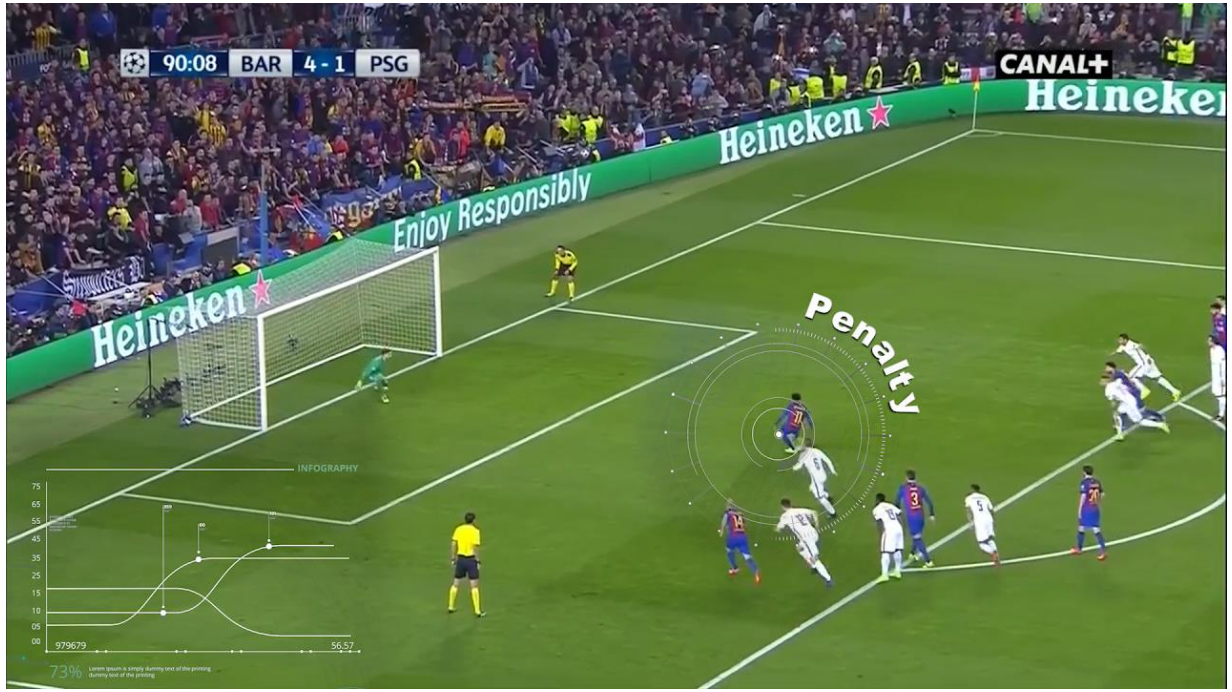
Master in
Computer Vision
Barcelona

Module:	C6 – Final Presentation
Project:	**Ball Action Spotting**
Coordinator:	A. Clapés
Team 8:	S. van der Linde, G. Grigoryan, V. Heuer, P. Zetterberg

Table of Contents

Introduction and Motivation: Ball Action Spotting in Football Broadcasts.



[1]

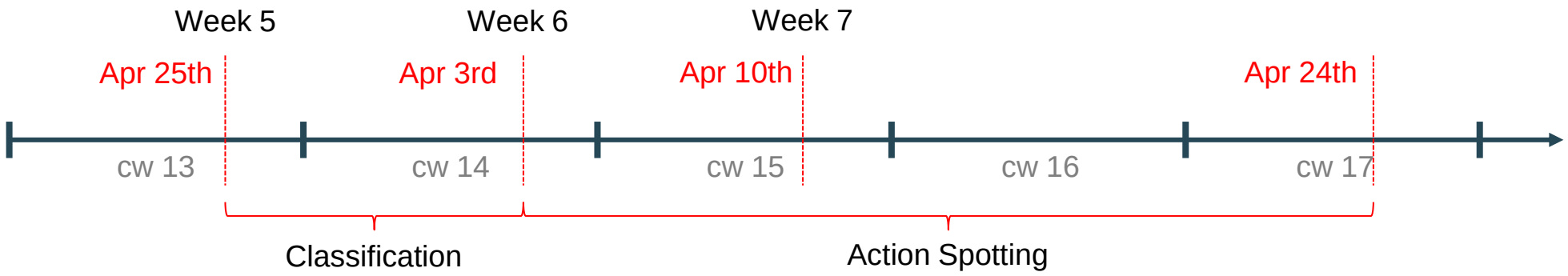
- Growing interest in automativ vide understanding in sports.
- Fine-grained temporal localization of ball-related actions in football is an open research problem.
- Real-world applications: game analysis, highlight generation, player/team analytics.

[1] https://www.soccer-net.org/challenges/2025

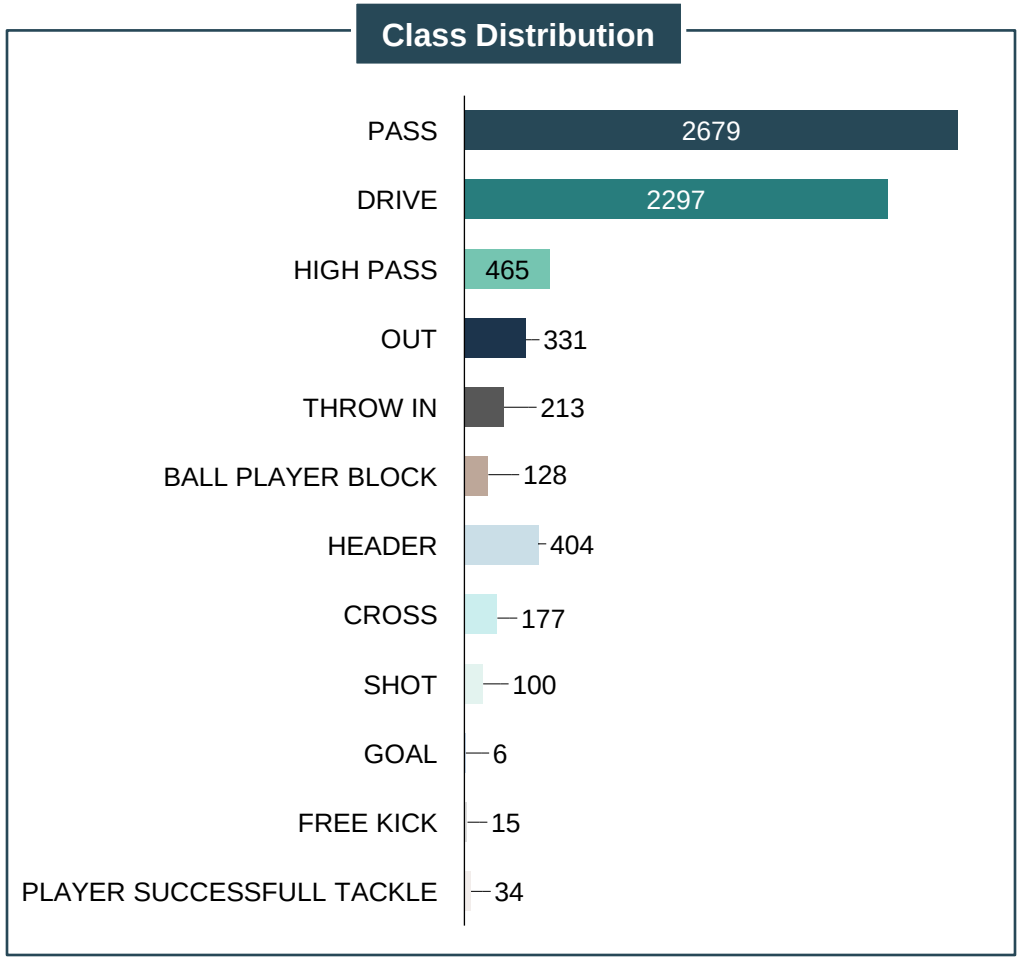
Introduction and Motivation 2: Motivation and Challenge Setup – Our project gradually increased in complexity – from basic classification to a realistic spotting scenario demanding fine temporal resolution.

