Assessment of Selective Kernels in a Convolutional Neural Network as a Supervised Learning Approach to Detection of Alzheimer's Disease

Dillon Finkenbinder UTP2ZJ@VIRGINIA.EDU
Lindsay Truong LET7EC@VIRGINIA.EDU
Saad Bux RDR5UV@VIRGINIA.EDU

Abstract

Convolutional Neural Networks (CNNs) typically portray neurons with receptive fields of fixed size in a given layer. Existing research by (Li et al., 2019) conceptualizes receptive fields that adapt their sizes to diverse inputs within an artificial neuron to mimic focused visual attention in a framework called selective kernel networks (SKNets). Their research found that classification performance of SKNets compared to other state-of-the-art models like ResNet and DPN was markedly better across a range of parameters. We utilize a set of magnetic resonance images within an SKNet model to observe its ability to correctly detect and classify Alzheimer's disease. Our discussion includes an analysis of predictive performance and attention-related metrics compared to a standard CNN model. Our findings validate the utility of SKNets in Alzheimer's disease detection and classification, highlighting their distinctive properties compared to traditional CNN models. The code is publicly available at https://github.com/dfinkenbinder/MLIA-SKNet-CNN.

Keywords: Classification, CNNs, Deep Learning, Healthcare, Neural Networks, SKNets, Supervised Learning.

1. Introduction

Convolutional Neural Network (CNN) architectures are the current fundamental technique for modern visual recognition systems. Inspired by the hierarchical pattern recognition of the human visual cortex, CNNs utilize layers of convolutional filters to extract progressively higher-level features from raw images. However, traditional CNNs have often relied on fixed-size receptive fields throughout their convolutional layers. This design lacks the dynamic scaling of receptive fields exhibited by biological neurons. This static nature limits the network's ability to process visual information effectively across various scales.

To bridge this gap, architectures such as InceptionNets introduced multi-scale feature aggregation, yet their static kernel series fails to offer the adaptability seen in natural visual systems. Selective Kernel Networks (SKNets) propose a novel solution with their "Selective Kernel" (SK) convolutional operation. As illustrated in Figure 1 (Li et al., 2019),

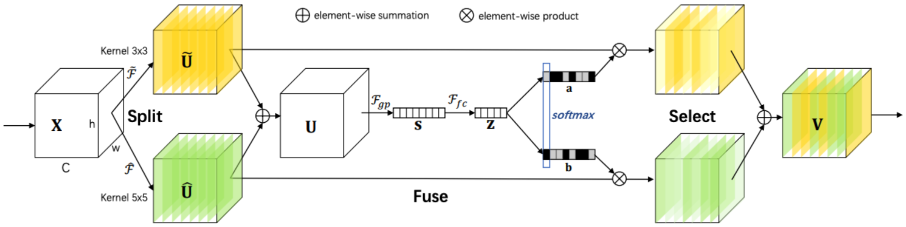


Figure 1: Selective Kernel Convolution (Two Branches)

SKNets incorporate a soft attention mechanism that adaptively enables the network to adjust receptive field sizes. This mechanism involves a Split operator that divides the input into paths with different kernel sizes, effectively creating a diverse set of receptive fields. The Fuse operator then synthesizes the information from these paths, and the Select operator employs a selection mechanism, guided by the fused information, to dynamically choose the optimal receptive field size for each neuron, thereby addressing the input's complexity.

Despite the advantages, SKNets have drawbacks. The process of selecting the most suitable kernel for a given input can introduce additional computational overhead. Figure 2 (Li et al., 2019) underscores the parameter efficiency of SKNets, yet this efficiency must be weighed against the increased computational demands of the selection process. Moreover, effectively training SKNets to leverage their dynamic selection capability requires careful consideration to prevent overfitting and to maintain generalization across diverse visual tasks.

