Sports Balls Multiclass Classification Using Pytorch

Anthony Kommareddy

November 2024

1 Overview

This project focuses on multi-class image classification of sports balls, leveraging a pre-trained ResNet26d model architecture. Using PyTorch and the timm library, two model versions were developed and evaluated. Key aspects included training performance, testing accuracy, and inference optimization. Results were obtained for both versions, providing a comparative analysis to determine the more effective model configuration.

2 Dataset

The dataset, sourced from Kaggle (Samuel Cortinhas's sports balls dataset), consists of labeled images of different sports balls, such as soccer balls, basketballs, and baseballs. The dataset was split into training and testing sets to evaluate the models.

- **Total Images:** 9,000
- **Classes:** 15 (Basketball, Football, Bowling ball, Gold ball, Etc.)
- **Train/Test Split:** 7,200 images for training; 1,800 for testing
- **Preprocessing:** Images resized to 224x224, normalized to ImageNet standards.

3 Sample Images from Dataset

Figure 1: Sample Sports balls images from the dataset

4 Model Versions and Adjustments

4.1 Model 1

- Architecture: ResNet26d (pre-trained on ImageNet)
- Pooling Layer: Max pooling was used in the classification layer.
- Optimizer: Adam optimizer with a learning rate of 0.001.

- Loss Function: Cross-Entropy Loss.
- Epochs: 5
- Batch Size: 32

4.1.1 Results for Model 1

- Training Loss and Accuracy:
  - Final epoch training loss: 90%
  - Final epoch training accuracy: 65%
- Testing Loss and Accuracy:
  - Final epoch test loss: 91%
  - Final epoch test accuracy: 73%