- The task: Predict start times of 12 ball-related actions with 1-second accuracy.
- Week 1: Simplified classification task (no temporal component).
- Weeks 2–3: Full spotting task requiring high temporal precision.
- Custom challenge introduced by our professor as an introduction to SoccerNet.

Week	Task
Week 5	Classification
Week 6	Spotting (model v1)
Week 7	Spotting (model v2)



Data Set Overview



Key Stat	Value
Number of matches (train)	4
Number of matches (val)	1
Number of matches (test)	2
Duration per match	90 minutes
FPS with stride of 2	12.5
Action labels	12
Labels:	pass, drive, header, high pass, out, throw in, ball player block, shot, goal, free kick, player successful tackle
Annotations	Action labels with timestamps
Granularity	1 s

drive

Approach for the first task in week 5.

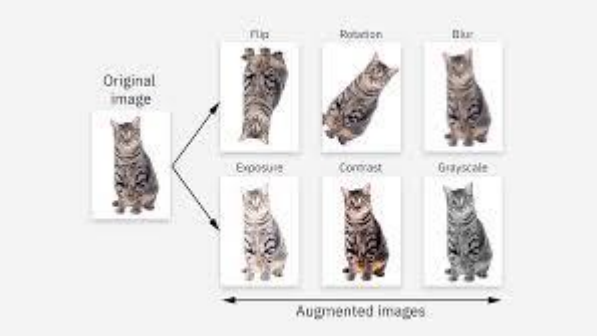
Key Facts

- **Task:** Frame-level ball action classification (single-label per clip)
- **Input:** Clip of 50 frames downsampled from 25 fps to 12.5
- **Model:** Simple image classification models
- **Output:** Predicted action class per clip

- Initial baseline for understanding model behavior on individual frames.
- Used as warm-up before addressing the temporal spotting challenge.

Ideas for Improving

- **Data augmentation:** We aim for improved generalization by searching for optimal data augmentation.
- **Testing different models:** The baseline uses RegNet, we aim for better performacne by using different and larger models.
- **Implementing YOLO:** By cropping the irrelevant parts of the frame we want to increase the quality of the learned features.



[1] https://www.ibm.com/think/topics/data-augmentation
[2] https://latenode.com/de/blog/what-is-resnet-50-and-how-can-it-transform-your-business-automation
[3] https://medium.com/@beyzaakyildiz/what-is-yolov8-how-to-use-it-b3807d13c5ce

Approach for the second task in week 6 and 7.

Key Facts

- **Task:** Temporal spotting of actions in full-length matches
- **Input:** Clip of 50 frames downsampled from 25 fps to 12.5
- **Model:** Simple image classification models + temporal models
- **Output:** Predicted action class and timestamp

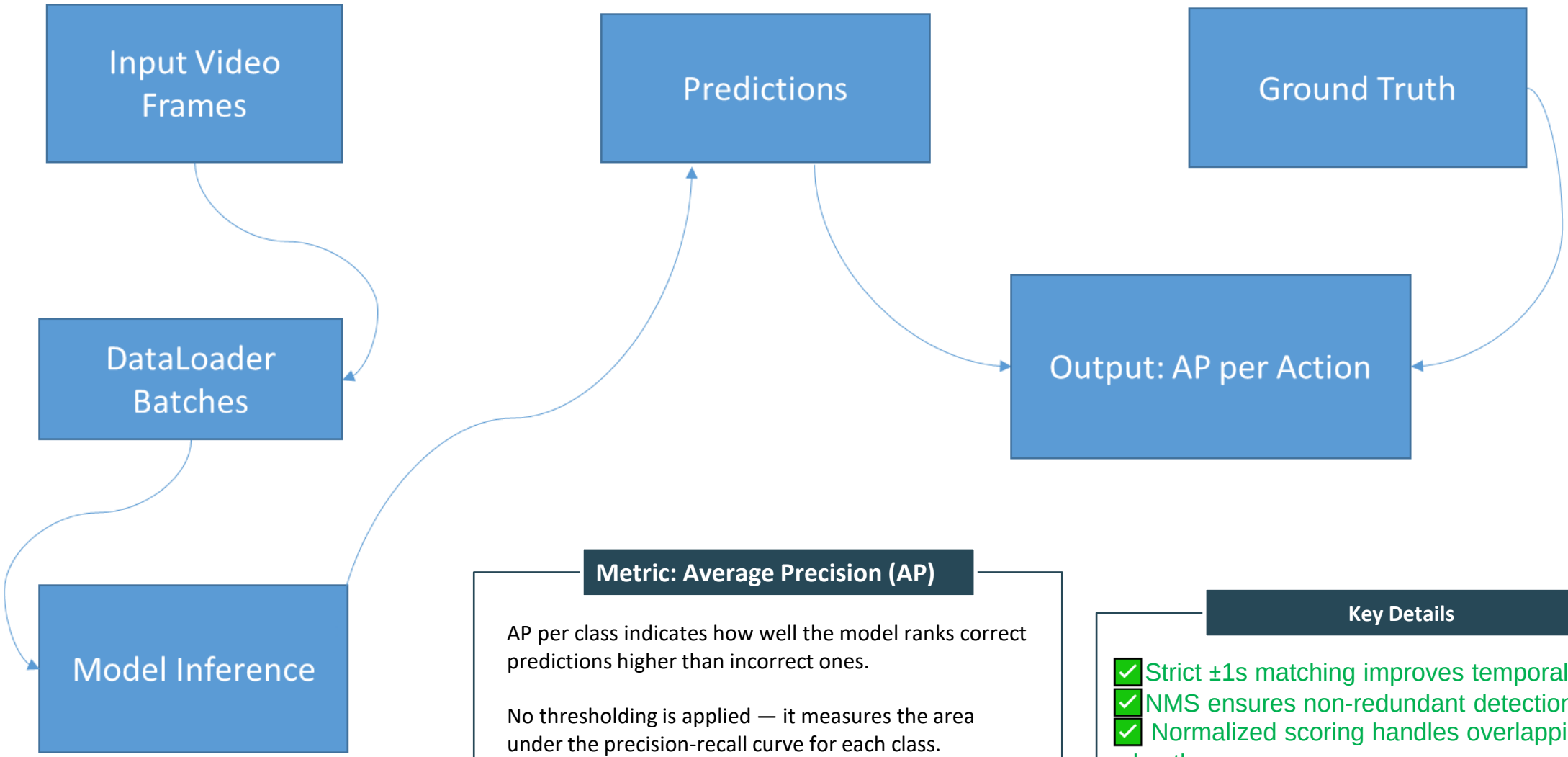
- Integrated temporal modeling (sequence-aware)
- Moved from classification to localization