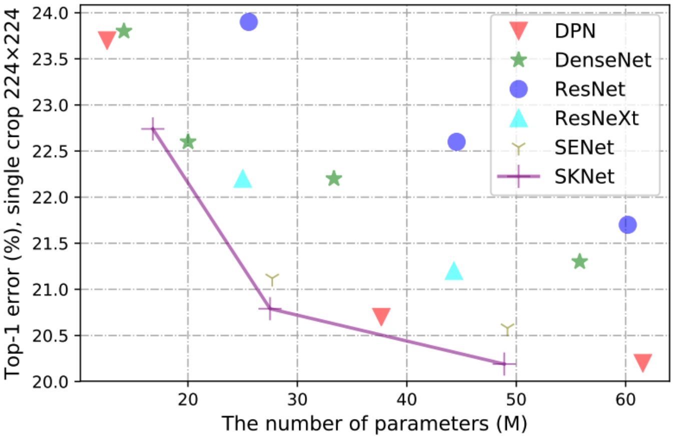


Figure 2: Relationship between the performance of SKNet and the number of parameters in it, compared with the state-of-the art

The adaptability of SKNets represents a significant stride toward replicating the unique processing capabilities of the human visual system. By allowing for dynamic receptive field sizes, SKNets advance the state-of-the-art in image recognition, offering a system that can more accurately interpret and analyze complex visual data. However, the computational intricacies of SKNets highlight an area of ongoing research seeking to optimize the balance between adaptability, efficiency, and performance in CNN architectures.

2. Methodology

This study utilizes the SK convolution, a mechanism that dynamically adjusts receptive field sizes in CNNs. SK convolution comprises three primary operations: Split, Fuse, and Select (mathematical derivations provided in Appendix A).

- **Split Operation**: in the Split operation, each input feature map undergoes two transformations using distinct kernel sizes, 3 and 5. This process includes efficient grouped/depthwise convolutions, Batch Normalization, and ReLU activation functions. A dilated convolution, employing a 3x3 kernel with a dilation size of 2, is utilized in place of the conventional 5x5 kernel for enhanced efficiency. The Split operation essentially deconstructs the information into different scales, extracting diverse features from the input data

- **Fuse Operation**: the Fuse operation integrates the outputs of the Split phase. Here, gates control the flow of multi-scale information into neurons in the subsequent layer. Outputs from various branches are combined through element-wise summation. Global average pooling is then applied to generate channel-wise statistics, which are further processed through a fully connected (fc) layer to produce a compact feature representation. This representation serves as a guide for adaptive kernel selection.

- **Select Operation**: the Select operation employs a soft attention mechanism across channels, allowing the model to focus on the most critical features. It uses softmax operators on channel-wise features, guided by the compact descriptor from the Fuse phase. The final feature map, representing selected spatial scales of information, is obtained by applying attention weights to the kernels.

The architecture of SKNet is based on the ResNeXt framework, with modifications to incorporate SK convolutions. SKNet is characterized by repeated bottleneck blocks, termed “SK units,” each consisting of a sequence of 1×1 convolution, SK convolution, and another 1×1 convolution. The integration of SK convolutions in place of large kernel convolutions enables the network to adaptively select receptive field sizes, enhancing its ability to process complex image data.

3. Experiment and Results

3.1. Results of Existing Research

a) **ImageNet Classification**: The ImageNet 2012 dataset, featuring 1.28 million training images and 50,000 validation images across 1,000 classes, was utilized to evaluate the performance of SKNets. Two specific models, SKNet-50 and SKNet-101, were benchmarked against other state-of-the-art architectures. The results showed that SKNets consistently outperformed comparable models in terms of accuracy and computational efficiency. This superior performance is attributed to their dynamic receptive field adjustment capability, which is particularly advantageous in handling diverse and complex image data found in ImageNet.

b) **CIFAR Classification**: Further, SKNets were tested on CIFAR-10 and CIFAR-100 datasets to assess their performance on smaller-scale image datasets. These datasets present a different set of challenges, with CIFAR-10 consisting of 60,000 images in 10 classes and CIFAR-100 comprising 60,000 images in 100 classes (Krizhevsky, 2009). Despite the reduced size and increased complexity of CIFAR-100, SKNets demonstrated better or comparable performance with fewer parameters compared to models like ResNeXt and SENet. This underscores the adaptability and efficiency of SKNets across various image classification tasks.

c) **Ablation Studies**: In-depth ablation studies were conducted to understand the impact of different configurations within the SKNet-50 model. These studies focused on variations in kernel size, dilation, and group number within the SK convolutional layers. The findings from these studies revealed that SKNets efficiently utilize multiple kernels and adaptive selection mechanisms, leading to improved performance over a range of image recognition tasks. Such studies are crucial in elucidating the architectural features that contribute most significantly to the effectiveness of SKNets.