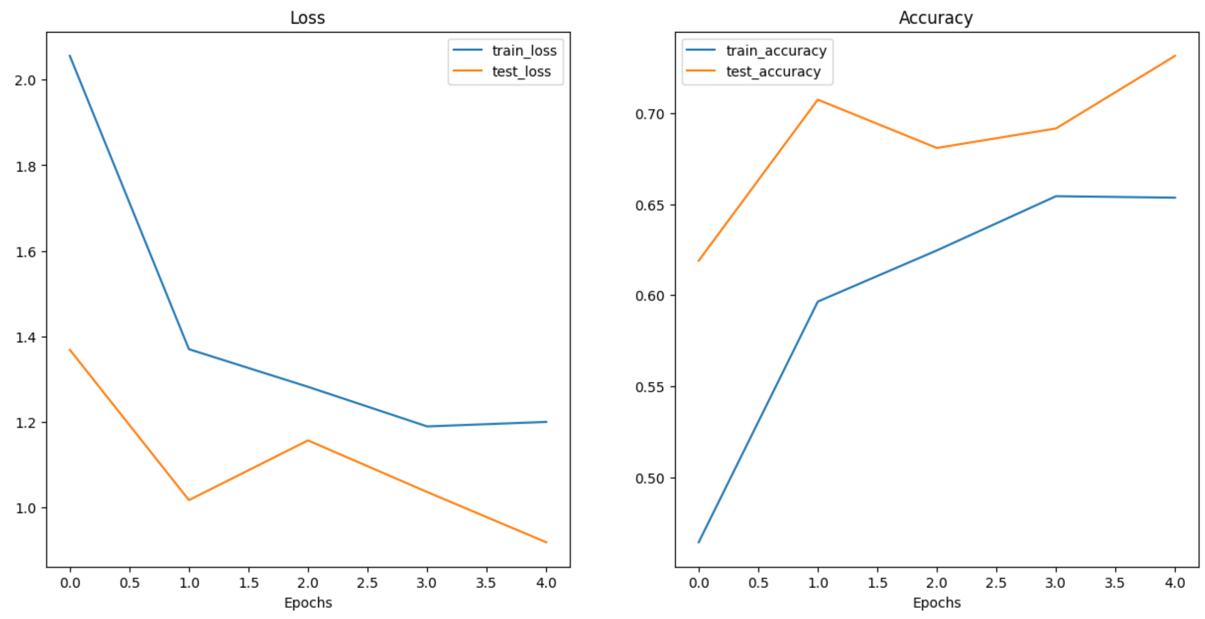


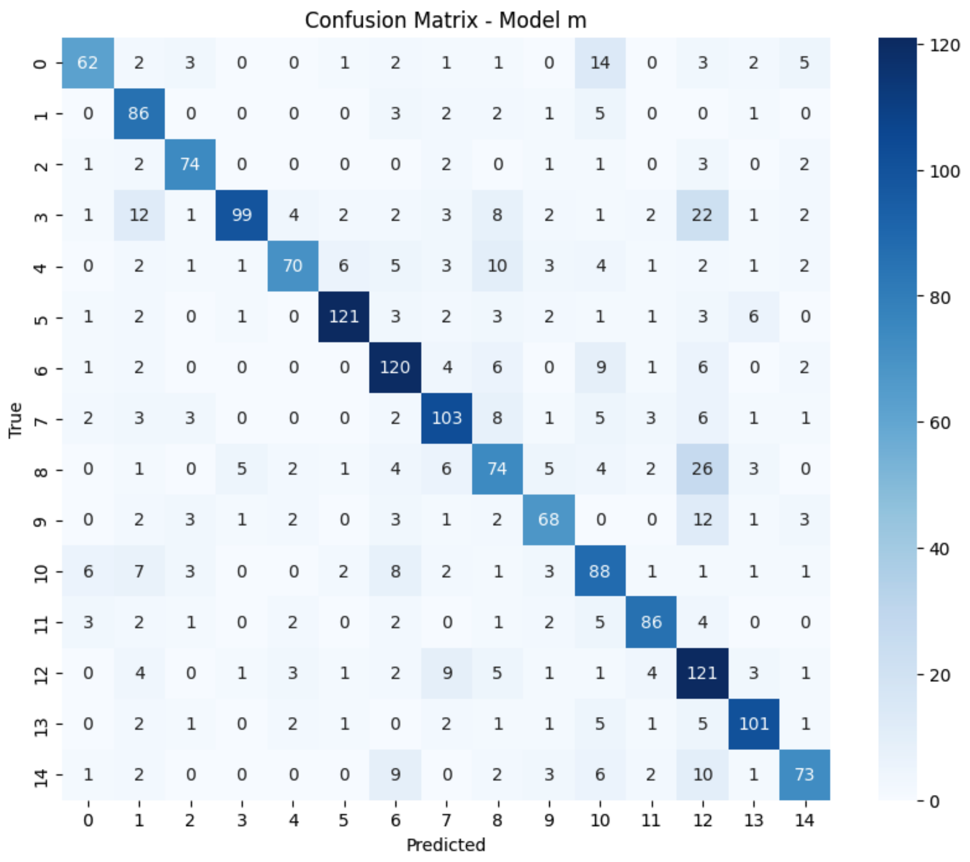
Figure 2: Model version 1 Training and Testing Loss and Accuracy curves across epochs

Loss Curve:

- **Training Loss:** The blue line demonstrates a consistent decline in training loss over the epochs, indicating that the model is successfully learning and reducing errors on the training data.
- **Testing Loss:** The orange line exhibits notable fluctuations and maintains higher values than the training loss, which may indicate overfitting. This suggests that the model's performance on the test data is inconsistent, possibly due to factors like model complexity or variations in the test data.

Accuracy Curve:

- **Training Accuracy:** The blue line displays a consistent rise, reaching approximately 98% by the final epoch, indicating that the model fits the training data effectively.
- **Testing Accuracy:** The orange line varies considerably, with a drop around the first epoch followed by a recovery. This variation suggests potential overfitting, as the model faces challenges in generalizing well to new data.



True \ Predicted	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	62	2	3	0	0	1	2	1	1	0	14	0	3	2	5
1	0	86	0	0	0	0	3	2	2	1	5	0	0	1	0
2	1	2	74	0	0	0	0	2	0	1	1	0	3	0	2
3	1	12	1	99	4	2	2	3	8	2	1	2	22	1	2
4	0	2	1	1	70	6	5	3	10	3	4	1	2	1	2
5	1	2	0	1	0	121	3	2	3	2	1	1	3	6	0
6	1	2	0	0	0	0	120	4	6	0	9	1	6	0	2
7	2	3	3	0	0	0	2	103	8	1	5	3	6	1	1
8	0	1	0	5	2	1	4	6	74	5	4	2	26	3	0
9	0	2	3	1	2	0	3	1	2	68	0	0	12	1	3
10	6	7	3	0	0	2	8	2	1	3	88	1	1	1	1
11	3	2	1	0	2	0	2	0	1	2	5	86	4	0	0
12	0	4	0	1	3	1	2	9	5	1	1	4	121	3	1
13	0	2	1	0	2	1	0	2	1	1	5	1	5	101	1
14	1	2	0	0	0	0	9	0	2	3	6	2	10	1	73

Figure 3: Model version 1 Confusion Matrix

Observations:
The confusion matrix reveals that the model correctly classifies the majority of samples across most classes, as indicated by high values along the diagonal. This suggests that the model performs well in identifying these classes accurately. However, there are noticeable misclassifications, particularly in classes such as 0, 7, and 8. For example, Class 0 has 14 instances misclassified as Class 10, and Class 7 shows misclassifications across several other classes, with small but frequent errors in classes 1, 4, 5, and 10. This pattern suggests that certain classes have overlapping features that make them challenging for the model to differentiate.

