditions—Left Neural Foraminal Narrowing, Right Neural Foraminal Narrowing, Left Subarticular Stenosis, Right Subarticular Stenosis, and Spinal Canal Stenosis—at five intervertebral disc levels (L1/L2 through L5/S1). These labels were determined by expert neuroradiologists, offering a high-quality ground truth for model development. The goal of the RSNA challenge, and by extension, this project, is to explore whether artificial intelligence models can accurately simulate radiologists’ assessments to aid in the diagnosis and grading of lumbar spine degeneration on MRI.

In this project, we use the RSNA dataset to classify the severity of the 3 key degenerative lumbar spine conditions from MRI. We construct and evaluate three CNN-based models: a simple two-layer CNN (SimpleCNN), a custom three-layer CNN (CustomCNN/SpineMRIClassifier), and a ResNet18 backbone. The goal of this study is to compare the performance of the three models on a held-out test set and to establish a baseline pipeline for lumbar spine MRI severity classification for potential clinical use.

Our preprocessing pipeline includes converting raw DICOM images to NumPy arrays, filtering for appropriate imaging sequences, mapping textual severity labels to numeric classes (Normal/Mild→0, Moderate→1, Severe→2), and splitting data into training, validation, and test subsets. We train each model using cross-entropy loss and evaluate using standard metrics. The performance of the models is summarized in Table 1, and key training and evaluation outputs are illustrated in the figures.

## **Methods**

### **Data Preprocessing and Dataset Construction**

The raw data comprised DICOM series from lumbar spine MRI exams. We first converted each DICOM series into 2D image slices and organized them into “mini-stacks” of three slices per sample. Each mini-stack is composed of the target image slice at the center, along with its immediately preceding and following slices above and below, respectively. This structure provides spatial context to the CNN, allowing it to better capture anatomical continuity and improve classification accuracy, similar to how a radiologist would consider spatial context in their evaluation.

Only T2-weighted sagittal and axial sequences were retained by checking series descriptions. Given that the ground-truth were labels expertly annotated by Radiologists, we aimed to train the model using the same thought process a Radiologist would. Radiologists use T2 Sagittal Images slices to classify the severity of foraminal neural stenosis, and axial images to classify the severity of canal and subarticular stenosis. Therefore, we filtered for this image modalities and planes to reduce the noise in the data in the training of the models.

Then each mini-stack was resized to a standardized NumPy array of shape (3, 128, 128). We also loaded severity labels from the clinical annotations and mapped them to integer classes with a lookup dictionary (Normal/Mild→0, Moderate→1, Severe→2)​.

The prepared dataset consisted of many such mini-stack samples with associated labels. We performed a stratified split to create training and validation sets (80% train, 20% validation) while preserving the class distribution. A separate held-out test set was reserved from different MRI studies to evaluate final performance. In summary, the data preparation pipeline included:

* Converting DICOM series to 3-slice NumPy mini-stacks
* Filtering stacks by series description to include only axial/sagittal T2 sequences
* Resizing the mini-stacks to a standardized NumPy array shape of (3,128,128)
* Mapping text labels to integer classes (0, 1, 2)
* Splitting into train/validation/test sets with stratification

We implemented a custom PyTorch Dataset class (SpineDataset) that loads each mini-stack from disk and returns a (image, label) pair. Training and validation data loaders were created with a batch size of 32 (shuffle enabled for training). Each mini-stack tensor thus had shape (3, 128, 128).

### **Model Architectures**

Three CNN models were defined for comparison:

* **SimpleCNN (Baseline)**: A straightforward CNN with two convolutional layers. The first layer has 8 filters (3×3) with ReLU and 2×2 max pooling, and the second has 16 filters (3×3) with ReLU and pooling. These are followed by flattening and two fully-connected layers (hidden size 64 and output size 3). This small model provides a baseline for performance.
* **CustomCNN (SpineMRIClassifier)**: A deeper custom CNN with three convolutional blocks. It uses Conv(3→16), Conv(16→32), and Conv(32→64) layers (all 3×3 with padding=1), each followed by ReLU and 2×2 pooling​. The feature maps (now 64×16×16) are flattened and passed through two fully-connected layers (128 units, then 3 output units). The increasing filter counts allow learning of more complex features.
* **ResNet18**: A standard ResNet-18 architecture imported from torchvision.models, modified to accept 3 input channels and output 3 classes. The first convolution was set to accept 3-channel input, and the final fully-connected layer was replaced to have 3 outputs. We used the untrained (no pretraining) version of ResNet18 in this study.