3.2. Application of SKNets to Detection of Alzheimer's Disease

Note: instructions for running the code in this section is provided in Appendix B.

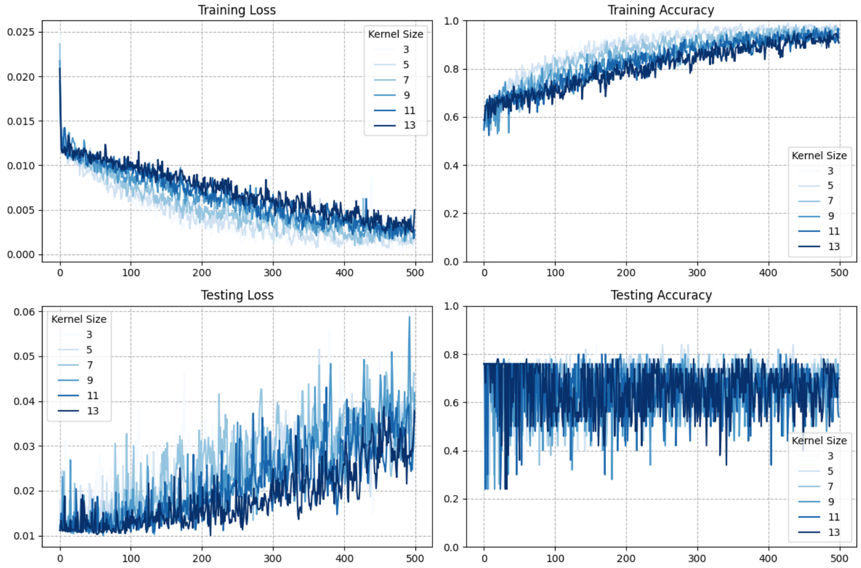


Figure 3: Training and Testing Dynamics of SKNets across Different Kernel Sizes

Figure 3 presents the evolution of training loss, testing loss, training accuracy, and testing accuracy over 500 epochs for multiple kernel sizes used within the SKNet architecture. Each subplot represents a different aspect of model performance, showcasing how different kernel sizes influence the learning process and generalization capabilities of the network.

The top left subplot visualizes the training loss across kernel sizes from 3 to 13. We observe a downward trend in loss values, indicating effective learning. The top right subplot shows the training accuracy, which increases over time, plateauing as the models converge. This pattern highlights the capability of SKNets to learn efficiently from the training data, despite the variance in receptive field sizes.

The bottom subplots present the test loss and accuracy, showcasing the generalization performance of the models. Despite some fluctuation, likely due to the complexity and variability of the validation data, the test metrics exhibit an overall increase in accuracy and a decrease in loss, demonstrating the robustness of SKNets. Notably, certain kernel sizes achieve better peak accuracies, suggesting a potential match between the receptive field size and the scale of features important for classification tasks.