In addition, the model seems to confuse specific classes with one another, such as Classes 6 and 5 (6 instances misclassified) and Classes 6 and 11 (9 instances misclassified). This recurring confusion might indicate similarities in feature space for these pairs of classes. On the other hand, some classes, like Classes 2 and 5, show fewer off-diagonal misclassifications, suggesting that the model is more confident and accurate in identifying these classes. Classes like 10 and 12 also show fewer errors, which may imply that these classes have more distinct features.

Overall, while the model demonstrates a good level of accuracy, it struggles with distinguishing between certain classes that may share similar characteristics. This indicates potential areas for improvement, such as additional feature engineering or regularization techniques, to help the model better separate these challenging classes.

4.2 Model 2

- Architecture: Same ResNet26d model with modified pooling and optimizer.
- Pooling Layer: Average pooling.
- Optimizer: SGD with a learning rate of 0.01 and momentum of 0.8.

- Loss Function: Cross-Entropy Loss.
- Epochs: 5
- Batch Size: 32

4.2.1 Results for Model 2

- Training Loss and Accuracy:
  - Final epoch training loss: 49%
  - Final epoch training accuracy: 84%
- Testing Loss and Accuracy:
  - Final epoch test loss: 46%
  - Final epoch test accuracy: 87%

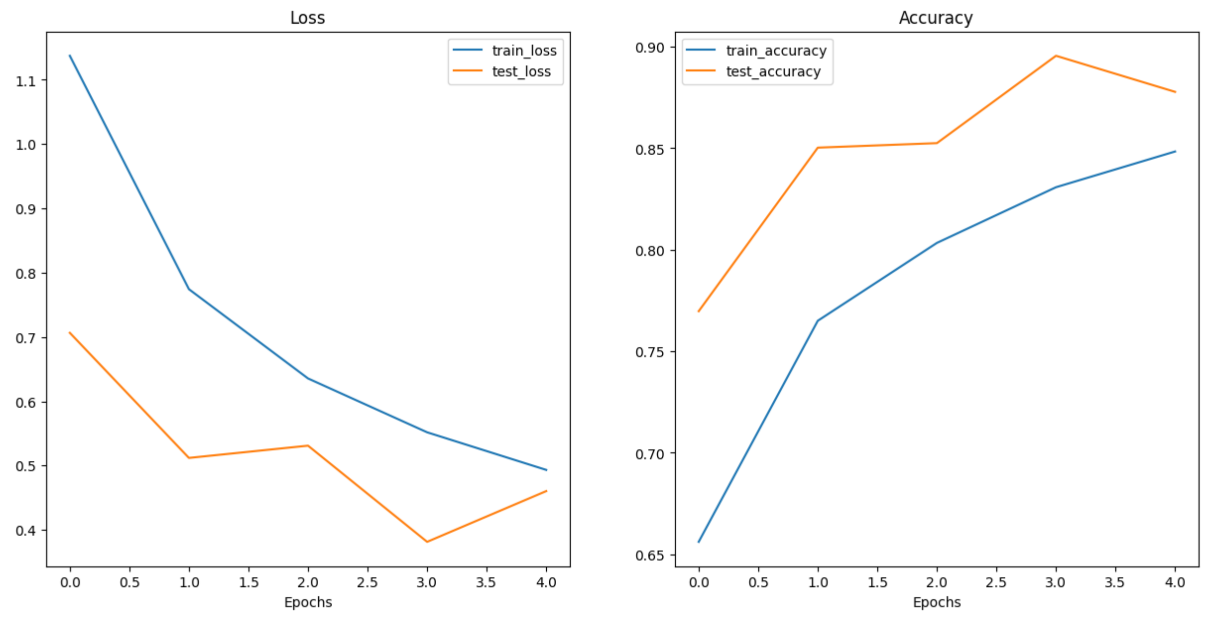
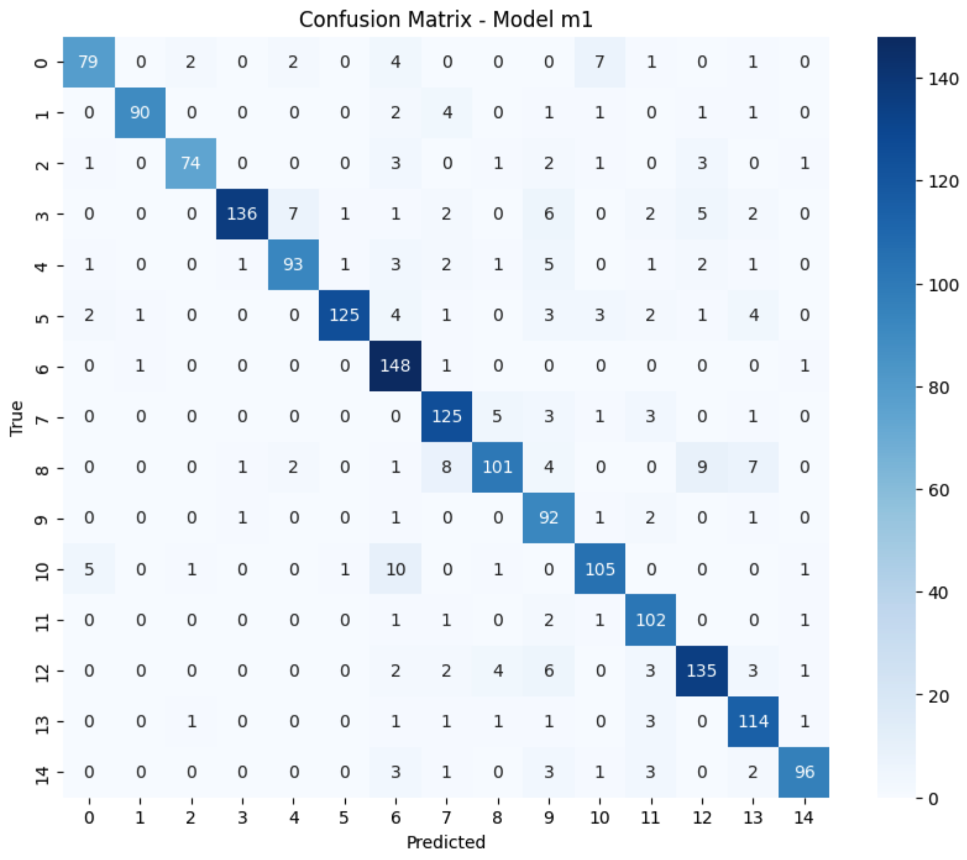