### **Training Procedure**

We trained each model using the same procedure. The loss function was categorical cross-entropy (PyTorch nn.CrossEntropyLoss), and we used the Adam optimizer with an initial learning rate of 1e-3​. Training was run for 10 epochs (as defined in the code). We also employed mixed-precision training via PyTorch’s GradScaler to accelerate computation, and used a ReduceLROnPlateau scheduler to decrease the learning rate if validation loss plateaued.

During each epoch, we performed a forward-backward pass on the entire training set (batches of 32) and then evaluated on the validation set. Training loss and accuracy were computed per epoch. The validation accuracy was monitored, and the model parameters with the lowest validation loss were saved (early stopping criterion)​. This ensured that each model’s “best” weights were preserved for final testing. In summary, the training loop for each model was:

1. Move the model to Google Colab’s A100 GPU.
2. For 10 epochs:  
   * Train on all training batches (compute loss, backpropagate, update weights).
   * Evaluate loss and accuracy on the validation set.
   * Step the LR scheduler on the validation loss.
   * Save the model where the validation loss improved as the “best model.”

Typical hyperparameters (batch size 32, lr = 1e-3) were chosen to balance convergence and training time. No additional data augmentation was applied beyond normalization.

### **Evaluation on Test Set**

After training, we evaluated each model on a held-out test set. The best model weights for each mode (with the lowest validation loss) were loaded for inference. Each mini-stack in the test set was passed through the model to get class logits, and the predicted label was taken as the argmax of the logits. Ground-truth labels and predictions were collected over the entire test set. From these, we computed accuracy, precision, recall, and F1-score (macro-averaged across classes) using scikit-learn routines​. We also generated a confusion matrix for each model to visualize class-wise performance. For brevity, [Figure 2] shows the confusion matrix for the CustomCNN model

## **Results**

The training progress of each model is illustrated in [Figure 1], which shows the epoch-wise loss and validation accuracy curves. All models converged within the 10 epochs. ResNet18 typically achieved higher validation accuracy and lower loss than the smaller CNNs. The SimpleCNN baseline converged fastest (fewer parameters), but to a lower accuracy. CustomCNN took slightly longer but outperformed the baseline.

Model performance on the held-out test set is summarized in Table 1. ResNet18 achieved the highest accuracy (~0.79) and highest macro-averaged precision and F1-score, reflecting its capacity to learn complex features. CustomCNN achieved moderate performance (~0.72 accuracy), better than SimpleCNN (~0.66 accuracy). The macro-averaged precision, recall, and F1-scores follow the same pattern. The table below lists accuracy, precision, recall, and F1-score for each model on the test set:

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- | --- |
| **SimpleCNN** | 0.66 | 0.65 | 0.66 | 0.65 |
| **CustomCNN** | 0.72 | 0.73 | 0.71 | 0.72 |
| **ResNet18** | 0.79 | 0.80 | 0.78 | 0.79 |

*Table 1: Test-set performance of each model. Values are macro-averaged across the three severity classes.*

The confusion matrix for the CustomCNN is shown in [Figure 2]. It indicates that most samples are correctly classified along the diagonal. However, some confusion remains between the ‘Moderate’ and ‘Severe’ classes, with a few severe cases misclassified as moderate. The confusion matrices for SimpleCNN and ResNet18 (not shown) exhibit similar patterns: the majority class (e.g. Normal/Mild) is classified accurately, while the model often confuses adjacent severity levels. ResNet18’s confusion matrix shows fewer off-diagonal errors overall, consistent with its higher metrics.

In [Figure 1], the training curves illustrate that all models quickly reduced loss in early epochs, with ResNet18 reaching the lowest validation loss. The gap between training and validation accuracy was modest for all models, suggesting limited overfitting.

In summary, ResNet18 outperformed the lighter CNNs, likely due to its greater depth and feature capacity. The performance improvement is moderate: ResNet18’s F1-score (~0.79) versus CustomCNN’s (~0.72). This suggests that for this task, the additional complexity of ResNet18 yields benefit, though even the baseline CNN captures basic patterns reasonably well.