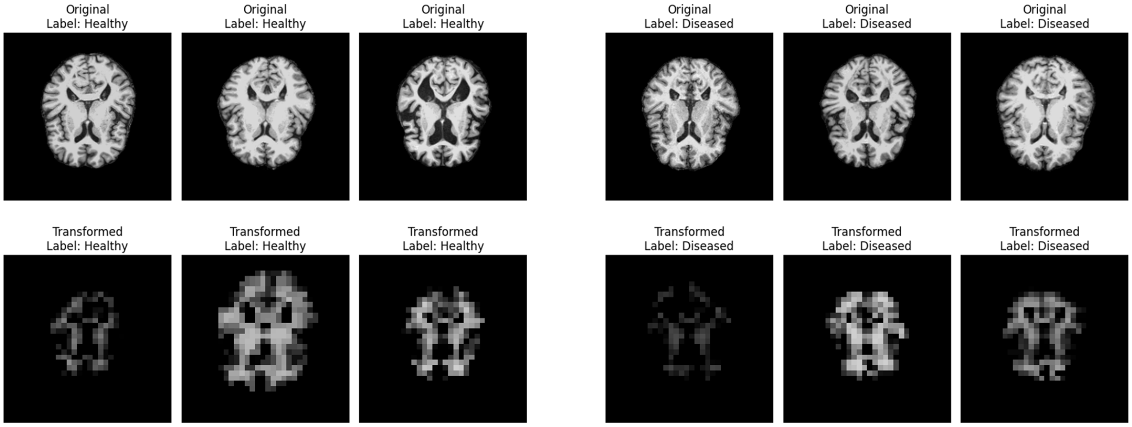


Figure 4: Sample Image Transformations

Figure 4 provides a visual overview of our study's original and transformed images, underscoring the preprocessing steps and augmentation strategies employed to prepare the data for model training and testing.

The training and test accuracy distribution across different kernel sizes in Figure 5 display the SKNet model's adaptability. Each kernel size's performance across the training and testing phases emphasizes the model's capability to generalize from training data to unseen images.

Figure 6 displays the best-case iterative model accuracy over 500 epochs with a kernel size of 3, showing a clear trend of improvement and convergence in the training phase and stable performance during testing. This iterative performance graph is crucial for understanding the learning trajectory and the epochs at which the model reaches optimal accuracy.

