PHOTO: DARCY PETERKA

SeasonScope: A Comprehensive System for Season Classification and Captioning for Outdoor Images

COMS W4995 Deep Learning for Computer Vision

Haowen Xu (hx2364) Jennifer Duan (jd3794) Shiying Chen (sc5299)

Introduction

- Season classification on outdoor images is challenging due to varying geographic and temporal factors
- SeasonScope aims to not only classify the images into four seasons but also enrich them with descriptive captions

Dataset

- Season Images: https://storage.googleapis.com/4995-dlcv-project-data/season_images.zip
  - Use Unsplash API to scrape 24,000 outdoor images (6,000 per season)
  - Label each image with corresponding season
- Image Captions: https://storage.googleapis.com/4995-dlcv-project-data/season_captions.txt
  - Use BLIP model to create captions for 6,000 images (1,500 per season)

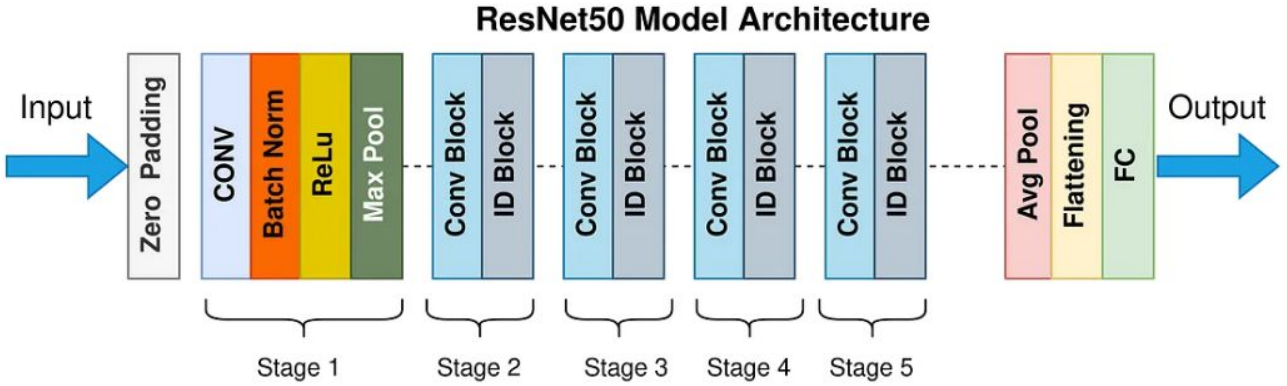
Season Classification

Method: Classification

- Transfer Learning using Pre-trained Models
  - VGG19
  - Resnet-50
  - GoogLeNet
  - DenseNet
  - EfficentNetV2
- Customized CNN Model