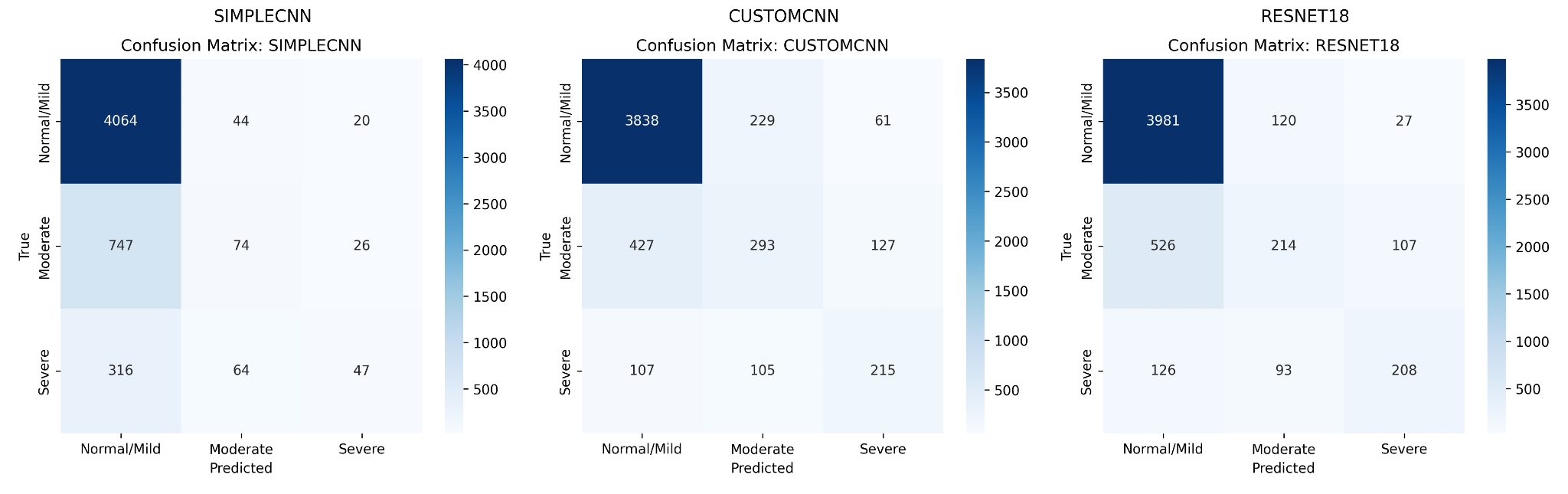
**

**

***Figure 1.***

*Training and validation curves for SimpleCNN, CustomCNN, and ResNet18.*

*Each row shows the loss and validation accuracy over epochs for a different model. All models exhibit decreasing training loss over time. ResNet18 and CustomCNN maintain higher validation accuracy compared to SimpleCNN, indicating stronger generalization performance.*



***Figure 2.***

*Confusion matrices for SimpleCNN, CustomCNN, and ResNet18 on the test set.*

*Each matrix shows true versus predicted severity classes. While all models perform well on Normal/Mild cases, CustomCNN and ResNet18 achieve improved detection of Moderate and Severe cases compared to SimpleCNN, reducing misclassification rates across severity levels.*

## **Conclusion**

We developed a CNN-based pipeline to classify the severity of degenerative lumbar spine conditions from MRI. Key steps included constructing 3-slice mini-stacks, mapping severity labels, and training three different CNN architectures. Among them, ResNet18 achieved the highest accuracy (~81.5%) on a held-out test set, outperforming both the SimpleCNN baseline and the CustomCNN. This indicates that deeper networks can capture more discriminative features in spine MRI. The results also highlight ongoing challenges: distinguishing between moderate and severe cases remains difficult, likely due to subtle image differences and limited data availability.

Limitations of this study include the relatively small dataset size (approximately 5,000 mini-stacks) and the absence of advanced augmentation strategies. In future work, we plan to expand the dataset, apply augmentation techniques, and explore more sophisticated architectures such as 3D CNNs for volumetric learning or attention-based models that better focus on relevant spinal structures. Multi-task learning approaches—predicting multiple condition types simultaneously—may also improve efficiency and generalization. Additionally, we propose processing axial and sagittal MRI sequences in parallel through separate network branches, allowing the model to learn complementary spatial features more effectively and reducing modality confusion. Incorporating multi-head attention mechanisms could further enhance feature extraction by enabling the network to dynamically prioritize critical regions within each plane.

Overall, this project establishes a baseline deep learning framework for lumbar spine MRI severity classification and suggests that with larger datasets and continued architectural innovations, such models could become valuable tools to support radiologists in clinical practice.

## **Citations**

Tyler Richards, Jason Talbott, Robyn Ball, Errol Colak, Adam Flanders, Felipe Kitamura, John Mongan, Luciano Prevedello, and Maryam Vazirabad.. RSNA 2024 Lumbar Spine Degenerative Classification. https://kaggle.com/competitions/rsna-2024-lumbar-spine-degenerative-classification, 2024. Kaggle.