Real-Time Sports Highlight Extraction Using Deep Learning

Project Group 12
Group Members:

Ella Xu
Yingshen Ye
Yiyang Lin

May 3, 2025

Contents

Abstract

Our report proposes a deep learning model that can automatically identify sports activities from static images. As traditional camera footage only shows the recording for certain periods, when someone is trying to identify certain activities (sports activities in our proposal) from the footage, it can be time consuming and troublesome. We trained and evaluated the on the Kaggle Human Action Recognition (HAR) dataset, totaling 12600 training images. In our report, we proposed implementing the HAR model in the camera system where the camera system can detect sports activities and report the activities. We used the static image of various human activities as our training data, as we aim to use this as a proof of concept.

Our report mainly includes two models and their comparison, aiming to figure out the driving elements for neural network model's accuracy. The base model suggests a mix of CNN and Transformer, in which we extract spatial features via a three-stage CNN, then apply multi-head self-attention Transformer blocks. The advanced custom model suggests a concatenation approach for convolutional blocks, which intends to extract hierarchical spatial relationships between the layers.

The base model demonstrates a better balance in recall and F1-score, particularly showing stronger performance in identifying actual sports activity. While the Advanced model achieves higher precision and accuracy overall. However, its significantly lower recall makes it prone to missing important detections.

Taking both stability and practical application into account, we recommend selecting the base CNN+Transformer model if a single choice is required. However, a combined strategy can be explored as it can offer the best of both worlds using comprehensive coverage and selective certainty.

1. Introduction

In today's fast-paced environment, accurately recognizing human activities in real-time from surveillance footage is crucial for applications such as security monitoring, sports analytics, and smart environments. Traditional video review processes are time-consuming and labor-intensive, especially when trying to locate and identify specific actions from static frames. To address this challenge, we propose a deep learning-based solution that enables automatic classification of sports activities from still images.

Deep learning models, particularly those based on convolutional neural networks (CNNs) and attention mechanisms, have demonstrated impressive performance in image-based recognition tasks. These models are capable of learning complex spatial and hierarchical features from large-scale datasets, making them well-suited for our use case. By training our models on the Kaggle Human Action Recognition (HAR) dataset, we aim to build a proof-of-concept system that could eventually be deployed in a camera pipeline to detect and report human activities with minimal human intervention.

2. Literature review

2.1 CNN + Transformer for processing images

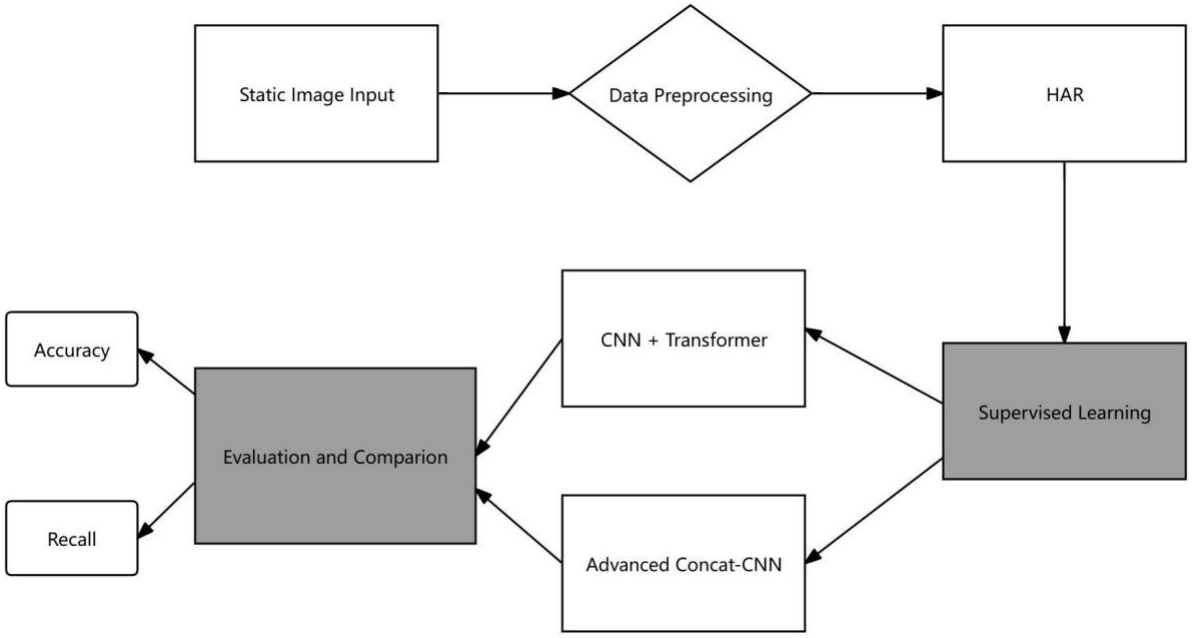
Feiniu Yuan et al. (2023) showed the model by combining CNN and Transformer for medical image segmentation tasks. CNN are used for identifying spatial features, while the Transformer blocks are introduced to model long-range dependencies and global context. This structure enables the model to maintain spatial features while it can also integrate information across far distance regions in an image.