Method: Classification

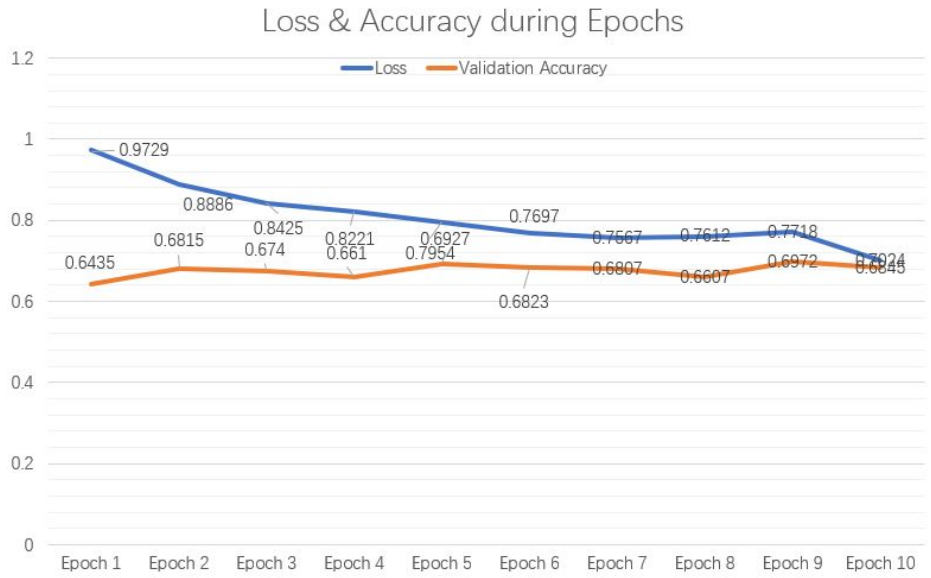
ResNet-50



Resnet-50 Model architecture

Method: Classification

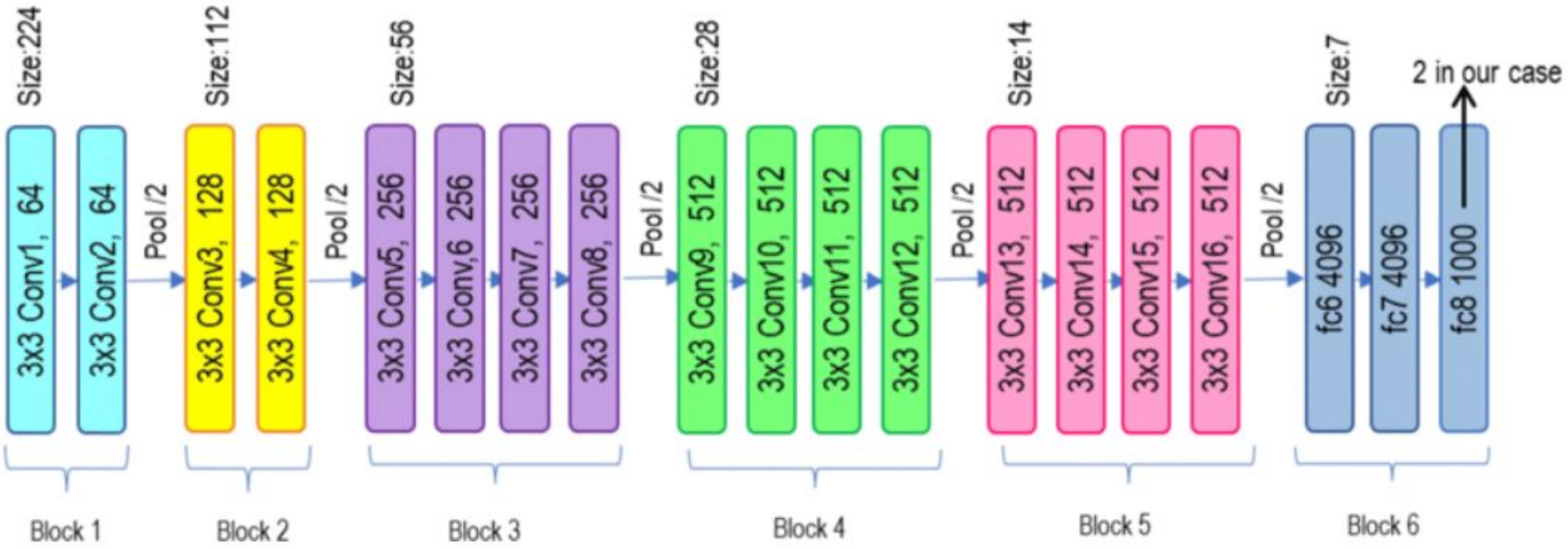
ResNet-50 Result



- The training loss consistently decreases across 10 epochs, indicating learning progress, while the validation accuracy improves from 64.35% to a peak of 69.725%.
- The final test accuracy is 68.8%
- The training process seems to be on the right track, but there is still room for improvement, especially in achieving higher and more stable validation accuracy.