Ideas for Improving

- **BiLSTM:** BiLSTM is a powerful bi-directional temporal model that allows for temporal modelling.
- **Transformers:** We used a transformer as temporal model with the idea of the self attention being the temporal component.
- **Implementing X3D:** X3D showed promising results in the first week, therefore we implement it in our workflow to enhance the classification part.
- **Adding methods from T-DEED:** We looked into the implementation of T-DEED by A. Xarles and copied some methods as in augmentation strategies and adding a displacement head.
- **Heavy fine Tuning:** We layed emphasis on finetuning the most performant model from week 2 to further increase its performance.
- **Adding methods from TriDet:** After reviewing TriDet from D. Shi we implemented the idea of an offset.

➢ Test Set Up?

Test Set Up 1



Metric: Average Precision (AP)

AP per class indicates how well the model ranks correct predictions higher than incorrect ones.

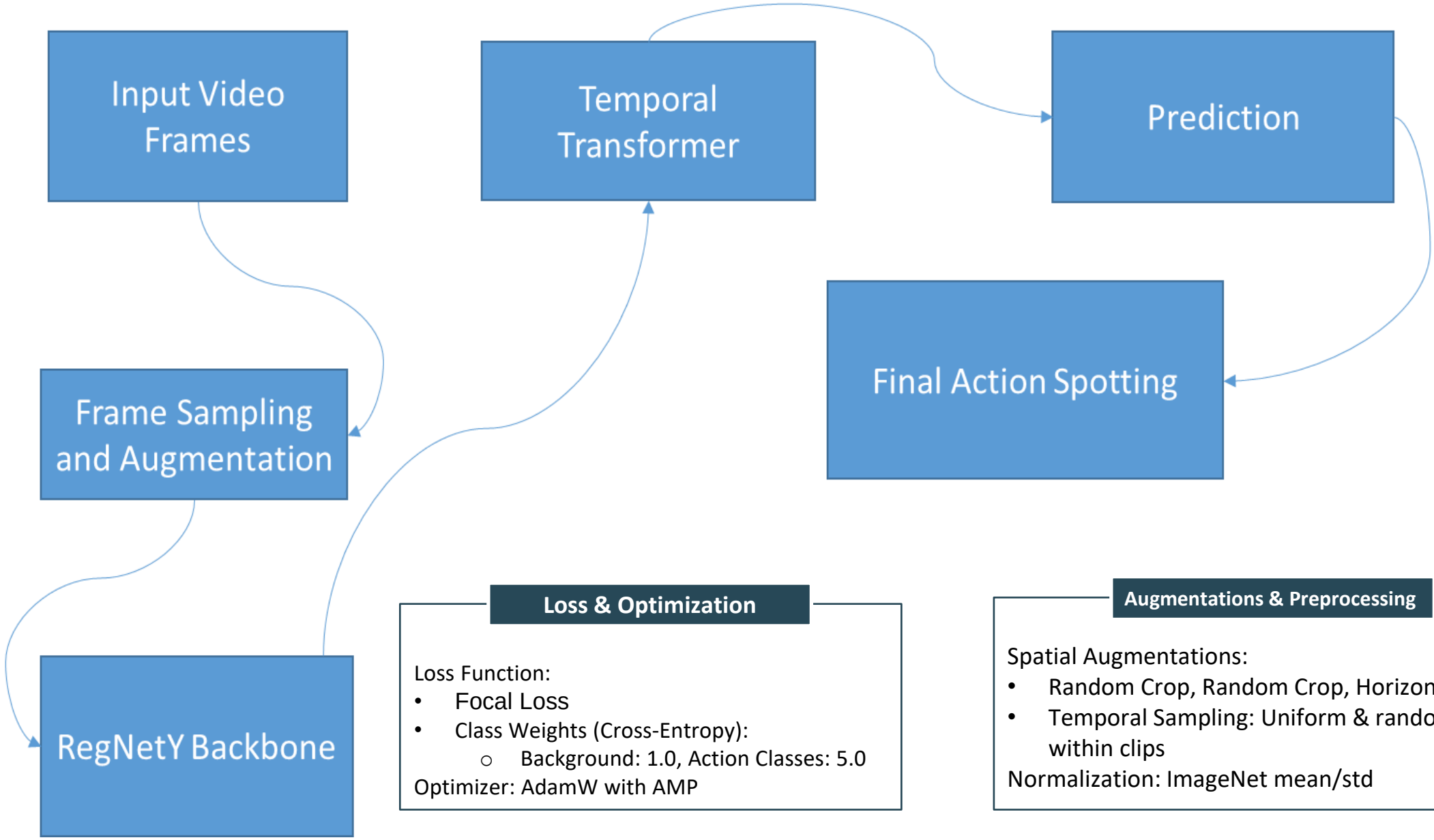
No thresholding is applied — it measures the area under the precision-recall curve for each class.

Returns a separate AP value for each action class.

Key Details

✅ Strict ±1s matching improves temporal accuracy
✅ NMS ensures non-redundant detections
✅ Normalized scoring handles overlapping clips robustly

Test Set Up 2



Loss & Optimization

Loss Function:
- Focal Loss
- Class Weights (Cross-Entropy):
  - Background: 1.0, Action Classes: 5.0

Optimizer: AdamW with AMP

Augmentations & Preprocessing

Spatial Augmentations:
- Random Crop, Random Crop, Horizontal Flip
- Temporal Sampling: Uniform & random within clips

Normalization: ImageNet mean/std

Results Week 5 (Quantitative)

The imbalance in the dataset clearly impacted performance, making it harder for the model to accurately detect rare but important actions. This highlights the need for more balanced data or improved sampling strategies to ensure better sampling strategies. To address this and boost performance, we tested several pretrained models as feature extractors: **ResNet**, **EfficientNet**, and **ConvNext**. All three outperformed our baseline model. Among them, EfficientNet delivered the best results, despite being smaller than ConvNext.