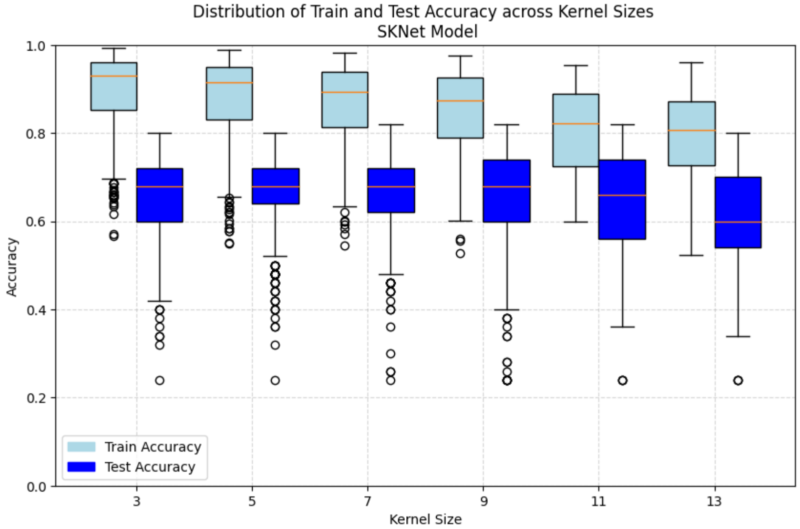


Figure 5: SKNet Performance Metrics

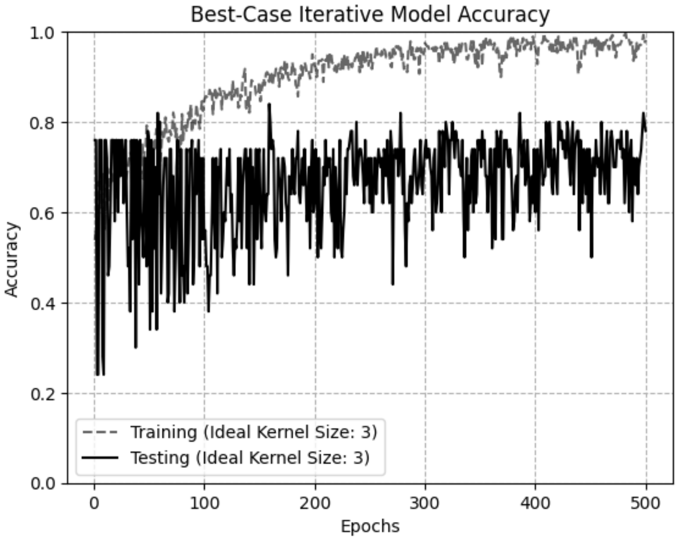


Figure 6: Iterative Performance – Training vs. Testing

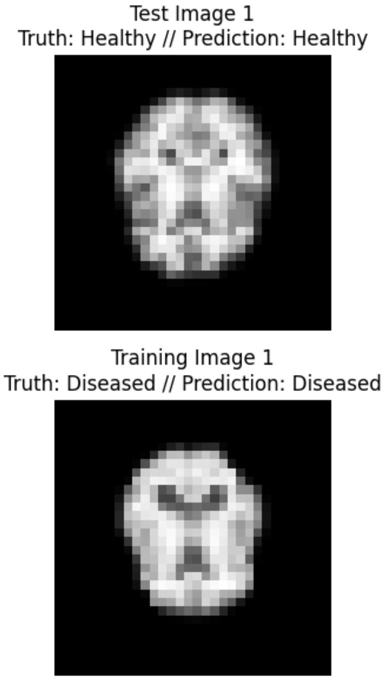

Test Image 1
Truth: Healthy // Prediction: Healthy
Test Image 2
Truth: Healthy // Prediction: Healthy
Training Image 1
Truth: Diseased // Prediction: Diseased
Training Image 2
Truth: Healthy // Prediction: Healthy

Figure 7: Sample Model Predictions with Ideal Kernel Size

Finally, Figure 7 demonstrates the SKNet's predictive accuracy with the ideal kernel size of 3, providing sample predictions that reinforce the model's proficiency in distinguishing between healthy and diseased cases.

The results from these ablation studies indicate that SKNets can efficiently use multiple kernels, leading to improved performance. This is particularly evident in the dynamic receptive field adjustment capability, which allows the network to focus on the most pertinent features within the images, a skill especially useful for complex datasets like ImageNet and CIFAR.

These ablation studies and performance metrics show the adaptability and efficiency of SKNets, solidifying their place as a viable choice for image classification tasks, particularly in the nuanced field of medical image analysis for Alzheimer's disease detection.

4. Discussion

The SKNet exhibited adequate classification performance for both diseased and healthy images across various kernel sizes. However, a notable observation is the testing accuracy plateau below 0.8. This may be indicative of potential biases introduced during the training phase, where the model could have disproportionately focused on features that were less prevalent or impactful in the testing set. Future research should prioritize the development of analytics to identify and assess the influence of features that draw the most attention during training and evaluate their relevance in the testing set.

A contributing factor to the model's performance could be attributed to the limited size of our dataset. The original study (Li et al., 2019) utilized an extensive dataset of 1.28 million images, which provided exposure to diverse features, thereby enhancing the utility of selective kernels within their neural network. Expanding the dataset size is a priority for future work, allowing us to determine the extent to which a larger and more varied training set can enhance classification accuracy.

Our hypothesis that SKNet would surpass a conventional CNN in classification capability was tested by constructing a CNN with fixed receptive fields, replacing the SK units of the SKNet model. Despite the expectation, both the SKNet and the CNN models demonstrated comparable performance in testing sequences, with the CNN model exhibiting larger interquartile ranges, suggesting increased uncertainty in the classification accuracy (Figure 8). This parity in performance challenges the assumption that the adaptive capabilities of SK units directly translate to superior accuracy. Further investigation is required to unravel the nuanced differences between the two models and understand the marginal advantage observed with SKNet in training sequences.