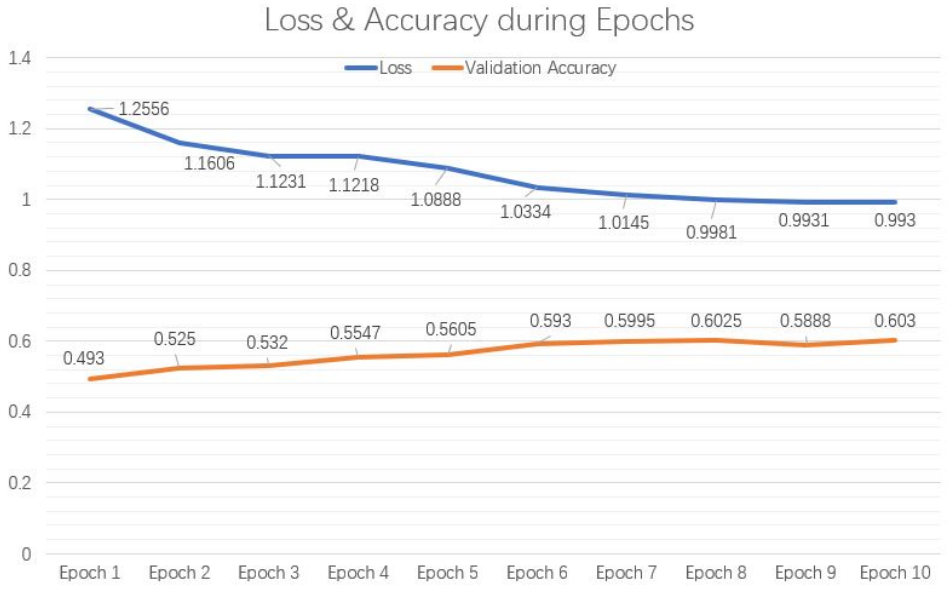
Method: Classification

VGG19



Method: Classification

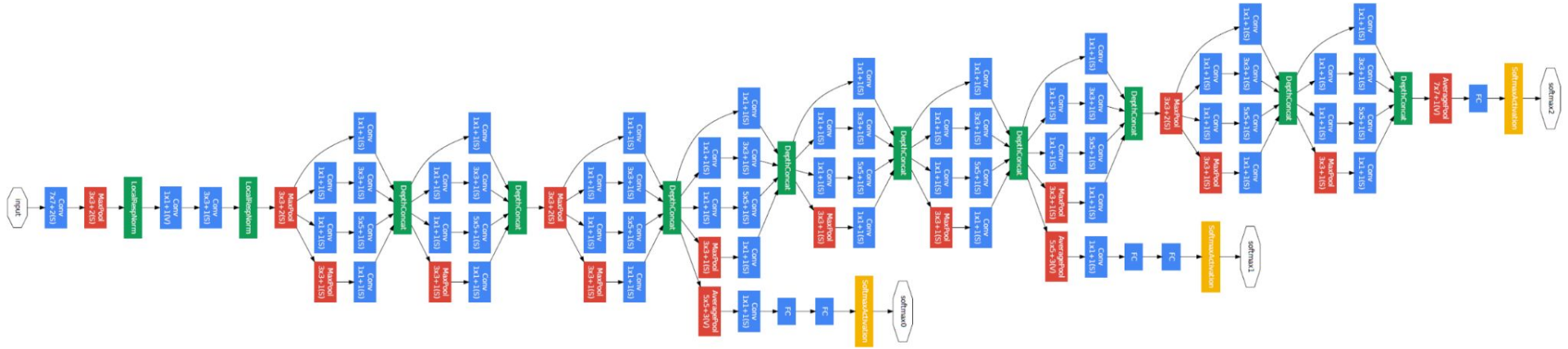
VGG19 Result



- The VGG-19 model shows a steady learning curve over 10 epochs for the 4 season classification task, with consistent decreases in loss and increases in validation accuracy.
- Despite an occasional dip, the model recovers and reaches a final validation accuracy of 60.3%.
- Final test accuracy is 60.3%

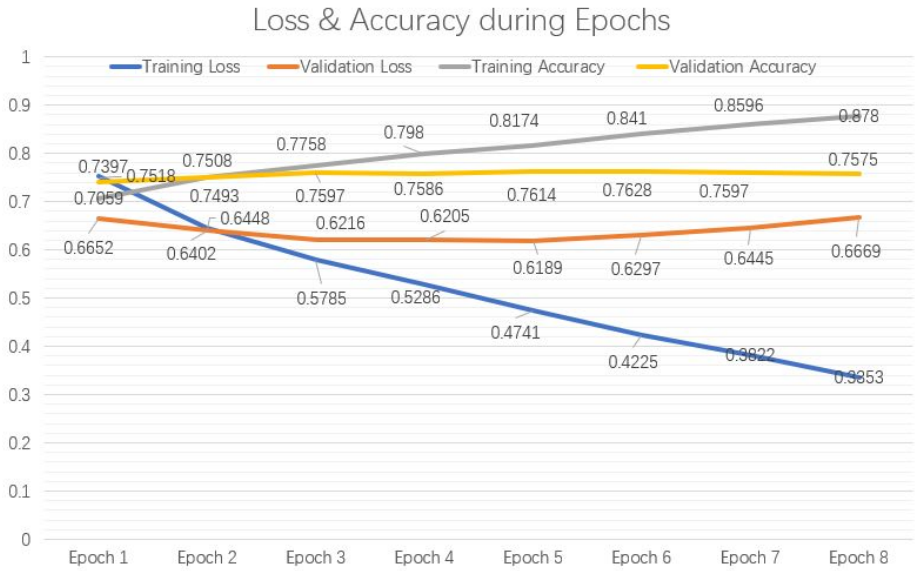
Method: Classification

GoogLeNet



Method: Classification

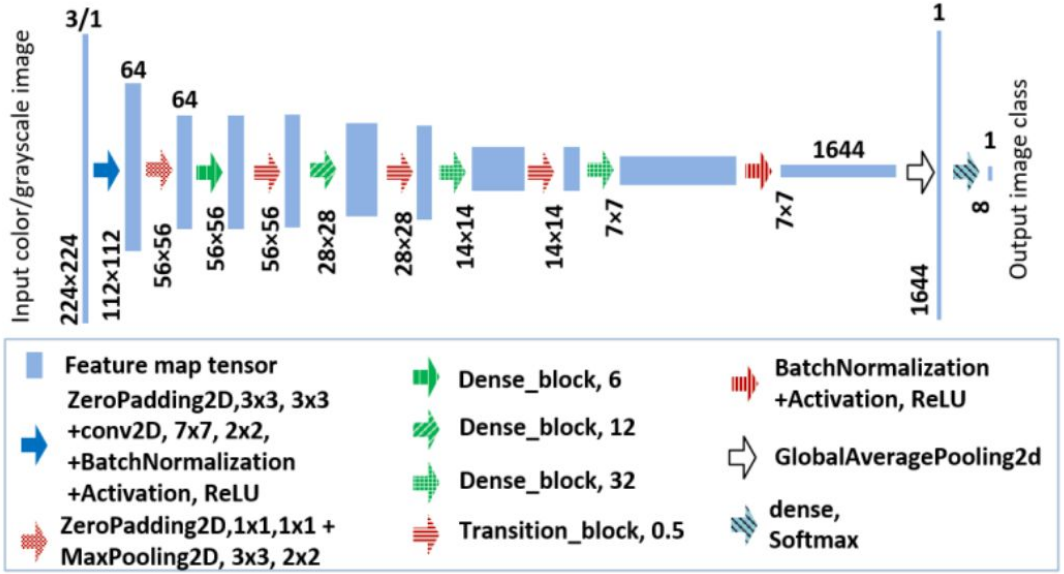
GoogLeNet Result



- The GoogLeNet model's performance improves steadily during training, with best validation accuracy reaching up to 76.14%. After epoch 5, the model starts to be overfitting.
- The model achieves 73.72% on test accuracy.
- The model is performing well but appears to have hit a limit in its learning progress.

