Automatic Highlights Generation for Cricket Matches

Alina Sarwar

Why cricket?

- Second most loved game across the globe
- Manual generation of highlights is a cumbersome task.
- Longer duration than most other sports (~3-3.5 hours for T20)
- Dataset: 12 T20 match videos and corresponding highlights from official broadcaster

Proposed Methodology

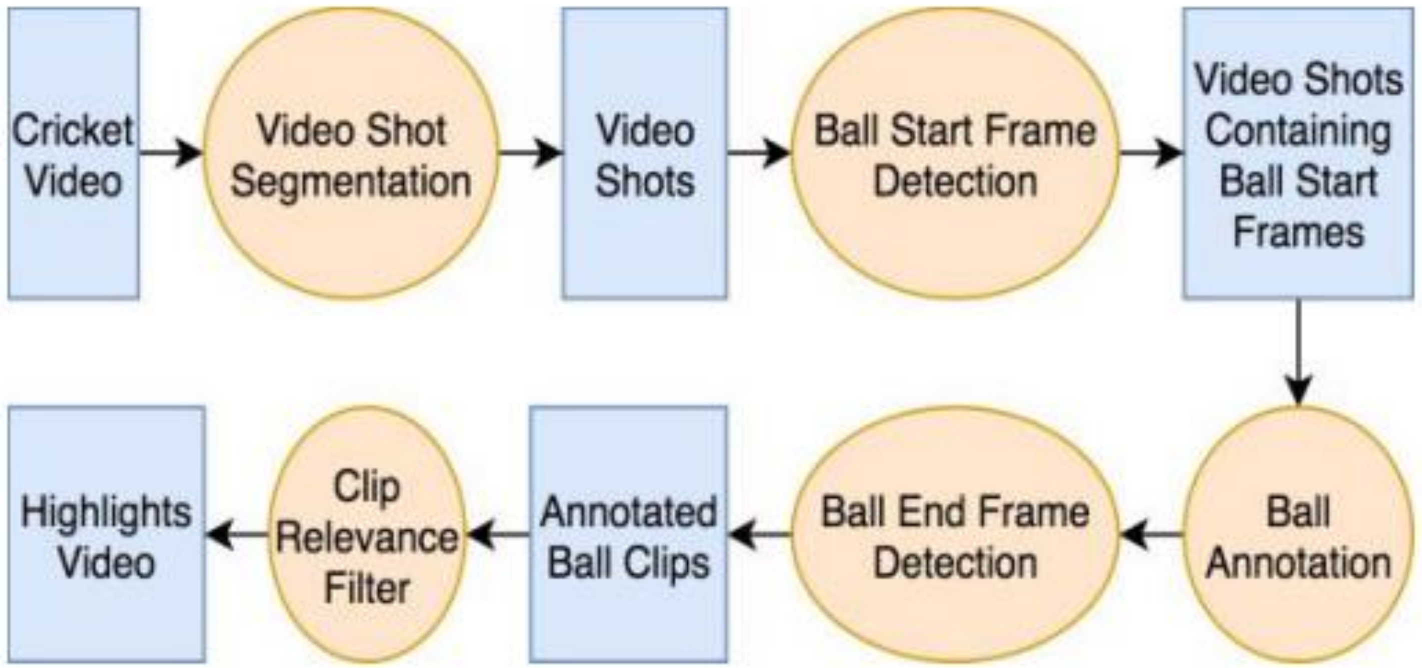

Figure 1: Architecture of proposed approach

Agarwal et al, "Automatic Annotation of Events and Highlights Generation of Cricket Match Videos", in International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2019.