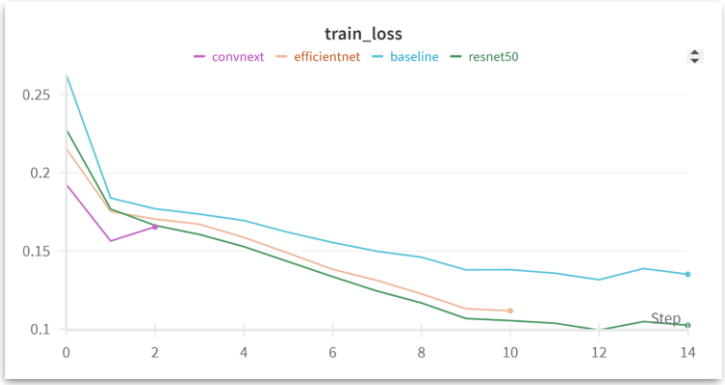
	Resnet 50	***EfficientNet 0***	***ConvNext Tiny***
mAP_12	29.48	31.31	29.44
mAP_10	35.33	35.57	34.48
params	23,532,620	4,022,920	27,829,356
gflops	369.39	35.95	386.54

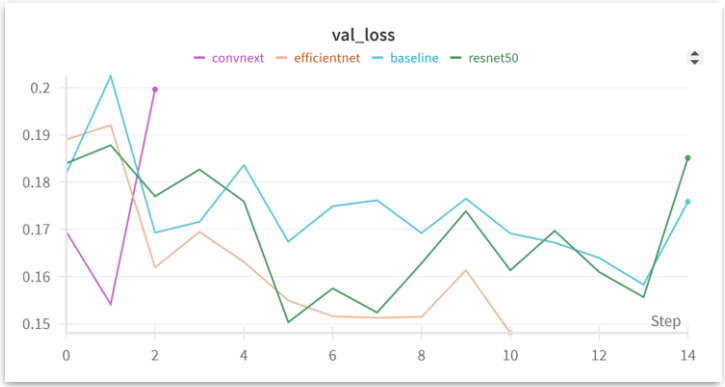
AP for every Class EfficientNet

Class	Average Precision
PASS	86.35
DRIVE	79.53
HEADER	26.08
HIGH PASS	48.59
OUT	16.73
CROSS	29.03
THROW IN	29.73
SHOT	24.14
BALL PLAYER BLOCK	13.36
PLAYER SUCCESSFUL TACKLE	2.12
FREE KICK	16.24
GOAL	3.84

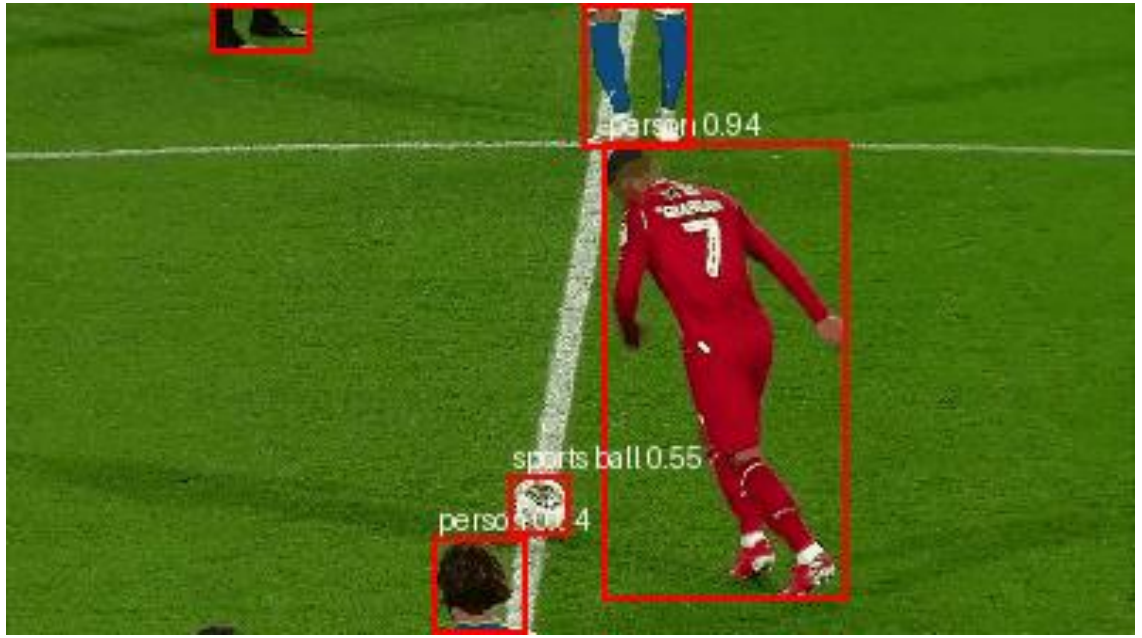
The model performs best on frequently occurring actions like **PASS (86.35)** and **DRIVE (79.53)**, but struggles with less common/more complex actions like **PLAYER SUCCESSFUL TACKLE (2.12)** and **GOAL (3.84)**

Results Week 5 (Qualitative)





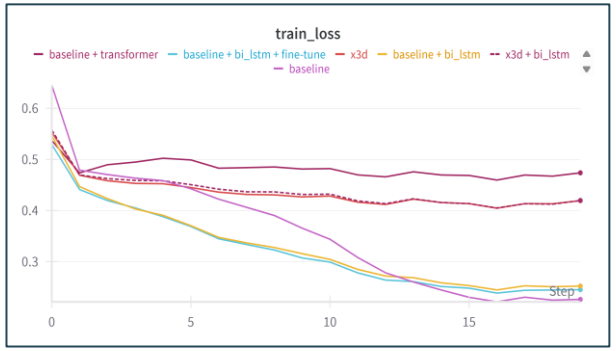
While the training loss of all models is relatively similar, EfficientNet's validation loss decreases **more steadily** than that of the other models, which show a lot of sudden peaks and decreases.

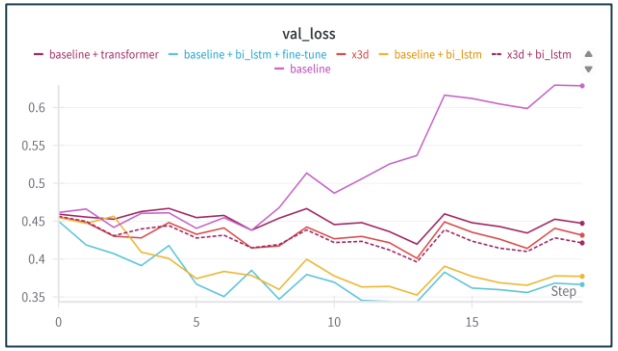


	Baseline	Baseline + aug	Resnet 50	EfficientNet 0	ConvNext Tiny	Yolo Tracking
mAP_12	27.79	33.73	29.48	31.31	29.44	12.73
mAP_10	32.74	30.32	35.33	35.57	34.48	15.22