Method: Classification

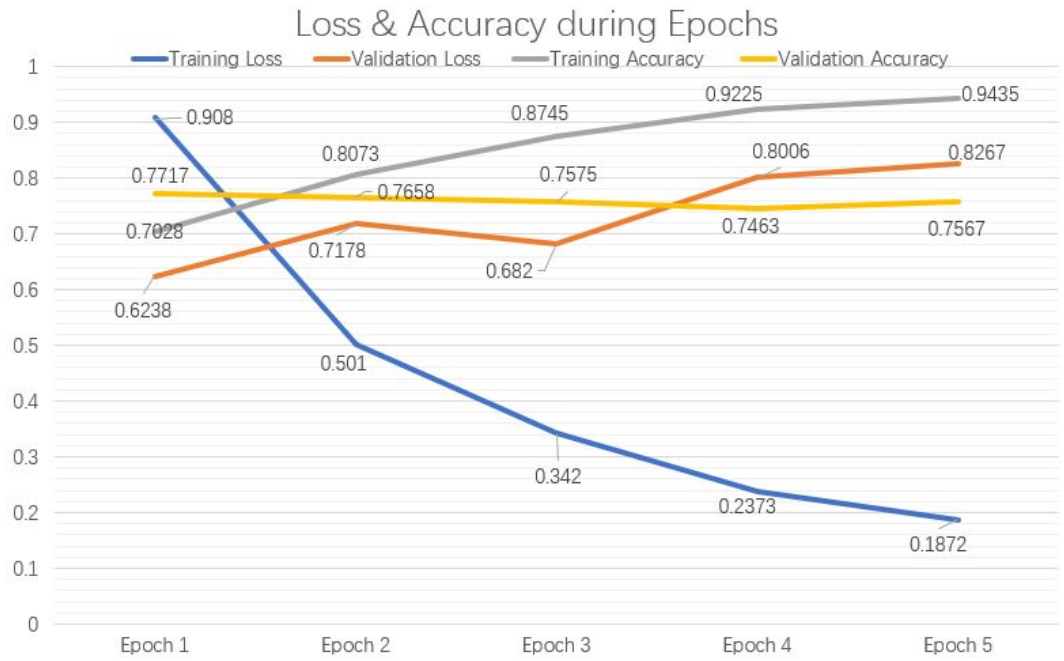
DenseNet



It connects each layer to every other layer directly. This ensures maximum information flow between layers, improving efficiency and reducing the vanishing gradient problem.

Method: Classification

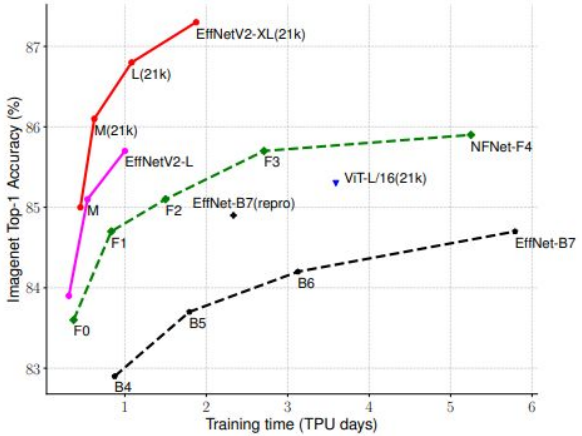
DenseNet Result



- Shows significant improvement in training loss and accuracy over five epochs, indicating that the model is learning effectively from the training data.
- However, the validation results tell a different story: despite an initial drop in validation loss and a peak in validation accuracy in the first epoch, both validation loss and accuracy worsen in subsequent epochs.
- This divergence between training and validation performance suggests that the model is overfitting to the training data.

Method: Classification

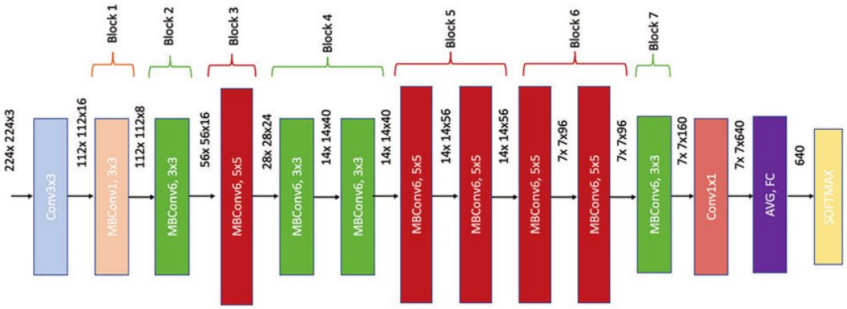
EfficientNetV2



(a) Training efficiency.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (ours)
Top-1 Acc.	84.3%	84.0%	83.1%	83.9%
Parameters	43M	164M	86M	24M

(b) Parameter efficiency.