Based on this approach, our base CNN+Transformer model applies a similar fusion: a CNN model to process local features, followed by Transformer blocks to learn global spatial relationships from our data. This architecture is suitable for this project as our project needs to classify actions from still images.

2.2 Blocks of CNN for extracting hierarchical information

Abdellatef et al. (2025) focus on multi-layer convolutional neural networks for human activity recognition. Their research emphasizes how stacking convolutional layers enables the model to learn hierarchical spatial representations.
This concept underlies our Advanced Concat-CNN model, where we employ multiple convolutional blocks and concatenate intermediate feature maps to preserve hierarchical information. By focusing purely on CNNs, this model aims to efficiently learn spatial dependencies at multiple levels with fewer parameters and faster inference speed.

3. Problem description



The goal of our project is to automatically recognize sports activities from static images using deep learning. In many real-world applications, identifying specific actions from still images can be crucial but hard to implement. To address this, we designed and compared two deep learning models to evaluate their effectiveness on the Human Action Recognition dataset.

The first model combines a convolutional neural network (CNN) with a Transformer-based self-attention mechanism. The second model adopts a purely CNN-based approach with concatenated convolutional blocks to learn hierarchical representations.

Our objective is to explore which architectural design better balances performance and generalization. We evaluate the models using accuracy, precision, recall, and F1-score, and visualize their confusion matrices.