Figure 4: Model version 2 Training and Testing Loss and Accuracy curves across epochs

Loss Curve: The training and testing loss both decrease consistently as the number of epochs increases, showing effective learning. The training loss starts at around 0.4 and drops to below 0.05, while the test loss follows a similar trend, starting higher but converging closely with the training loss by the final epoch. This convergence suggests the model generalizes well to new data without significant overfitting.

Accuracy Curve: Both training and testing accuracy improve steadily, with training accuracy approaching around 98% and testing accuracy reaching close to 97%. This high accuracy and minimal gap between the curves indicate strong model performance on unseen data.



True \ Predicted	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	79	0	2	0	2	0	4	0	0	0	7	1	0	1	0
1	0	90	0	0	0	0	2	4	0	1	1	0	1	1	0
2	1	0	74	0	0	0	3	0	1	2	1	0	3	0	1
3	0	0	0	136	7	1	1	2	0	6	0	2	5	2	0
4	1	0	0	1	93	1	3	2	1	5	0	1	2	1	0
5	2	1	0	0	0	125	4	1	0	3	3	2	1	4	0
6	0	1	0	0	0	0	148	1	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	125	5	3	1	3	0	1	0
8	0	0	0	1	2	0	1	8	101	4	0	0	9	7	0
9	0	0	0	1	0	0	1	0	0	92	1	2	0	1	0
10	5	0	1	0	0	1	10	0	1	0	105	0	0	0	1
11	0	0	0	0	0	0	1	1	0	2	1	102	0	0	1
12	0	0	0	0	0	0	2	2	4	6	0	3	135	3	1
13	0	0	1	0	0	0	1	1	1	1	0	3	0	114	1
14	0	0	0	0	0	0	3	1	0	3	1	3	0	2	96

Figure 5: Model version 2 Confusion Matrix

Observations:
This confusion matrix demonstrates that the model has high accuracy across most classes, as indicated by the strong diagonal values. The majority of samples are correctly classified within their respective classes, suggesting that the model performs well overall. However, there are some misclassifications, though fewer than the previous model, indicating an improvement in generalization.