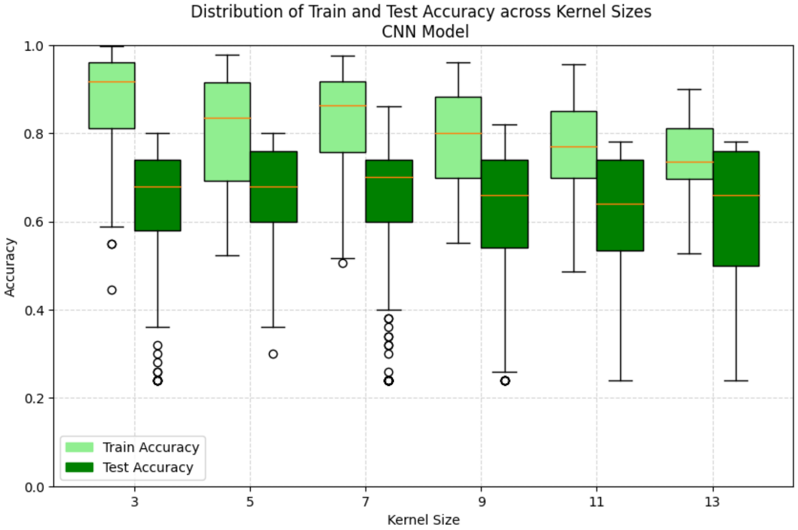


Figure 8: CNN Performance Metrics

5. Conclusion

In summary, the investigation into SKNets suggests their potential tool in the automated detection and classification of Alzheimer's disease through medical imaging. However, the influence of possible biases and the restricted size of the training dataset on the model's performance cannot be ignored. Addressing these potential factors will be critical for future research, which should include (1) the development of metrics to identify and assess the impact of key features learned during training on the model's performance during testing, and (2) the expansion of the training dataset to determine if a larger and more varied set of images can enhance the model's diagnostic accuracy.

In our experimental study, SKNet's performance was closely matched by that of the conventional CNN model, contrary to our initial assumptions that selective kernel utilization would lead to significant performance gains. This observation suggests that the advantages of selective kernels may not be as pronounced in certain contexts and warrants a deeper investigation into the factors affecting model performance in medical image classification such as similarities between the testing accuracies between the two model frameworks.

References

Vikas Bhadoria. Sknet. https://kaggle.com/code/heyi2020/cifar10-sknet/notebook, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009.

Yonghye Kwon. Sknet-pytorch. https://github.com/developer0hye/SKNet-PyTorch, 2019.

Zhiqiang Lang. Sknet. https://github.com/pppLang/SKNet, 2019.

Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–519, 2019. doi: 10.1109/CVPR.2019.00060.

Appendix A. Split, Fuse, and Select Operations

Note: the derivations assume operations with two branches as depicted in Figure 1.

Split: for a given feature map $\mathbf{X} \in \mathbb{R}^{H' \times W' \times C'}$, the first operation involves two transformations:

$$\tilde{\mathcal{F}} : \mathbf{X} \to \tilde{\mathbf{U}} \in \mathbb{R}^{H \times W \times C},\ \hat{\mathcal{F}} : \mathbf{X} \to \hat{\mathbf{U}} \in \mathbb{R}^{H \times W \times C} \tag{1}$$

with kernel sizes of 3 and 5 respectively.

Fuse: this operation uses 'gates' to control information flow from branches carrying various scales of information to neurons in subsequent layers. This involves the integration of data from all existing branches via element-wise summation:

$$\mathbf{U} = \tilde{\mathbf{U}} + \hat{\mathbf{U}}, \tag{2}$$

The integrated information is embedded via global average pooling to compute channel-wise statistics, $s \in \mathbb{R}^{C}$. For c channel-wise statistics, the c-th element is computed by shrinking $\mathbf{U}$ through spatial dimensions $H \times W$:

$$s_c = \mathcal{F}_{gp}(\mathbf{U}_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{U}_c(i, j) \tag{3}$$

This creates a 'compact feature' $\mathbf{z} \in \mathbb{R}^{d \times 1}$, enabling the logical structure for adaptive selection. A fully-connected (fc) layer facilitates this behavior:

$$\mathbf{z} = \mathcal{F}_{fc}(s) = \delta(\mathcal{B}(\mathbf{W}_s)), \tag{4}$$

where δ represents the ReLU function, and $\mathcal{B}$ represents Batch Normalization of $\mathbf{W} \in \mathbb{R}^{d \times C}$. A reduction ratio is then applied to 'control' the generated value and can be expressed as:

$$d = \max(C/r, L), \tag{5}$$

where L represents the minimal value of d.