Results week 5 & 6

Train loss vs Validation loss

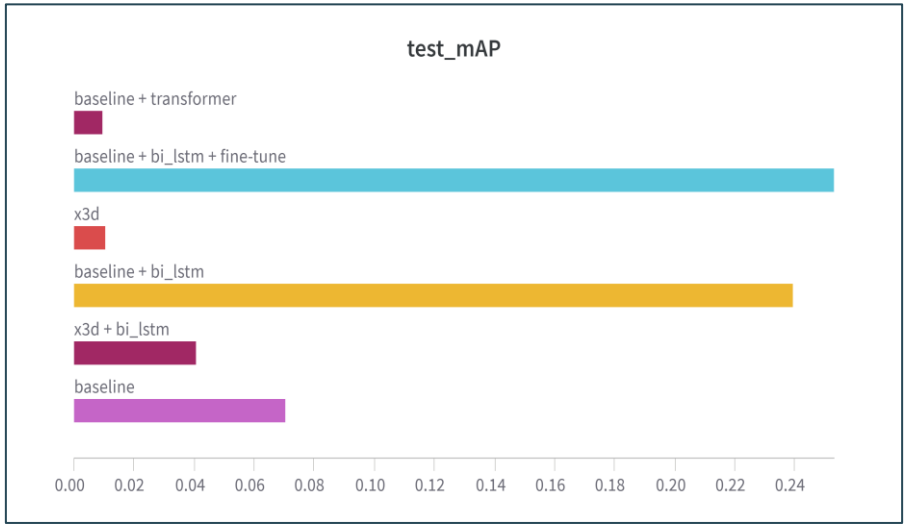




Key Observations:

1. **baseline + bi_lstm + fine-tune (light blue):** achieves the lowest validation loss, consistently improving through training while avoiding overfitting. Fine-tuning by increasing the hidden size from 256 ⟶ 512 gave the model more capacity to learn temporal patterns, while still maintaining strong training loss reduction.
2. **baseline + bi_lstm (orange):** follows closely with slightly higher validation loss, but stable training. Bi-LSTM significantly improves temporal understanding, though the smaller hidden size limits its capacity compared to the fine-tuned version.
3. **x3d + bi_lstm (dark red):** shows modest train loss reduction but val_loss plateaus—likely due to freezing most X3D layers, which limits feature adaptability and causes underfitting or restricted generalization.
4. **baseline (purple):** achieves low train loss but overfits quickly, with val_loss diverging after epoch 10. Without temporal modeling or regularization, it fails to generalize.
5. **baseline + transformer (maroon):** struggles with both high train and validation loss. Transformer underperforms due to poor positional encoding.
6. **x3d (dark orange):** shows high train and val loss, indicating failure to learn meaningful patterns. Without temporal modeling and with frozen weights, it lacks adaptability for SoccerNet.

MAP scores across experiments



mAP Results:

- **Best performance**: baseline + lstm + fine tune
⟶ Achieves the highest mAP (≈ 0.25)
- **Baseline + lstm**: performs well, close to fine-tuned version, with a mAP just bellow that (≈ 0.24)
- **Baseline**: relatively low baseline mAP (≈ 0.06), benefited significantly from temporal modeling
- **X3D**: performs poorly as a standalone, has the worst mAP
- **X3D + lstm**: slight improvement over x3d alone, but still lags behind the baseline + lstm combination

Results week 5 & 6

Good example

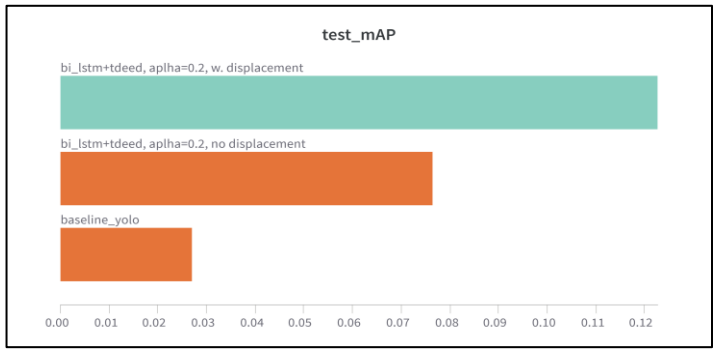
When comparing the results to our initial expectations, it became clear that the anticipated benefits of using X3D did not materialize. Despite testing various freezing strategies and incorporating LSTM, there were no notable improvements, so we decided not to use X3D in the upcoming challenge. Similarly, the use of transformers led to reduced performance. The baseline model outperformed X3D, and adding a BiLSTM to it significantly improved results. After fine-tuning this setup, we achieved our best results of the week: 0.25 mAP@12 and 0.31 mAP@10.

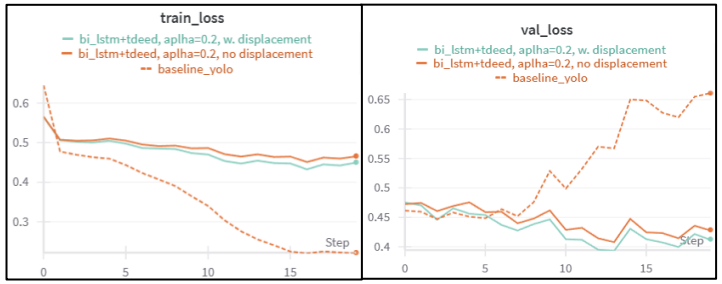
mAP														
Model/Class	Pass	Drive	Header	High Pass	Out	Cross	Throw In	Shot	Ball Player Block	Player Successful Tackle	Free Kick	Goal	mAP12	mAP10
baseline lstm + finetune	**0.65**	**0.59**	**0.36**	**0.58**	**0.06**	**0.29**	**0.19**	**0.15**	**0.02**	**0**	**0.14**	**0**	**0.25**	**0.31**
baseline lstm	0.65	0.55	0.32	0.53	0.03	0.25	0.20	0.12	0.04	0	0.18	0	0.24	0.29
baseline	0.27	0.17	0.15	0.01	0.02	0.02	0.08	0.02	0.01	0	0	0	0.07	0.08
x3d lstm	0.08	0.06	0.02	0.11	0.10	0.02	0	0	0	0	0	0	0.04	0.04
baseline transformer	0.06	0.05	0	0	0	0	0	0	0	0	0	0	0.01	0.01
x3d	0.05	0.04	0.01	0.01	0	0.01	0.01	0	0	0	0	0	0.01	0.01

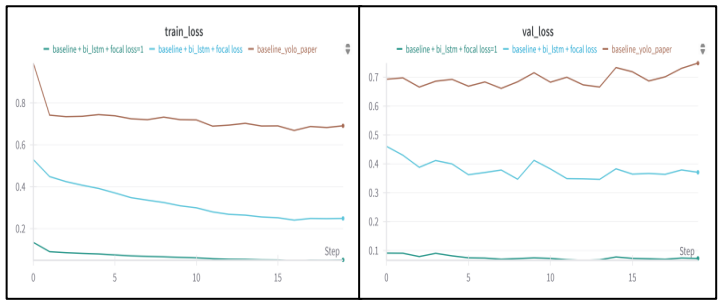
Results week 7

We evaluated six models to try and improve our mAP scores:

- **Baseline + bi_lstm + focal_loss=1**: achieves the highest mAP (mAP@12=31.34), showing that adding a Bi-LSTM and focal loss (γ=1) improves temporal modeling and handles class imbalance effectively without being overly selective.
- **Baseline + bi_lstm + focal_loss=2**: second best mAP, but a higher focal loss value may overly down-weight easy examples, likely worsening generalization in this case
- **Baseline + yolo_paper**: lowest performance in its group. YOLO is designed for spatial object detection, and without temporal modeling, it struggles with SoccerNet's event detection task
- **Bi_lstm + tdeed + displacement**: best performance in the second group. Incorporating displacement features with temporal difference embeddings helps capture motion patterns crucial for event understanding.
- **Bi_lstm + tdeed + no displacement**: performs worse than with displacement, suggesting that motion cues are valuable for detecting subtle event changes in SoccerNet.
- **Baseline + yolo**: performs the worst overall. Similar to the “yolo_paper” version, it lacks temporal context and struggles with event-level predictions in a video setting.