4. Database Background and Data Preprocessing

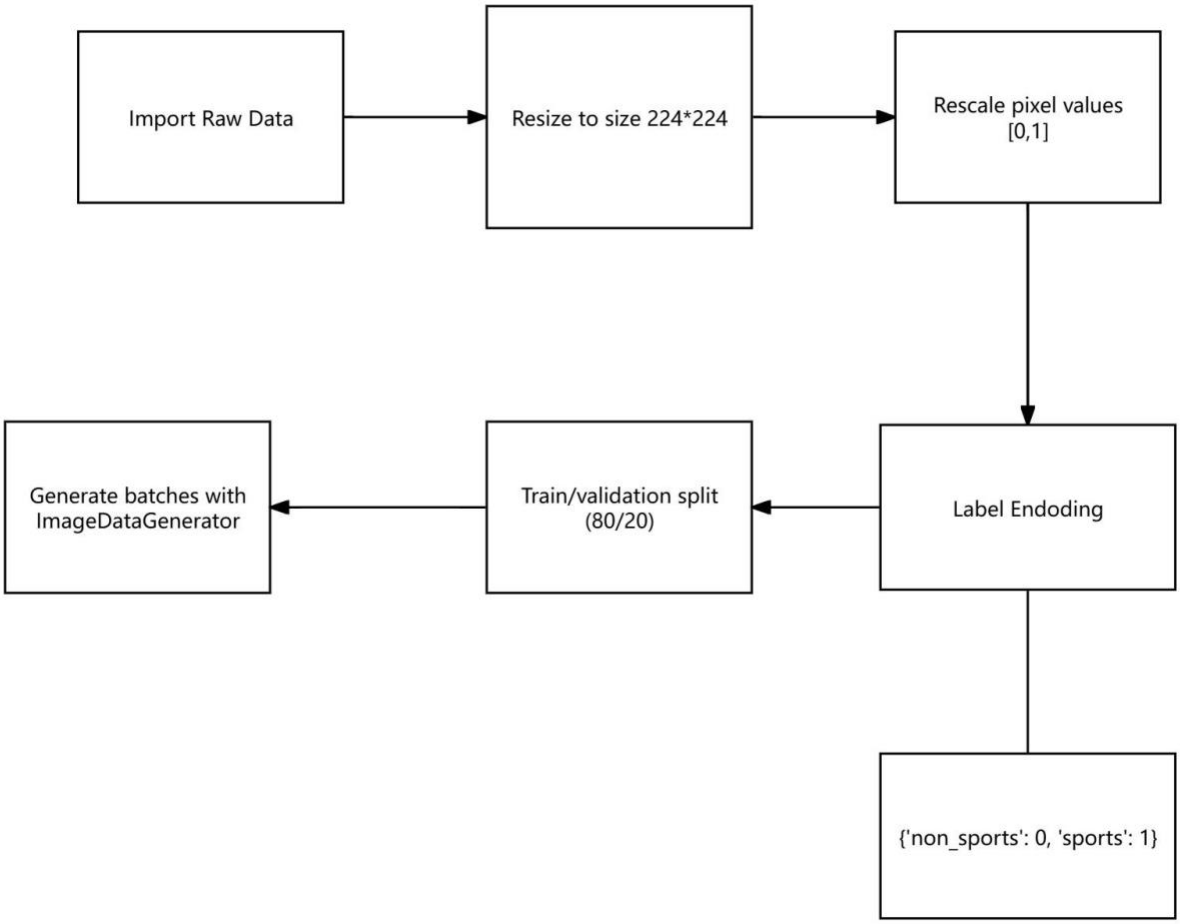


Figure 2 Data Preprocessing Process Diagram

We use a labeled dataset of **12,600 images**, consisting of 9,240 “non-sports” frames and 3,360 “sports” frames. We split the dataset into 80% training set and 20% validation data/test set. All images are resized to **224×224 pixels** and normalized to the [0,1] range. Data generators apply normalization and augmentation (zoom, horizontal flip) for training, and rescaling for validation/testing. The dataset as follow:

```
[ ]  X = "filename"
     Y = "binary_idx"
     train_df, test_df = train_test_split(train_df, test_size=0.2, stratify=train_df[Y], random_state=42)

     print("Shape:", train_df.shape, test_df.shape)
     img_dir     = "/content/Human Action Recognition/train"
     target_size = (224, 224)
     batch_size  = 64
```

```
Shape: (10080, 4) (2520, 4)
Found 10080 validated image filenames belonging to 2 classes.
Found 2520 validated image filenames belonging to 2 classes.
Found 2520 validated image filenames belonging to 2 classes.
```

Figure 3 Data Preprocessing Process

5. Model Architectures for Sports vs. Non-Sports Classification

5.1 Convolutional Neural Network + Transformer

1) Model Description

To fully leverage the advantages of Transformers and Convolutional Neural Networks, we propose a CNN and Transformer Complementary Network sport segmentation. It can further enhance spatial understanding and global context awareness.

The model begins with three sequential Conv2D blocks with increasing filters 64, 128 and 256, each followed by batch normalization, ReLU activation, and 2×2 max pooling, reducing the input image size from 224×224 to 28×28. These spatial features are reshaped into a sequence of 784 tokens, each of dimension 256, and add a learnable positional embedding, which can capture the specific location.