Video Shot Segmentation

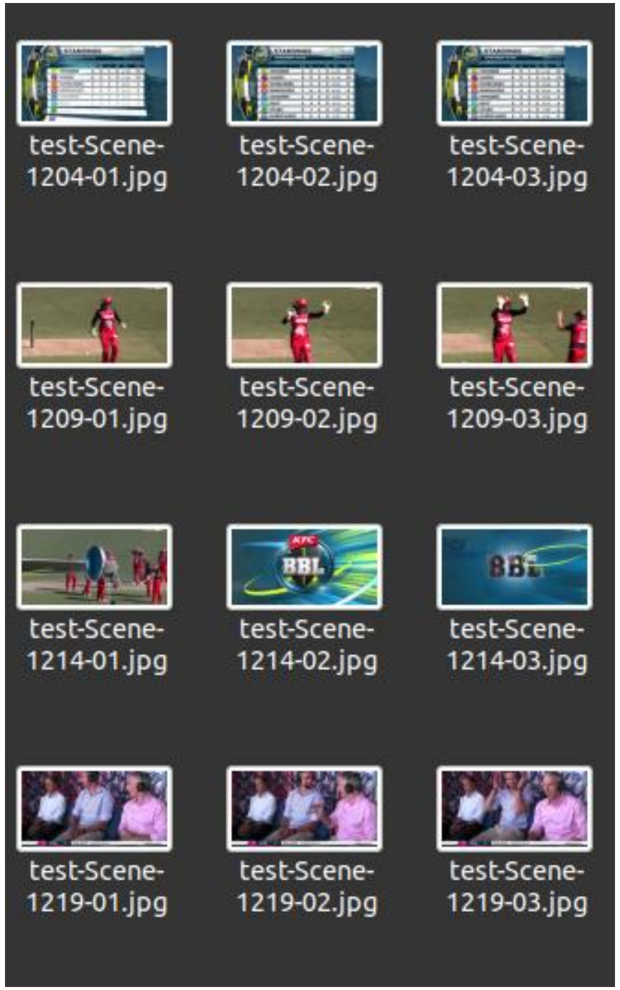

Figure 2: Sample scenes from a match

- Divide complete match video into separate scenes – PySceneDetect
- Dissimilarity between consecutive frames measured using Hue, Saturation, Value
- Each scene is represented by 3 key frames – start, middle, end
- Results saved to csv with start/end timecodes
- Helps cut down processing cost

Ball Start Frame Detection

Figure 3: Generating potential positive and negative training samples for ball start frame detection

- Train a CNN (binary classifier) for detecting ball start frames
- Prepare potential +/- training samples using image dilations
- If frame difference < threshold, collect as + sample
- Manually review generated samples