Best achieved mAP

Class	Average Precision
PASS	67.09
DRIVE	59.46
HEADER	38.26
HIGH PASS	59.39
OUT	12.92
CROSS	34.86
THROW IN	24.98
SHOT	23.38
BALL PLAYER BLOCK	5.7
PLAYER SUCCESSFUL TACKLE	0
FREE KICK	50
GOAL	0

	Average Precision
Mean	31.34

Team 8 Summary: Using the baseline model with Bi-LSTM and fine tuning yielded the best results.

Model and Parameter Tuning

Models tested:

RegNet
ResNet
X3D
EfficientNet
ConvNext

Temporal Models:

Bi-LSTM
Transformer Based
TDEED Inspired
TriDed Inspired

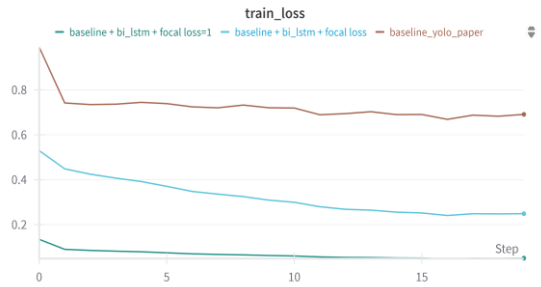
Combination	mAP 12	mAP 10
RegNet+Bi-LSTM	31.34	32.60
RegNet+TDEED	12.27	13.27
X3D+Bi-LSTM	05.38	05.99
RegNET+TriDed	02.57	02.72

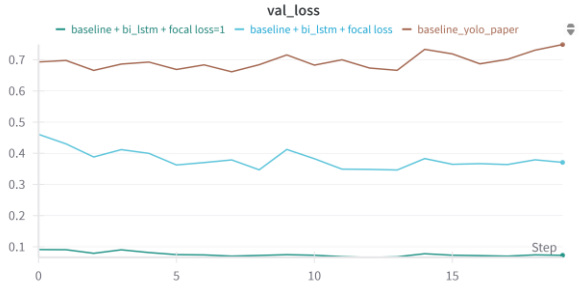
Method

We tested many different combinations of backbones, temporal models, displacement heads and data mixing.

We always trained for 20 epochs with LR-scheduler and the same LR.

Results





- Both train and validation loss leveld after only a few epochs. Indicating that early stopping could have decreased training times.

Predictions

Good prediction

Bad prediction

Discussion of the results of week 5

Discussion

- EfficientNet delivered the best results (mAP_12: 31.31, mAP_10: 35.57), outperforming larger models like ResNet and ConvNext despite having fewer parameters — proving efficiency matters more than size.
- Severe class imbalance affected performance, with common actions like “Pass” (86.35 AP) and “Drive” (79.53 AP) dominating, while rare actions like “Goal” (3.84 AP) and “Tackle” (2.12 AP) were poorly detected.
- Data augmentation had a major impact, boosting baseline mAP from 27.79 to 33.73. Simple techniques like flipping and color jitter improved generalization without overfitting.
- EfficientNet showed smooth, stable validation loss, in contrast to the instability seen in other models — indicating better learning and robustness.
- YOLO-based tracking underperformed (mAP_12: 12.73), likely due to unoptimized input pipelines and the lack of temporal context, though it may be useful with further refinement.
- **Key takeaway:** Architecture quality, balanced data, and thoughtful augmentation are more impactful than model size alone.

Discussion of the results of week 6

Discussion

- Bi-LSTM significantly improved performance over the baseline, confirming that temporal modeling is essential for spotting tasks
- Fine-tuning Bi-LSTM with a larger hidden size (512) further improved validation loss and generalization, leading to the highest mAP scores of the week (mAP@12: 0.25)
- X3D underperformed, even when combined with Bi-LSTM. Freezing most of its layers likely limited learning capacity and adaptability
- Transformers struggled due to ineffective positional encoding, resulting in high training and validation loss
- The baseline model quickly overfit, lacking temporal awareness and regularization
- **Key takeaway:** Bi-LSTM was the most reliable temporal model, and fine-tuning its parameters can extract more temporal context effectively.

Discussion of the results of week 7

Discussion

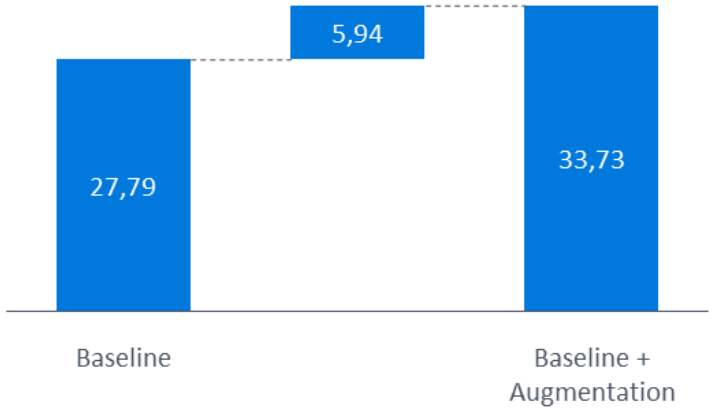
- Best performance came from Bi-LSTM + Focal Loss (γ=1), reaching the highest mAP_12 (31.34). This setup improved both temporal understanding and robustness to class imbalance.
- Higher focal loss (γ=2) slightly reduced performance, likely due to over-suppressing easy examples, which hurt generalization.
- Adding displacement features (from T-DEED) to Bi-LSTM improved results, showing that motion cues are valuable for detecting nuanced football actions.
- Without displacement, performance dropped, emphasizing the importance of capturing movement dynamics in spotting tasks.
- Overall: Temporal modeling + motion-aware features are key to success in video action spotting.