Then apply with a transformer encoder, which includes one layer of multi-head attention, residual connections to keep the original information, a feed-forward neural network to process information, and layer normalization to stabilize training. The blocks used in this transformer include Multi-Head Self-Attention (MHA), LayerNorm, and a position-wise Feed-Forward Network (MLP). The MHA block lets each spatial token (patch) attend to every other one and weight their relevance, capturing global context. Its output is added back to the input (a residual connection), then passed through LayerNorm to keep activations zero-mean and unit-variance. Next comes the position-wise MLP—two Dense layers (first expanding to a higher dimension with ReLU and dropout, then projecting back)—which injects non-linearity and recombines features. Finally, we add in the MLP's input via another residual connection and apply LayerNorm again, producing the block's output for the next layer.

The output goes through a ReLU layer with dropout to prevent overfitting, and ends with a sigmoid layer for binary classification. The complete architecture is shown in Figure 4.

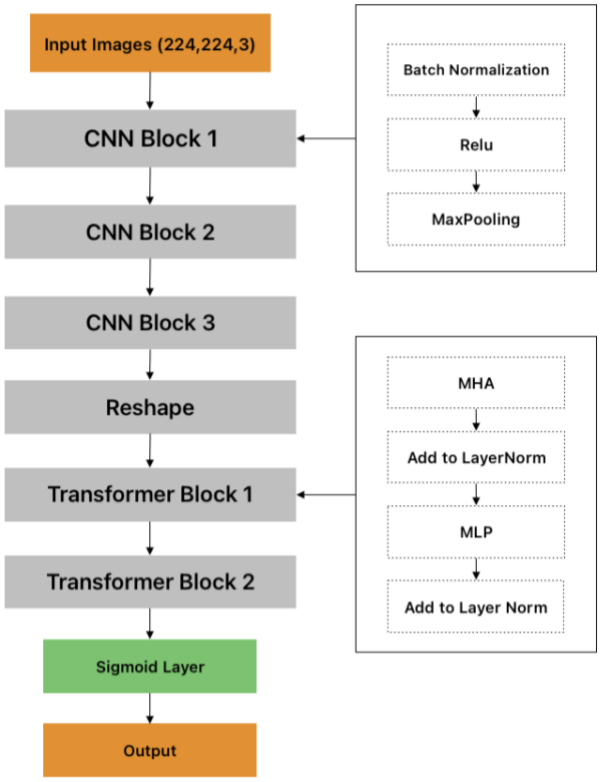


Figure 4 CNN+Transformer Architecture

Through our experimenting on changing CNN block numbers, transformer block numbers, adjusting dropout rate, adjusting l2 regularization weight decay, we chose the dropout rate at 0.5 and l2 = 1e-4, as increasing the drop out would lead to further underfitting by dropping too many activations and further increase the l2 would over-constraint the weight.

2) Data and Experimental Result

a) Basic Model

The training was conducted over a maximum of 50 epochs with early stopping enabled. The model stopped at epoch 13, restoring the best weights from epoch 7. The final training accuracy is about 79.74%, the validation accuracy is about 55.32%, The training session's progression over time is stated in Figure 5, the graph for training progress over epochs is stated in Figure 6:

```
Epoch 1/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 139s 781ms/step - accuracy: 0.7307 - loss: 0.6005 - val_accuracy: 0.5571 - val_loss: 0.7886
Epoch 2/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 685ms/step - accuracy: 0.7550 - loss: 0.5265 - val_accuracy: 0.7361 - val_loss: 0.6011
Epoch 3/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 683ms/step - accuracy: 0.7898 - loss: 0.4786 - val_accuracy: 0.6595 - val_loss: 0.6728
Epoch 4/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 684ms/step - accuracy: 0.7878 - loss: 0.4706 - val_accuracy: 0.7611 - val_loss: 0.4937
Epoch 5/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 685ms/step - accuracy: 0.7777 - loss: 0.4834 - val_accuracy: 0.7226 - val_loss: 0.5580
Epoch 6/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 683ms/step - accuracy: 0.8007 - loss: 0.4758 - val_accuracy: 0.6897 - val_loss: 0.5895
Epoch 7/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 688ms/step - accuracy: 0.8046 - loss: 0.4604 - val_accuracy: 0.7690 - val_loss: 0.4903
Epoch 8/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 689ms/step - accuracy: 0.7974 - loss: 0.4674 - val_accuracy: 0.2762 - val_loss: 1.4380
Epoch 9/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 106s 672ms/step - accuracy: 0.8140 - loss: 0.4459 - val_accuracy: 0.7417 - val_loss: 0.6086
Epoch 10/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 686ms/step - accuracy: 0.7736 - loss: 0.5192 - val_accuracy: 0.4579 - val_loss: 0.8192
Epoch 11/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 684ms/step - accuracy: 0.7943 - loss: 0.4845 - val_accuracy: 0.5179 - val_loss: 0.8645
Epoch 12/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 680ms/step - accuracy: 0.7810 - loss: 0.4930 - val_accuracy: 0.7802 - val_loss: 0.4967
Epoch 13/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 693ms/step - accuracy: 0.7974 - loss: 0.4810 - val_accuracy: 0.5532 - val_loss: 0.7132
Epoch 13: early stopping
Restoring model weights from the end of the best epoch: 7.
```