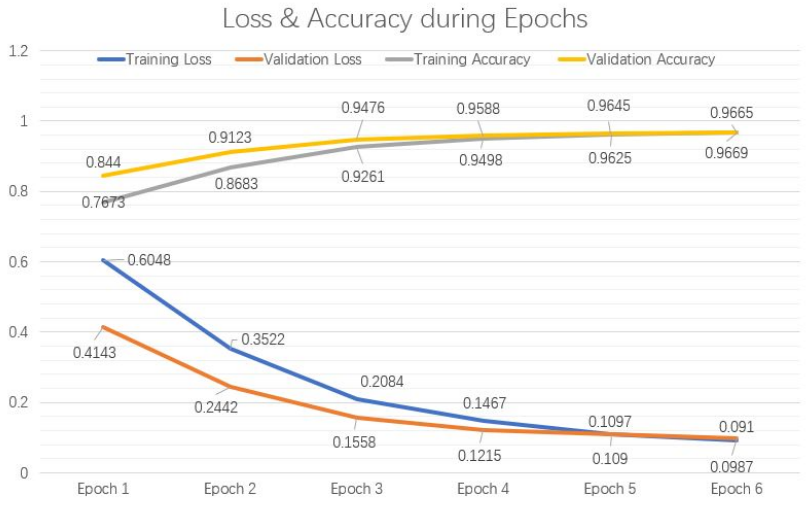
EfficientNetV2 Architecture

Optimized Training: Designed with an improved training process, which makes it more efficient and faster to train than its predecessors.

Better Performance: Shown better performance on several benchmarks compared to its predecessors. It achieves this superior performance even though it requires less computational resources, making it a more efficient model.

Method: Classification

EfficientNetV2 Result



The training results indicate that the EfficientNetV2 model has performed well on the 4-season classification task.

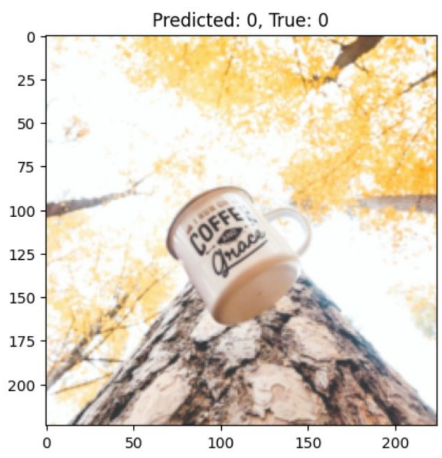
- The training and validation loss consistently decreased
- The training and validation accuracy improved steadily
- The model was not only able to fit the training data well but also generalize to unseen validation data.
- No apparent sign of overfitting or underfitting

The training process appears to be successful and stable, making the model a strong candidate for this classification task.

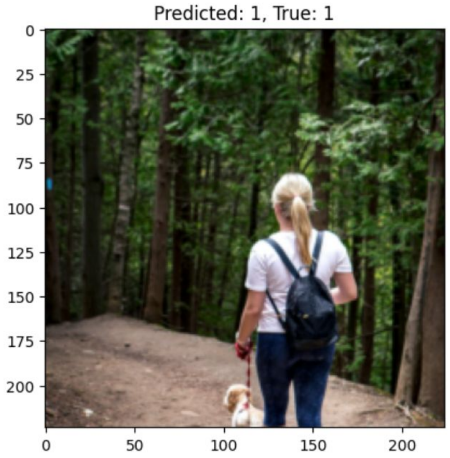
Result: Classification

- Achieved over 96% accuracy on test dataset

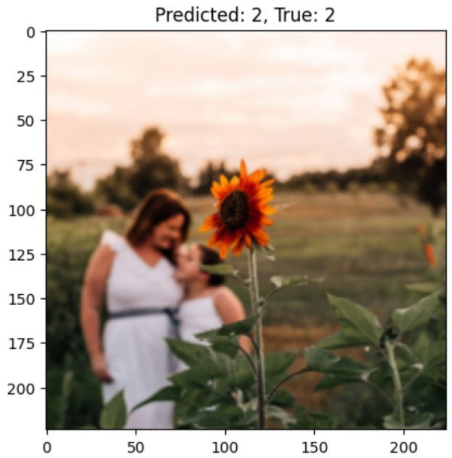
```
Testing Loss: 0.1029 - Testing Accuracy: 0.9613
```