Team 8, 1: Performance of Baseline

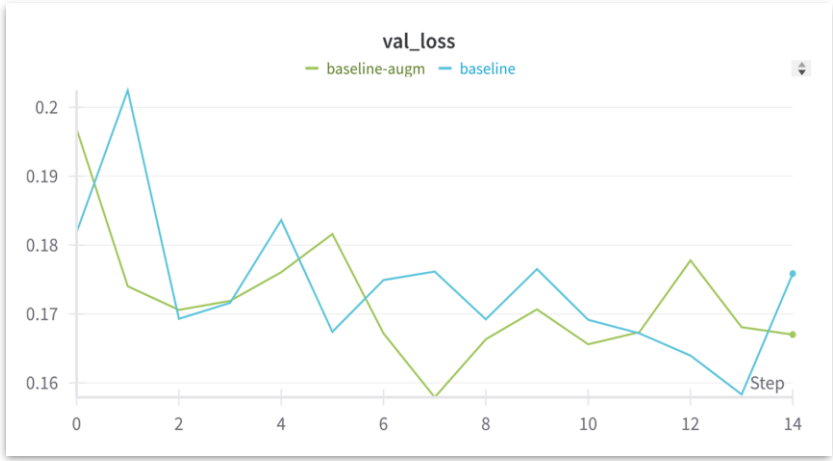
Augmentation

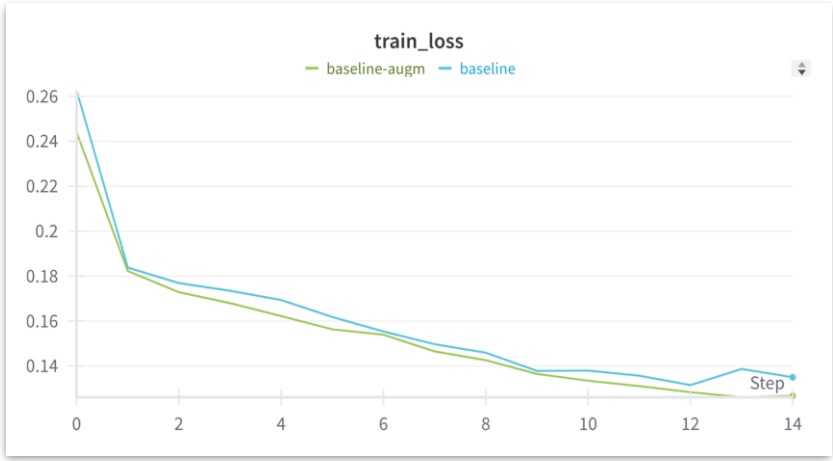
The model was trained with color jitter (hue, saturation, brightness, contrast), Gaussian blur, and horizontal flip applied randomly. Later, the augmentation was simplified to just color jitter and flipping. With learning rates tested between **1e-4 and 8e-4(baseline)**, the model performed better with simpler augmentation, likely because excessive transformations distorted important features.

Adding the augmentation and changing the lr improved the baseline mAP by almost **+6%**



Train and Validation Loss





Team 8, 2: Testing Different Pretrained Models

One idea to improve the model‘s performance was to test different pretrained models as feature extractors, we suspected that bigger models will perform better.

The findings are that ResNet, EfficientNet and ConvNext outperformed the baseline RegNet. But due to the increased size they trained a lot slower and due to multiple server crashes could not all be trained completely in time

EfficientNet achieved the overall best performance. This contradicts our assumption that the bigger the model, the better the performance.
This is likely because of its **compound scaling** which balances depth, width and resolution, whereas ResNet and ConvNext rely on deep residual connection, which made training **more time consuming**.

	Resnet 50	***EfficientNet 0***	***ConvNext Tiny***
mAP_12	29.48	31.31	29.44
mAP_10	35.33	35.57	34.48
params	23,532,620	4,022,920	27,829,356
gflops	369.39	35.95	386.54

Train and Validation Loss

While the training loss of all models is relatively similar, EfficientNet’s validation loss decreases **more steadily** than that of the other models, which show a lot of sudden peaks and decreases.

AP for every Class EfficientNet

Class	Average Precision
PASS	86.35
DRIVE	79.53
HEADER	26.08
HIGH PASS	48.59
OUT	16.73
CROSS	29.03
THROW IN	29.73
SHOT	24.14
BALL PLAYER BLOCK	13.36
PLAYER SUCCESSFUL TACKLE	2.12
FREE KICK	16.24
GOAL	3.84

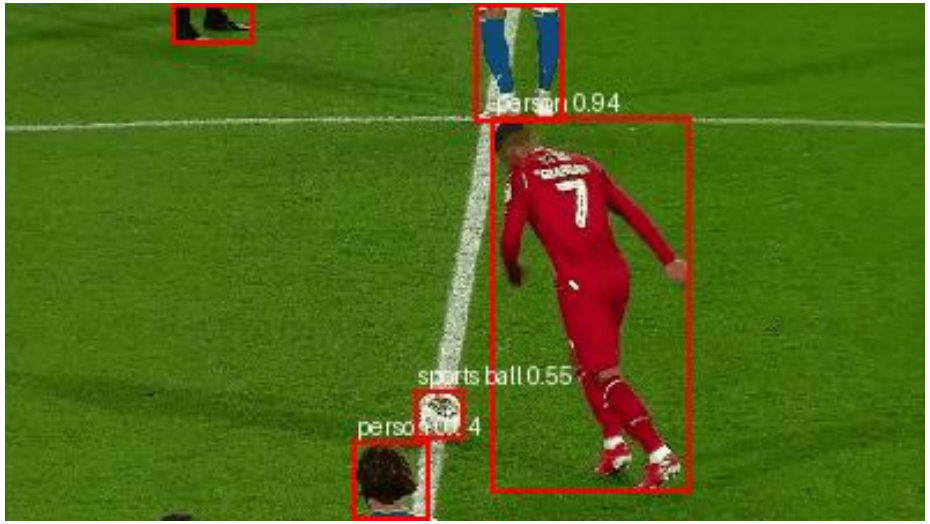
The model performs best on frequently occurring actions like **PASS (86.35)** and **DRIVE (79.53)**, but struggles with less common/more complex actions like **PLAYER SUCCESSFUL TACKLE (2.12)** and **GOAL (3.84)**

Team 8, 3: Adding YOLO Tracking and Conclusion

Idea: Use the pretrained YOLOv8s model to detect and track object identities using its built-in ByteTrack tracking module.

Method: Each video clip is paired with YOLO-based tracking features, which are fused with image features from the RegNetY backbone and passed through the classifier.

Observation: Training didn't converge fast due to the increased input dimensionality and an unoptimized dataloader and the results were also not very good, but this approach could still be promising for the coming week.