Figure 5 Result of CNN + Transformer Model

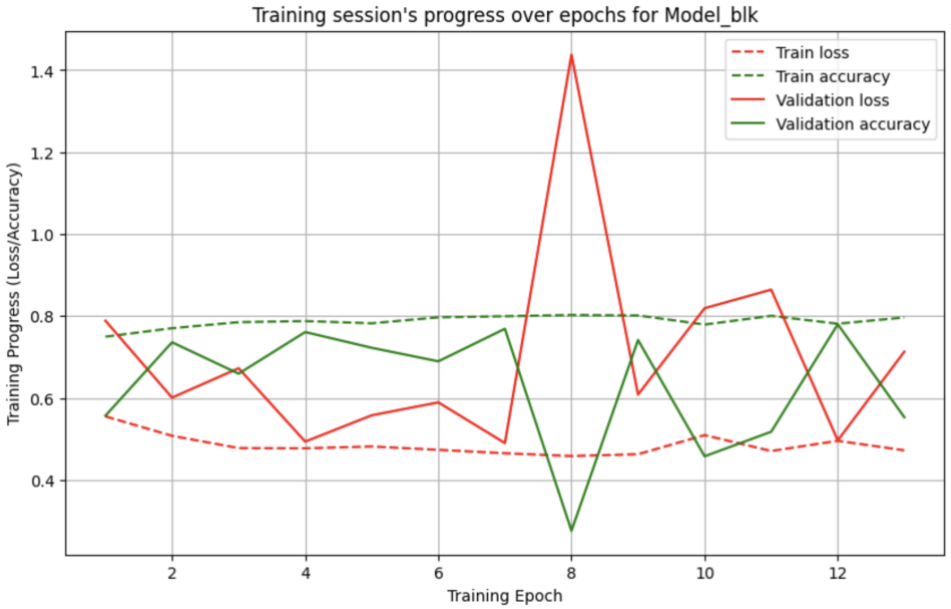


Figure 6 Training progress over epochs for CNN + Transformer Model

b) **Performance Evaluation**

As seen in the result above, during training, the model maintained a training accuracy of 73–80% with relatively stable training loss, indicating consistent convergence in the CNN backbone. While for validation accuracy fluctuated significantly, ranging from 27% to 76%, with corresponding spikes in validation loss, which suggests the model is sensitive to the data in each batch and might be overfitting. As for the learning efficiency, as the epochs increase, we don't see much increase in accuracy, indicating low learning efficiency.

3) Conclusion

The updated CNN + Transformer model exhibits potential for sports frame classification, achieving a 79.74% training accuracy with early stopping. Despite the transformer's ability to incorporate spatial context, the model still exhibits overfitting behavior beyond 10 epochs, as seen in the gap between validation and training performance.