Figure 4: Examples of positive samples

~1500 positive samples increased to ~4000 after data augmentation

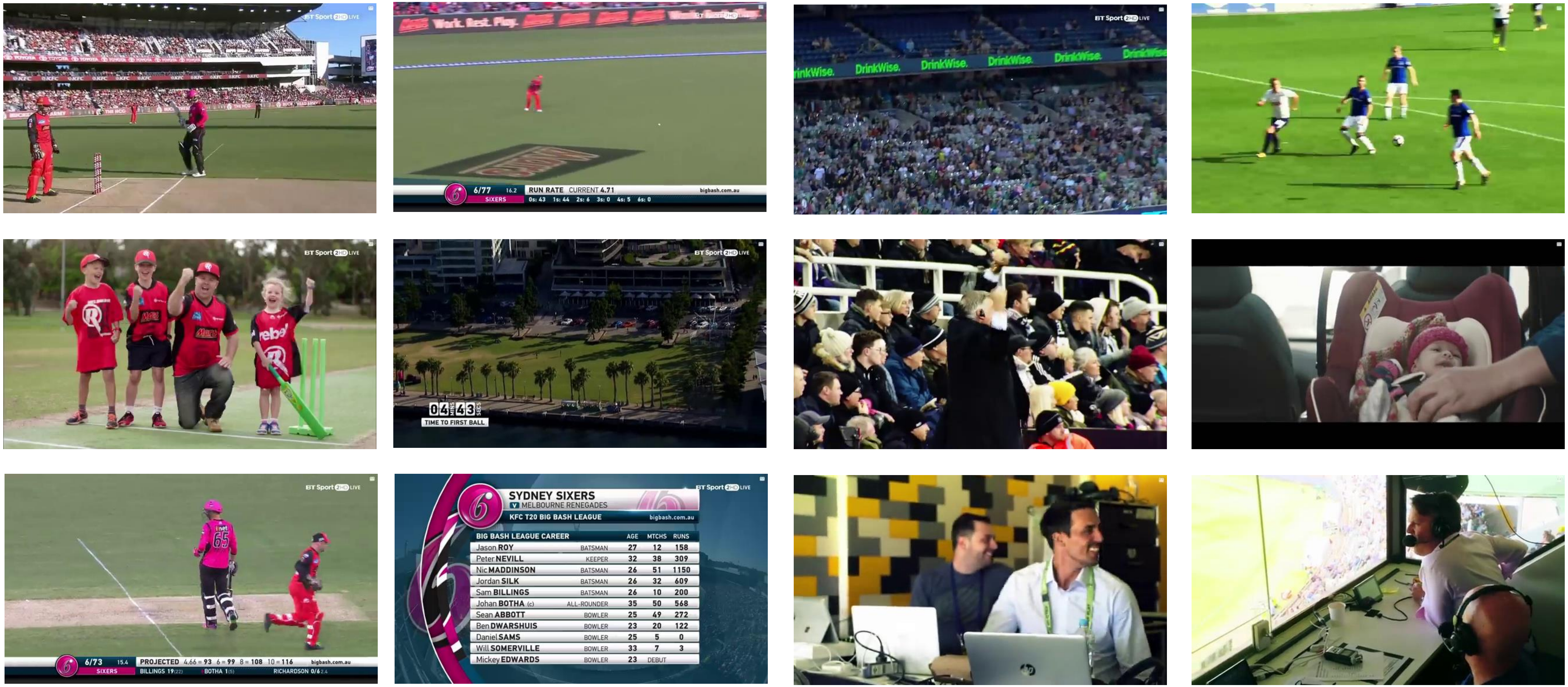

Figure 5: Examples of negative samples

~5000 used in training after x10 downsampling

Ball Start CNN Architecture

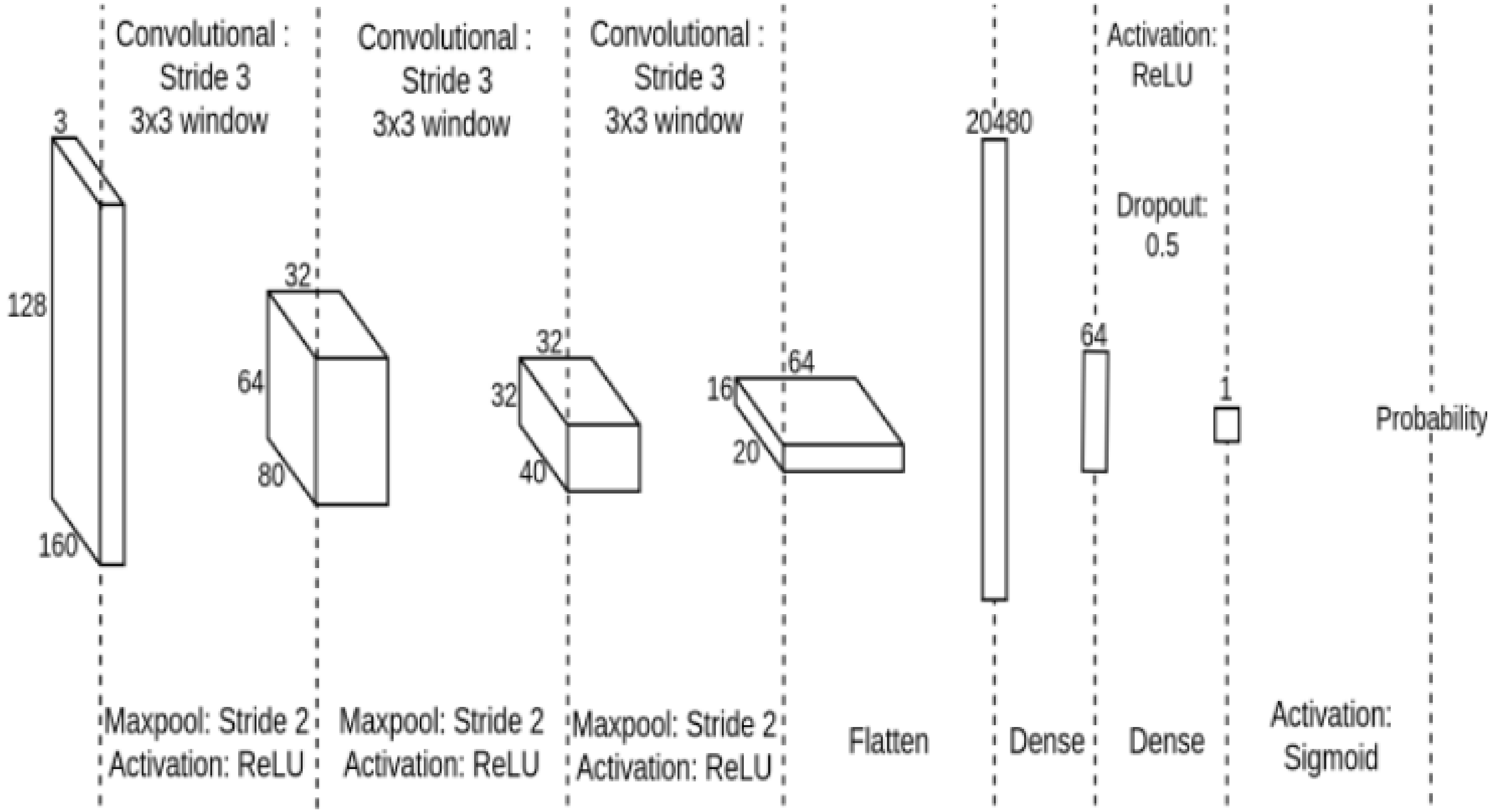

Figure 6: Ball start CNN architecture

Model Training

- RGB frames, downsized to dimensions 160x128
- Train-validation split: 80:20 – with stratified sampling to maintain equal class proportions in each set
- ~450 test images
- Loss Function: Binary Cross Entropy
- Learning Rate: 0.0001

Model Training

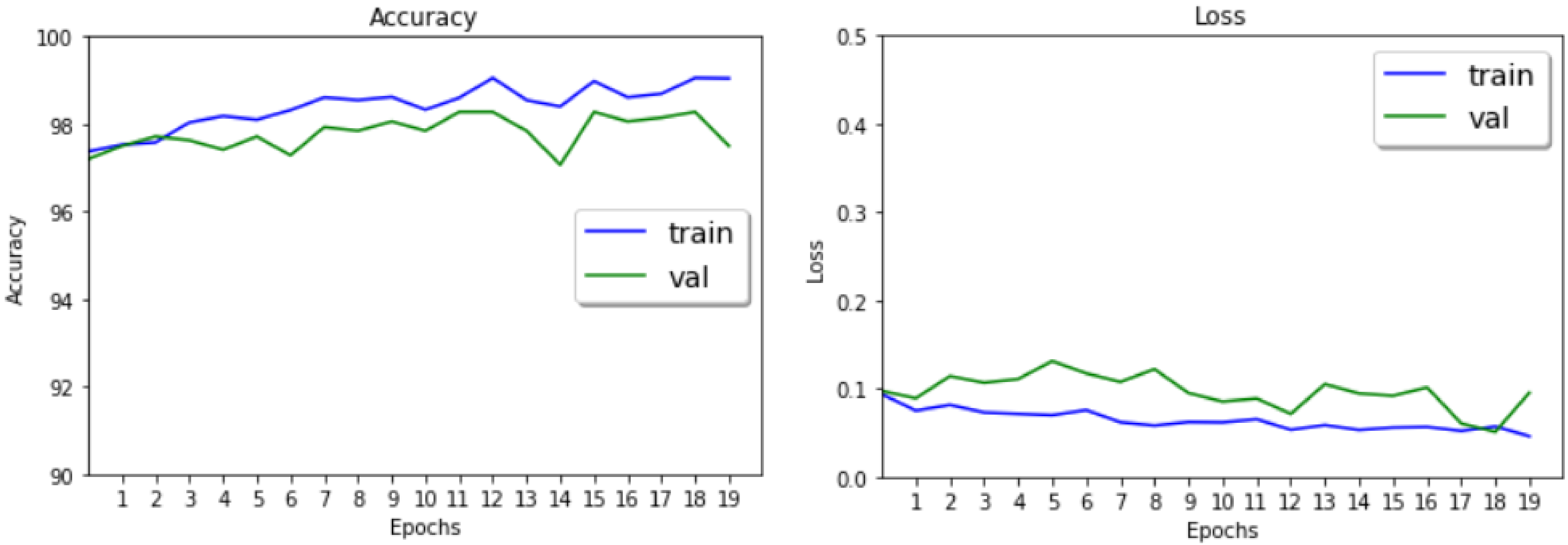

Figure 7: Training statistics

Model Results

- Model weights saved at the best validation accuracy
- Test Accuracy: 98.25%
- 8/458 incorrect predictions - mostly labelling errors

Figure 8: Visualizing wrong predictions

Further steps..

- Find model predictions on the bulk of negative samples originally collected and incorporate any false positives in the training set
- Switch color channels and convert images to gray scale as part of data augmentation
- Rerun an iteration of training

Project milestones

- **Phase I progress update**
  - Dataset acquisition
  - Generation of scene clips
  - Generation of training samples for CNN
- **End of Phase I**
  - Complete training of CNN
- **Phase II progress update**
  - Complete OCR implementation for detecting ball end frame
  - Ball clip annotations
- **End of Phase II:**
  - Final highlights generation
  - Debugging + code clean ups
  - Evaluation of results and reporting

- Player specific highlights
- Customized highlights generation based on personal preferences

Questions?