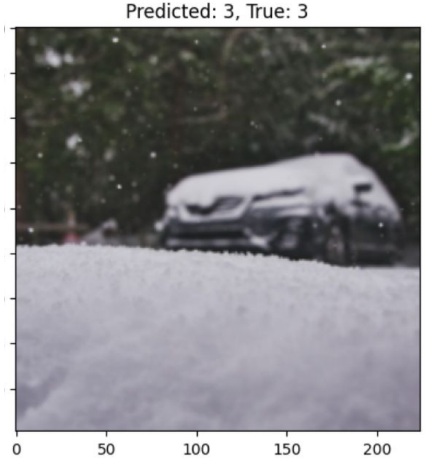
Fall



Spring



Summer



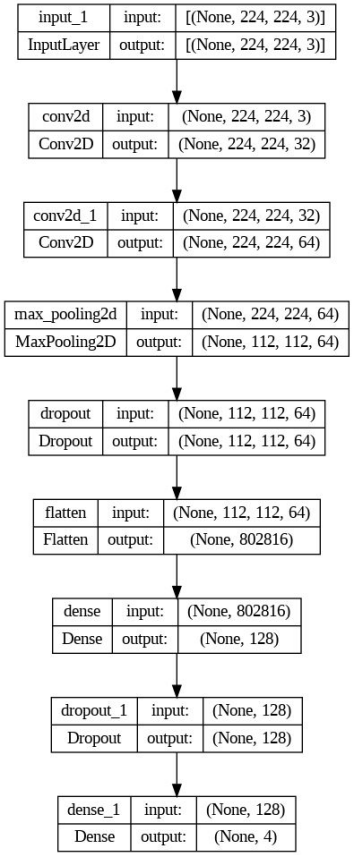
Winter

Method: Classification

Customized CNN Model

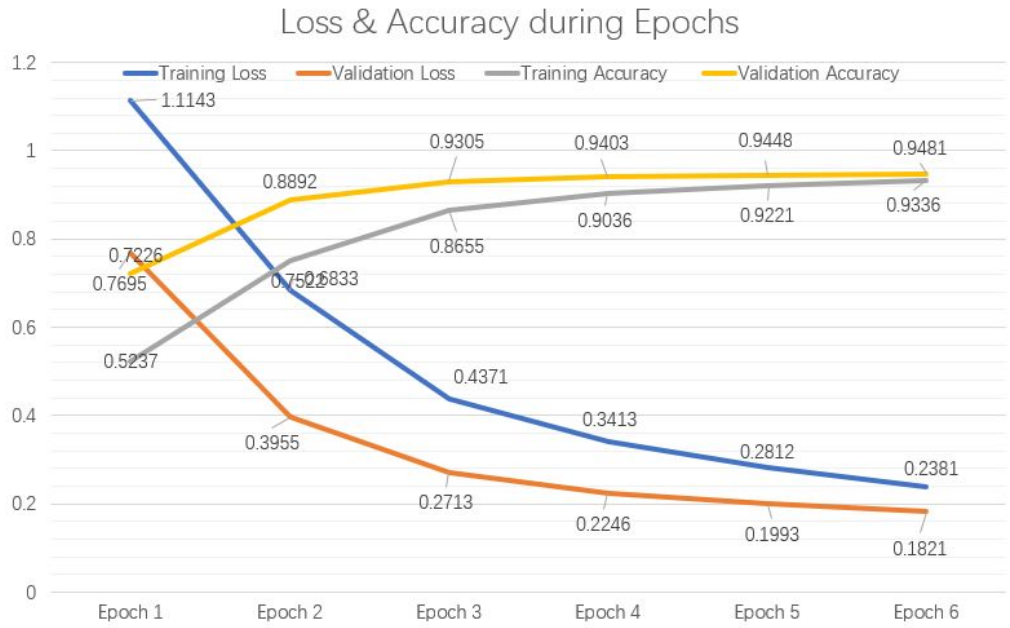
A simple CNN that takes in 3-channel images. It consists of:

- Two convolutional layers
- After each convolution, ReLU activation is applied
- A max-pooling layer with a 2x2 kernel size to reduce spatial dimensions by half.
- Dropout is applied after each convolutional layer (25% after the first and 50% after the second) to prevent overfitting
- The flattened output of the convolutional layers is fed into two fully connected layers
- A log softmax function



Method: Classification

Customized CNN Model Result



Both training and validation accuracies have consistently improved across the 6 epochs, indicating effective learning and good generalization to unseen data.

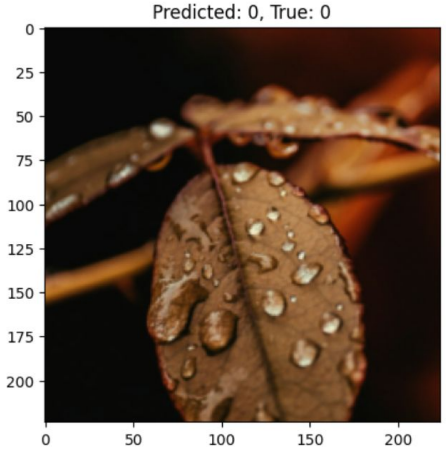
The training loss and validation loss have both decreased well, further demonstrating model effectiveness.