5.2 Human Action Recognition CNN_Blk

1) Model Description

We built a model named CNN_Blk with reference to Abdellatef's HARCNN (Human Action Recognition Convolutional Neural Network) hierarchy model. This model is designed specifically for image-based classification using RGB frames, which conforms to the goals of our project - develop a binary classifier for detecting sports-related content in video frames, enabling automated sports highlight extraction.

Instead of a flat sequential CNN, we implement a multi-branch structure with hierarchical feature concatenation and pooling to understand complex patterns and keep useful information from different time spans. The complete architecture is shown in Figure 7.

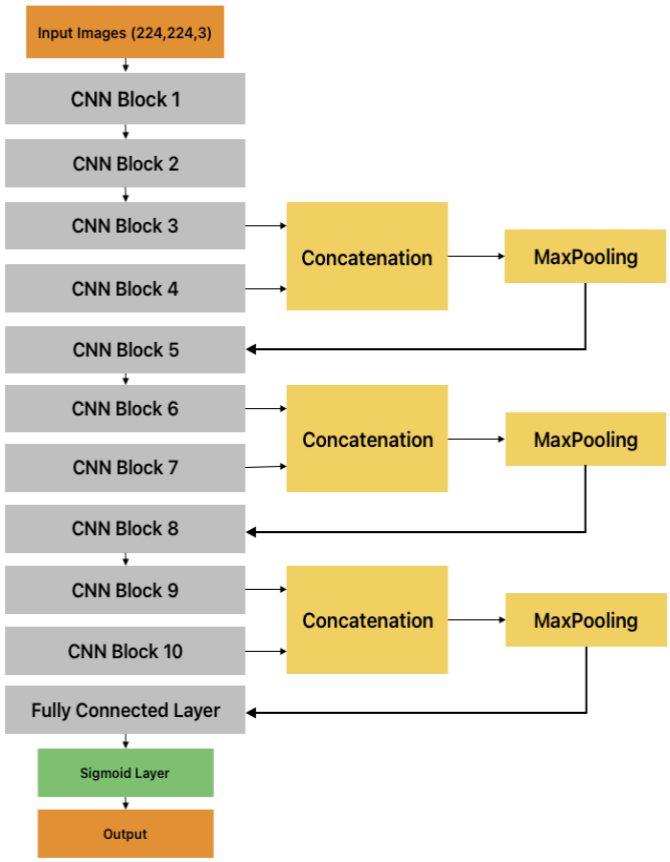


Figure 7 CNN_Blk Architecture with Concatenation and Pooling Blocks

By constructing CNN blocks, which include both the “shallow” and “deep” representations into one rich tensor through concatenation, the model can learn relationships at multiple spatial scales at once. After aggregating both the information passed from the “shallow” and “deep” layer, the max-pool can select the strongest activations while reducing the spatial dimensions. This also cuts the amount of data the next layer needs to process and focus on the most salient part of the image.

We use RGB video frames resized to (224, 224, 3) as inputs. After trying with different numbers of CNN Blocks, neuron number for each layer, adjusting dropout rate, adjusting l2 regularization weight decay, different optimizers, the model was compiled using the Adam optimizer and binary cross-entropy loss with label smoothing ($\varepsilon = 0.05$). To reduce overfitting, we used dropout (0.3 in conv layers, 0.5 in dense layers) and implemented EarlyStopping and ReduceLROnPlateau during training.