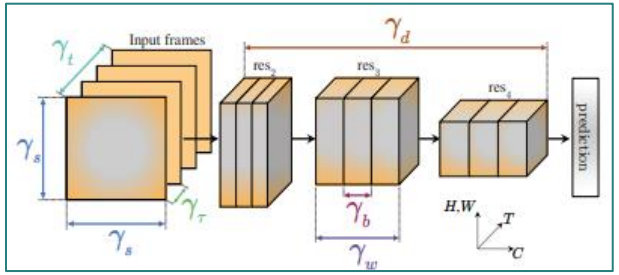
We explored various techniques to enhance a ball action classifier. Our approach included testing **data augmentation** strategies, experimenting with different **pretrained feature extractors**, and incorporating an **auxiliary YOLO-based tracking task.** Through this process, we gained valuable insights into the complexities of video action classification, particularly its **high computational demands and sensitivity to model architecture**. Moving forward, we aim to refine our methods, optimize training efficiency, and further enhance model performance in the upcoming weeks.

mAP scores across all approaches

	Baseline	**Baseline + aug**	**Resnet 50**	**EfficientNet 0**	**ConvNext Tiny**	**Yolo Tracking**
mAP_12	27.79	33.73	29.48	31.31	29.44	12.73
mAP_10	32.74	30.32	35.33	35.57	34.48	15.22

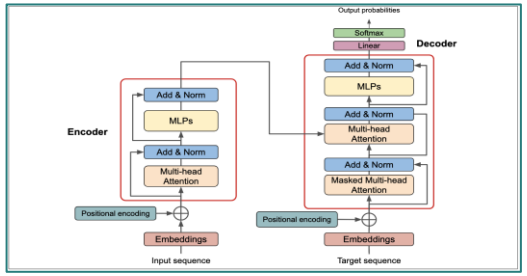
Team 8, 1: Concepts for improving our results based on our own findings from last week as well as other groups.

X3D



After seeing many groups use the **X3D** architecture and experiencing good results, we decided to use it as well this week.

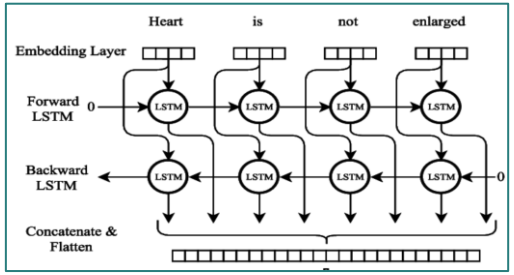
Transformer as a temporal layer



We opted to use a transformer layer because it can model long-range temporal dependencies across video frames by using self-attention.

- This allows the model to focus on key contextual moments throughout the clip.

Bi-directional LSTM as a temporal layer



A BiLSTM (bidirectional LSTM) processes the frame sequence forward and backward:

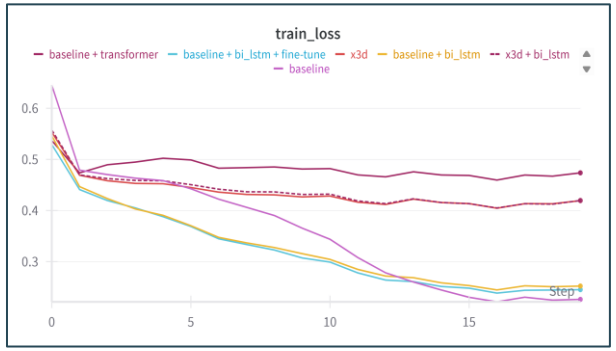
- This allows the model to consider what just happened and what's about to happen.
- We opted for this because it is very useful in sports where actions (like goals, passes, fouls) often rely on past buildup and future confirmation.

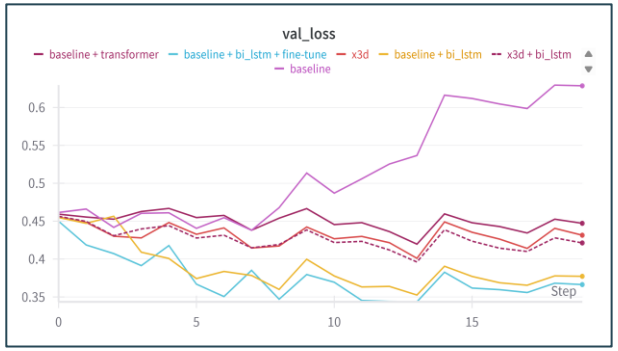
Experiment Set Up

- Since we decided to use smaller models training could be completed much faster than last week.
- We trained each model for 20 epochs.
- Learning rate = 0.0008
- Loss: CrossEntropyLoss with class weights ([1.0] + [5.0 × class])
- Lr scheduler: Linear warmup + cosine annealing
- Optimizer: AdamW

Team 8, 2: Experiment results and analysis

Train loss vs Validation loss

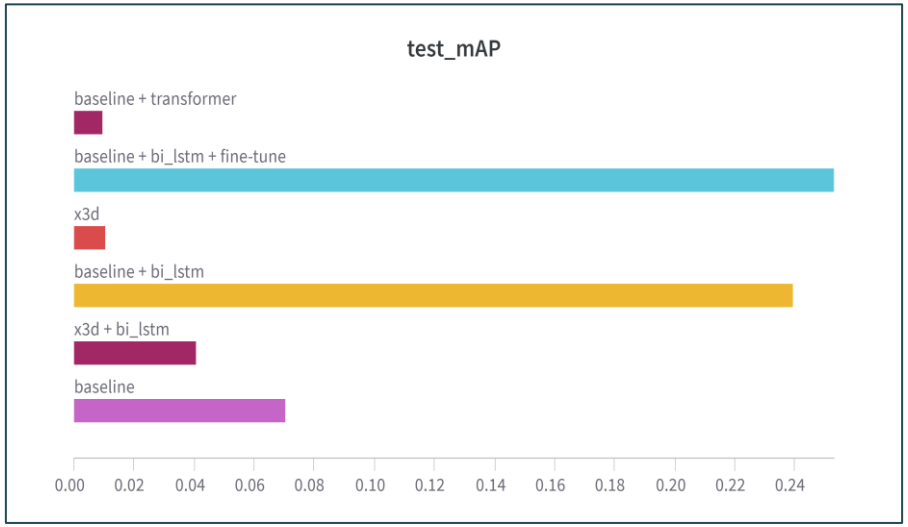




Key Observations:

1. **baseline + bi_lstm + fine-tune** (light blue): achieves the lowest validation loss, consistently improving through training while avoiding overfitting. It also shows strong training loss reduction.
2. **baseline + bi_lstm** (orange): follows closely, with a slightly higher validation loss but very stable behavior. Bi-LSTM helps the baseline significantly.
3. **x3d + bi_lstm** (dark red): shows modest train loss reduction but plateaus in val_loss—suggesting underfitting or limited generalization, possibly due to freezing most of the layers.
4. **baseline** (purple): achieves low train loss but suffers from clear overfitting, with val_loss diverging after epoch 10.
5. **baseline + transformer** (maroon): suffers from the highest train loss and shows a similar val loss.
6. **x3d** (dark orange): shows both a high train loss and val loss, resulting in the model not performing well.

MAP scores across experiments



mAP Results:

- **Best performance**: baseline + lstm + fine tune

⟶ Achieves the highest mAP (≈ 0.25)

- **Baseline + lstm**: performs well, close to fine-tuned version, with a mAP just bellow that (≈ 0.24)
- **Baseline**: relatively low baseline mAP (≈ 0.06), benefited significantly from temporal modeling
- **X3D**: performs poorly as a standalone, has the worst mAP
- **X3D + lstm**: slight improvement over x3d alone, but still lags behind the baseline + lstm combination