The model appears robust and well-performing for this classification task.

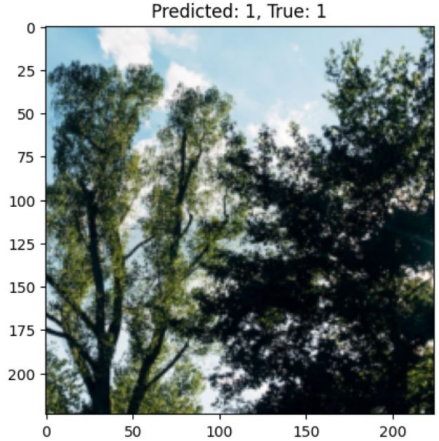
Result: Classification

- Achieved over 95% accuracy on test dataset

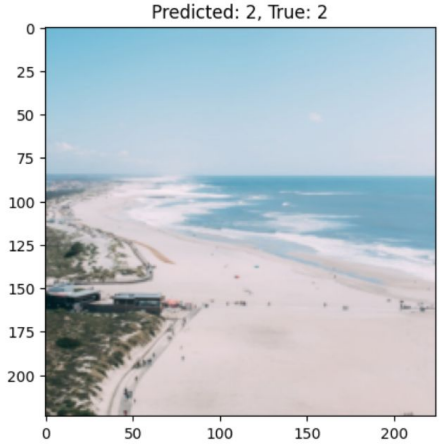
```
Testing Loss: 0.1287 - Testing Accuracy: 0.9593
```