2) Data and Experimental Result

a) Basic Model

The training was conducted over a maximum of 50 epochs with early stopping enabled. The model stopped at epoch 26, restoring the best weights from epoch 18. Final test accuracy is about 88.11%, validation accuracy is about 73.93%. The training session's progression over time is stated in Figure 8, the graph for training progress over epochs is stated in Figure 9:

```
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 690ms/step - accuracy: 0.8704 - loss: 0.3916 - val_accuracy: 0.7909 - val_loss: 0.4829 - learning_rate: 2.5000e-04
Epoch 19/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 110s 695ms/step - accuracy: 0.8630 - loss: 0.3916 - val_accuracy: 0.7452 - val_loss: 0.5606 - learning_rate: 2.5000e-04
Epoch 20/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 681ms/step - accuracy: 0.8624 - loss: 0.3964 - val_accuracy: 0.7393 - val_loss: 0.7032 - learning_rate: 2.5000e-04
Epoch 21/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 0s 660ms/step - accuracy: 0.8593 - loss: 0.3928
Epoch 21: ReduceLROnPlateau reducing learning rate to 0.0001250000059371814.
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 686ms/step - accuracy: 0.8593 - loss: 0.3928 - val_accuracy: 0.7718 - val_loss: 0.4920 - learning_rate: 2.5000e-04
Epoch 22/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 687ms/step - accuracy: 0.8751 - loss: 0.3784 - val_accuracy: 0.7405 - val_loss: 0.5615 - learning_rate: 1.2500e-04
Epoch 23/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 107s 675ms/step - accuracy: 0.8803 - loss: 0.3718 - val_accuracy: 0.7393 - val_loss: 0.6061 - learning_rate: 1.2500e-04
Epoch 24/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 0s 655ms/step - accuracy: 0.8737 - loss: 0.3739
Epoch 24: ReduceLROnPlateau reducing learning rate to 6.25000029685907e-05.
158/158 ━━━━━━━━━━━━━━━━━━━━ 108s 681ms/step - accuracy: 0.8737 - loss: 0.3739 - val_accuracy: 0.7456 - val_loss: 0.5664 - learning_rate: 1.2500e-04
Epoch 25/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 109s 691ms/step - accuracy: 0.8768 - loss: 0.3719 - val_accuracy: 0.7567 - val_loss: 0.5489 - learning_rate: 6.2500e-05
Epoch 26/50
158/158 ━━━━━━━━━━━━━━━━━━━━ 110s 692ms/step - accuracy: 0.8811 - loss: 0.3681 - val_accuracy: 0.7393 - val_loss: 0.6528 - learning_rate: 6.2500e-05
Epoch 26: early stopping
Restoring model weights from the end of the best epoch: 18.
```

Figure 8 Result of CNN_BLK model

Figure 9 Training progress over epochs for CNN_BLK model

b) Performance Evaluation

As seen in the graph above, training accuracy increases steadily with decreasing loss, showing the model is learning well and training successfully. But the validation accuracy peaks early and slightly decline, indicating early signs of overfitting. The gap

between training and validation metrics widens after epoch 15, where the divergence shows signs of overfitting. The model reached its best performance around epoch 8-10. The model can learn useful features for detecting sports from single frames for the first 10 epochs, but the divergent validation results indicate the model doesn't generalize well to new images. As for the learning efficiency, we can see that as the number of epochs increases, the accuracy climbs up steadily, indicating well learning efficiency.

3) Conclusion

The CNN_BLK model, which uses hierarchical concatenation and pooling, shows overfitting in classifying image frames into "sports" and "non-sports" categories. It achieves a strong training accuracy of 88% but a lower validation accuracy of 74%. While the divergence persists, the model proves training efficiency, but the concatenation fails to capture the most salient features. However, the model shows capability for relatively higher learning efficiency.

6. Model Performances Comparison

The primary concern for our project which aims at sports highlight extraction is recall: failing to detect an actual sports frame would result in missing key content. As the table shown in Figure 10, the CNN_BLK Model offers high precision value, but its low recall limits its utility for this specific task. The CNN + Transformer Model provides a better balance for precision and recall, ensuring a better precision when capturing true sports actions.

	Accuracy	Precision	Recall	F1-score
Base CNN+Transformer	0.843651	0.742160	0.633929	0.683788
Advanced Concat-CNN	0.790873	0.887701	0.247024	0.386496