Select: adaptive selection of different spatial scales of information across channels occurs through soft attention mechanisms controlled by the compact feature descriptor $\mathbf{z}$. In other words, a softmax operator is applied to the channel-wise digits:

$$a_c = \frac{e^{\mathbf{A}_c \mathbf{z}}}{e^{\mathbf{A}_c \mathbf{z}} + e^{\mathbf{B}_c \mathbf{z}}}, b_c = \frac{e^{\mathbf{B}_c \mathbf{z}}}{e^{\mathbf{A}_c \mathbf{z}} + e^{\mathbf{B}_c \mathbf{z}}} \tag{6}$$

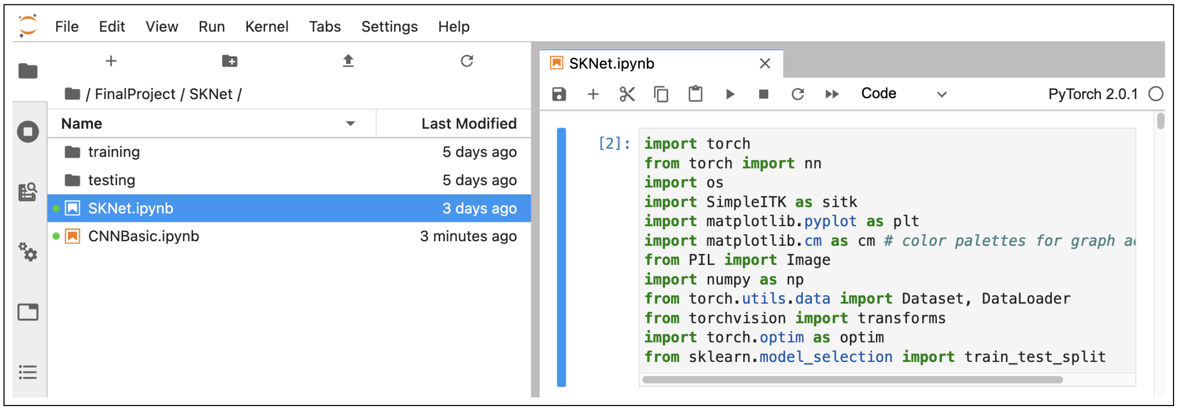
$\mathbf{A}, \mathbf{B} \in \mathbb{R}^{C \times d}$ and $\mathbf{a}, \mathbf{b}$ represent the soft attention vector for $\tilde{\mathbf{U}}$ and $\hat{\mathbf{U}}$ respectively. The resulting feature map $\mathbf{V}$ materializes through the attention weights on various kernels:

$$\mathbf{V}_c = a_c \cdot \tilde{\mathbf{U}}_c + b_c \cdot \hat{\mathbf{U}}_c, a_c + b_c = 1, \tag{7}$$

where $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_C], \mathbf{V}_c \in \mathbb{R}^{H \times W}$.

Appendix B. Model Running Instructions

To run the SKNet and CNN models associated with this report, ensure that your file structure is properly configured in a high-performance computing (HPC) environment (e.g., Rivanna). An acceptable file structure is shown in Figure 9.



/ FinalProject / SKNet /

Name	Last Modified
training	5 days ago
testing	5 days ago
SKNet.ipynb	3 days ago
CNNBasic.ipynb	3 minutes ago

```
[2]: import torch
     from torch import nn
     import os
     import SimpleITK as sitk
     import matplotlib.pyplot as plt
     import matplotlib.cm as cm # color palettes for graph a
     from PIL import Image
     import numpy as np
     from torch.utils.data import Dataset, DataLoader
     from torchvision import transforms
     import torch.optim as optim
     from sklearn.model_selection import train_test_split
```

Figure 9: File Structure Example on Rivanna

Note that we utilize the PyTorch 2.0.1 kernel on Rivanna while running the code. It is critical to use a kernel that supports the libraries shown in the first code cell in Figure 9.

The documentation provided in `SKNet.ipynb` and `CNNBasic.ipynb` serves as a guide for users to understand what a particular cell does and how it is used in the modeling framework. It is recommended to run the individual cells consecutively such that the libraries and data will load properly. Note that running the SKNet and CNN models takes several minutes of processing time. Performance metrics are written to `.csv` files in the same file path pictured in Figure 9.

To run the code within JupyterLab on the Rivanna cluster, it is recommended to configure a session with the computing options shown in the table below.

Configuration	**Value**
Rivanna partition	GPU
Number of hours	12
Number of cores	8
Memory request in GB	100
Work directory	SCRATCH
GPU type for GPU partition	default
Number of GPUs	2

Please contact the authors if any issues arise from attempting to run the code from this report.