For example, Class 0 has a few misclassifications, with 7 samples incorrectly predicted as Class 10. Similarly, Class 10 has a minor misclassification issue, with 10 samples misclassified as Class 5. The classes showing fewer off-diagonal values, such as Classes 6, 3, and 11, indicate that the model is more confident and accurate in identifying these classes, which may have more distinct features. Additionally, Class 8 shows some spread in misclassifications across classes 7 and 9, indicating potential overlap in feature space for these specific classes.

Overall, this matrix suggests that the model has improved in reducing misclassifications compared to other models, with fewer errors spread across the classes. However, there remains some minor confusion in classes with overlapping features, suggesting that further refinement could help the model achieve even better performance.

Model Export and Inference Optimization

The models were exported in an optimized format for inference, ensuring efficient deployment. Several optimizations were applied, such as:

- Device Selection: CUDA/GPU support, if available, to accelerate inference.
- Export: The trained model was exported in a form compatible with high-performance serving.

5 Comparison

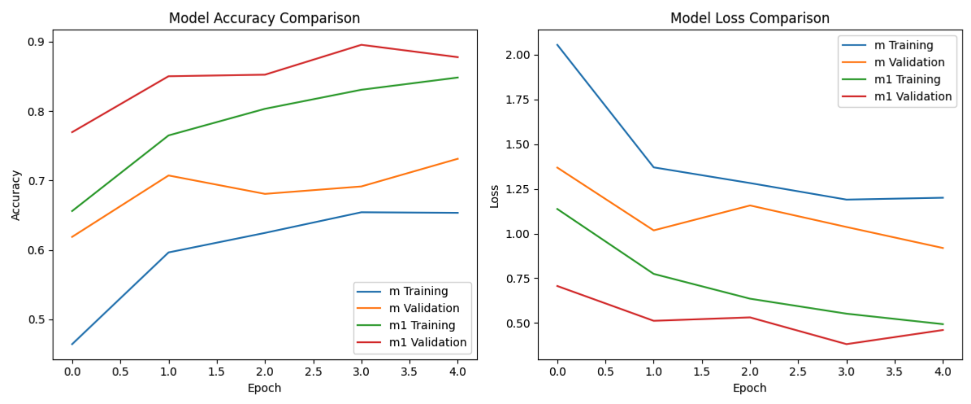


Figure 6: Model version 2 Confusion Matrix

The graph compares the performance of two models, in terms of accuracy and loss over five epochs.

In the accuracy comparison (left graph), model 2 consistently outperforms model 1 in both training and validation accuracy. Model 2 starts with a higher training accuracy, reaching around 0.9 by the third epoch, while model ”m” starts lower and improves steadily, reaching approximately 0.75 by the fourth epoch. Similarly, for validation accuracy, model 2 shows better generalization, maintaining a higher validation accuracy than model 1 throughout the epochs.

In the loss comparison (right graph), model 1 also demonstrates superior performance with lower training and validation loss values compared to model 1. Model 2 starts with a lower loss and continues to decrease steadily, indicating better learning efficiency. On the other hand, model 1 starts with higher loss values and shows slower improvement. Notably, while the model 2 maintains a low validation loss throughout, model 1 experiences a slight increase in validation loss after the third epoch, which could indicate some overfitting. Overall, model 2 performs better in both accuracy and loss metrics, suggesting it is a more effective and generalizable model compared to model 1.

6 Conclusion

The classification task of sports balls using ResNet26d revealed that both model versions were effective, with Model Version 2 yielding slightly better accuracy due to the use of SGD with momentum and average pooling. This improved the model’s ability to generalize on unseen test data. Future improvements could involve experimenting with deeper architectures, hyperparameter tuning, and augmenting the dataset for enhanced accuracy and robustness.

7 Links:

Github: https://github.com/anthonysandesh/cmpe258$_hw1$
Google Colab: https://colab.research.google.com/drive/12wL
Kaggle Dataset: https://www.kaggle.com/datasets/samuelcortinhas/sports-balls-multiclass-image-classification

References

[1] lkk688, “Deepdataminingle arning cmpe pytorch8 2024fall timm.ipy lkk688,” 2024.

[2] “Tutorial 8: Timm models for image classification — deepdataminingle arning 0.1 documentation,” 2023.

[3] G. Singh, “From sgd to adam,” 05 2020.