Figure 10 Comparison of the metrics for the CNN + Transformer and CNN_BLK models

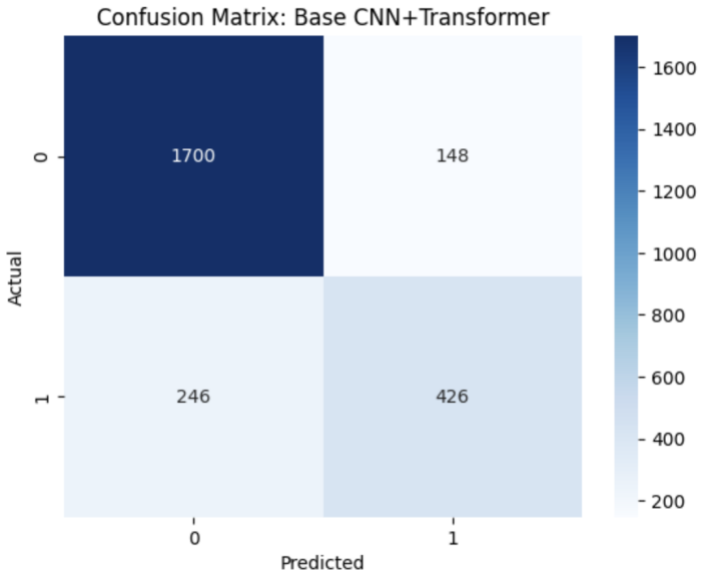


Figure 11 The Confusion Matrix of CNN + Transformer Model

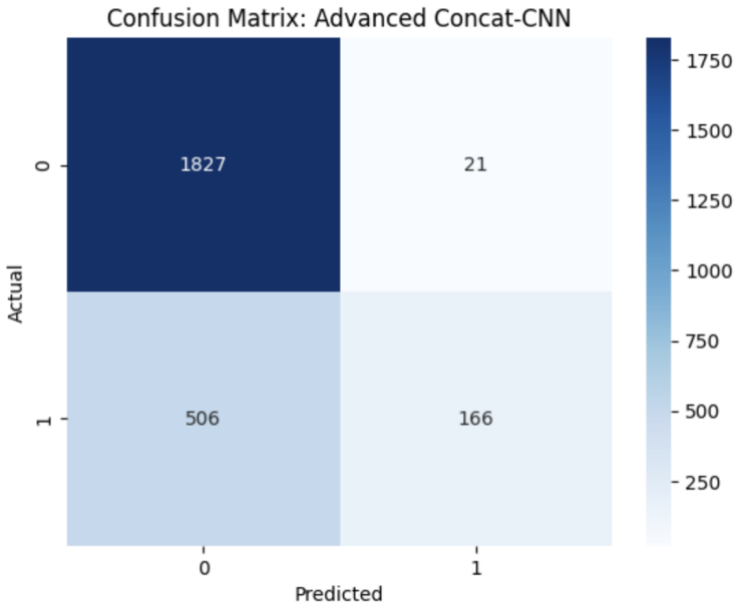


Figure 12 The Confusion Matrix of CNN_Blk Model

7. Conclusion

If choosing a single model, the CNN+Transformer is the better fit for sports highlight detection. But we can also use the hybrid approach which could offer the best of both worlds, using the CNN+Transformer for broad detection since its high recall, then use the CNN_Blk as a second-stage filter for high precision output as this model shows higher learning efficiency, aligning well with the project's goal.

8. Business impact

This system enhances the productivity and efficiency by automating the highlight extraction process, effectively reducing manual editing time and enabling near real-time content delivery for social media. Plus, by integrating CNN + Transformer and CNN_BLK model, it can achieve higher coverage for true sports highlight moments, enhancing viewer engagement and even improving time-to-market for promotional clips.

For the next step, we would address current limitations for precision-recall tradeoffs and class imbalance by combining the high-recall and high-precision models and apply oversampling or augmentation for the minority class.

References

1. Abdellatef, E., Al-Makhlasawy, R. M., & Shalaby, W. A. (2025). Detection of human activities using multi-layer convolutional neural network. Scientific Reports, 15, 7004. https://doi.org/10.1038/s41598-025-90307-6
2. Yuan, F., Zhang, Z., & Fang, Z. (2023). An effective CNN and Transformer complementary network for medical image segmentation. Pattern Recognition, 136, 109228. https://doi.org/10.1016/j.patcog.2022.109228