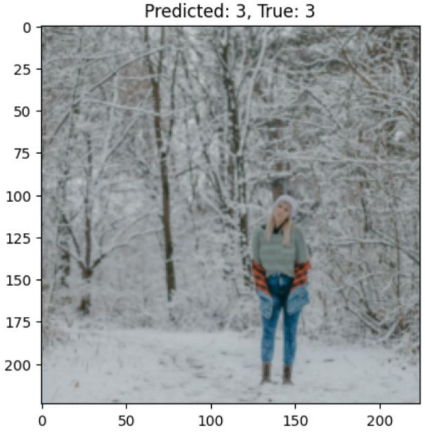
Fall



Spring



Summer

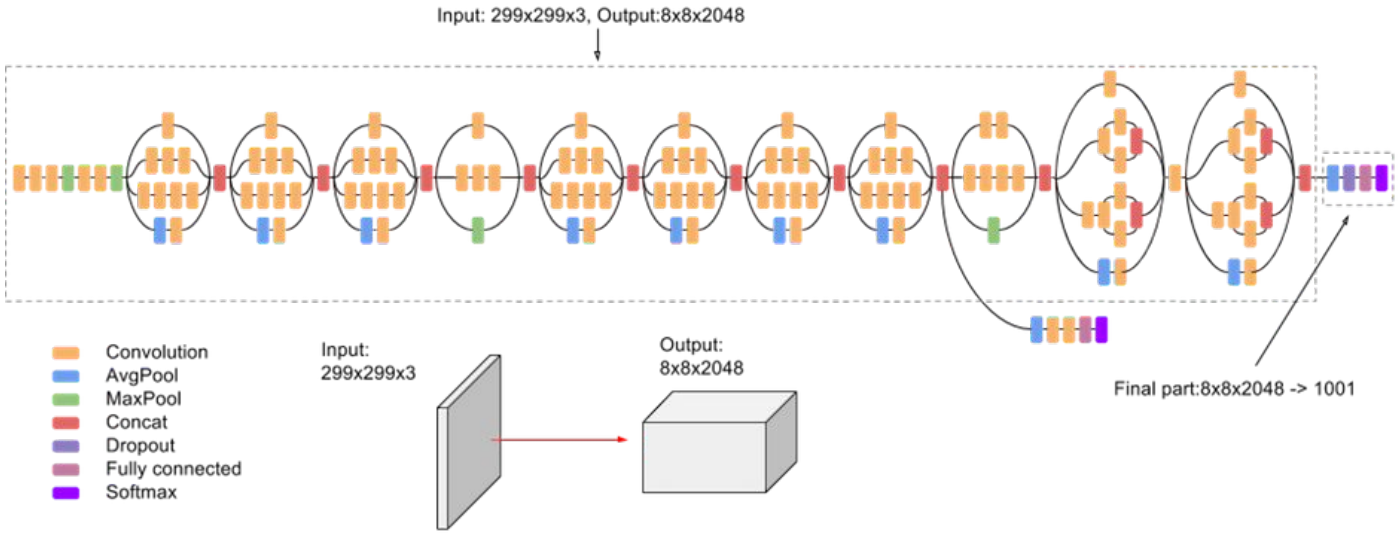


Winter

Image Captioning

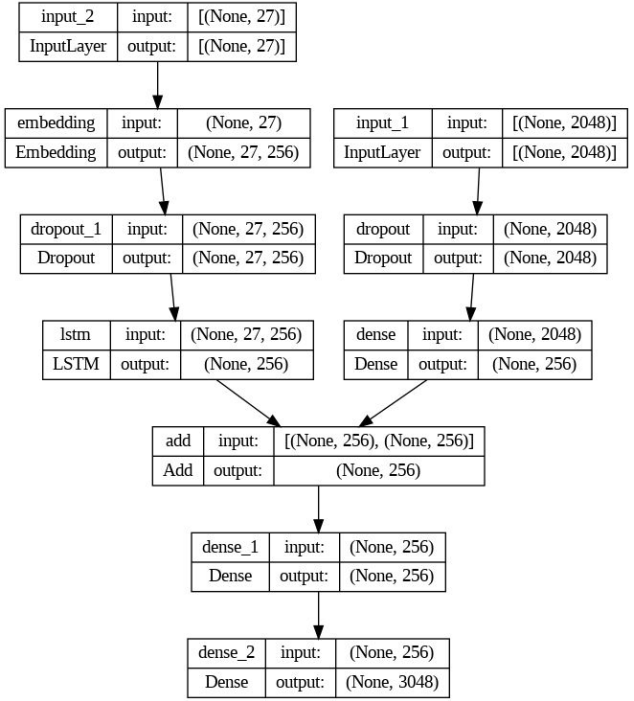
Method: Image Captioning

- Use CNN model for feature extraction: Inception V3



Method: Image Captioning

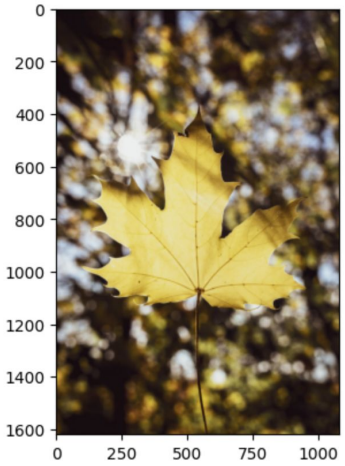
- Use LSTM for caption generation
  - Two inputs: CNN features and tokenized captions
  - Pass word embeddings into LSTM for sequence processing
  - Merge CNN features and LSTM outputs
  - Pass the merged output into dense decoder layers
  - Output a probability distribution via softmax for the next word in the caption



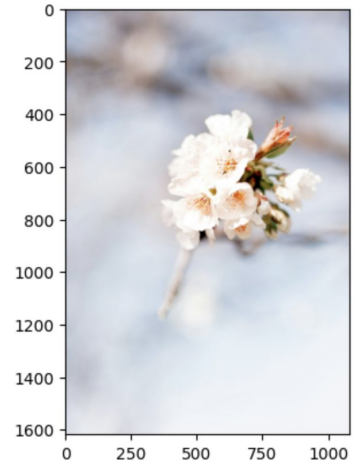
Result: Image Captioning

- Captions generated for test images are contextually accurate and descriptive

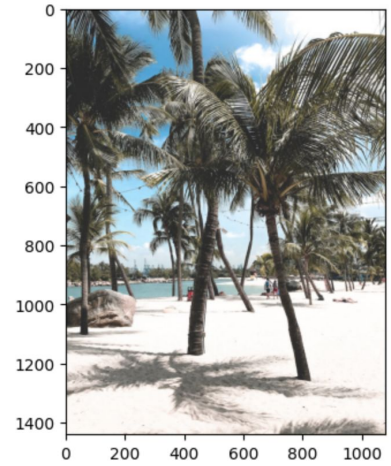
<START> autumn leaves in the shade <END>
<matplotlib.image.AxesImage at 0x7fe080511150>



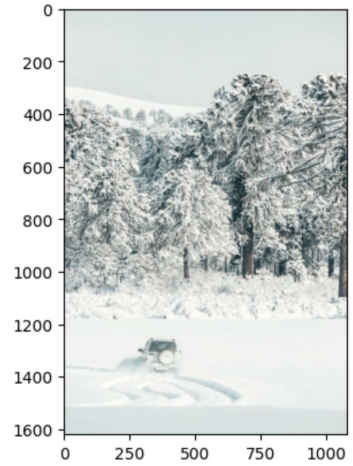
<START> a white flower with blurry petals <END>
<matplotlib.image.AxesImage at 0x7fe080b9b3a0>



<START> a beach with palm trees <END>
<matplotlib.image.AxesImage at 0x7fe0808bc640>



<START> snow covered trees in the snow <END>
<matplotlib.image.AxesImage at 0x7fe082810160>



Conclusion

- We are able to achieve high test accuracy (over 95%) using EfficientNetV2 model and our customized CNN model in season classification tasks
- Our image captioning model can generate contextually accurate and descriptive captions for unseen outdoor images relevant to seasons

Discussion

- Data
  - Collect more images that incorporate more variances in lightness, angle, color, as well as number of objects in the frame
  - Create more referenced captions for each image to make the training vocabularies more diverse
- Model
  - Explore the possibility of utilizing CNN model with better season classification performance to improve feature extraction prior to caption generation
  - Use rigorous evaluation metrics to provide quantitative measures on generated captions

Thank you for your listening!