---

**Article:**
Ringer, Charles, Nicolaou, Mihalis and Walker, James Alfred orcid.org/0000-0003-2174-7173 (2022) Autohighlight:Highlight Detection in League of Legends Esports Broadcasts via Crowd-Sourced Data. Machine Learning with Applications. 100338. ISSN: 2666-8270

https://doi.org/10.1016/j.mlwa.2022.100338

---

# Autohighlight: Highlight Detection in *League of Legends* Esports Broadcasts via Crowd-Sourced Data

Charles Ringer[a,*], Mihalis A. Nicolaou[b], James Alfred Walker[a]

[a] *Department of Computer Science, University of York, York, YO10 5GE, United Kingdom*
[b] *Computation-based Science and Technology Research Center, The Cyprus Institute, 20 Konstantinou Kavafi Street, 2121, Aglantzia, Cyprus*

[*]Corresponding Author

*Email addresses:* charles.ringer@york.ac.uk (Charles Ringer ), m.nicolaou@cyi.ac.cy (Mihalis A. Nicolaou), james.walker@york.ac.uk (James Alfred Walker )

**Abstract**

Every minute, 500 hours of footage is uploaded to Youtube.com, and $\sim$ 1900 hours of footage is livestreamed on Twitch.tv. It can therefore be challenging for viewers to find the content they are most likely to enjoy. Highlight videos can entertain users who did not watch a broadcast, e.g. due to a lack of awareness, availability, or willingness. Furthermore, livestream content creators can grow their audiences by using highlights as advertisement, while also engaging casual followers who do not watch full broadcasts. However, hand-generating these videos is laborious, thus automatic highlight detection is an active research challenge. We examine automatic highlight detection by focusing on esports broadcasts. Esports are an emerging genre of sport played using a video games. We focus on *League of Legends*, a popular title with multiple professional leagues. Esports broadcasts are high-quality and professionally produced, mirroring traditional sports. We tackle the problem in a weakly supervised manner, utilising two datasets, one of 'crowd-sourced' highlight videos and one of unedited broadcasts. These datasets allow us to leverage massive data while hugely reducing the human cost of data curation and annotation. We propose two novel extensions to state-of-the-art rank-based highlight detection architectures. Firstly, a multimodal hybrid fusion architecture that enables audio-visual highlight detection, and secondly, a smoothing step to incorporate context into decision making. Both extensions show significant improvement over state-of-the-art ranking models, in places performing nearly twice as well as competing architectures. Additionally, we examine the effectiveness of each modality and compare ranking models with classification based systems.

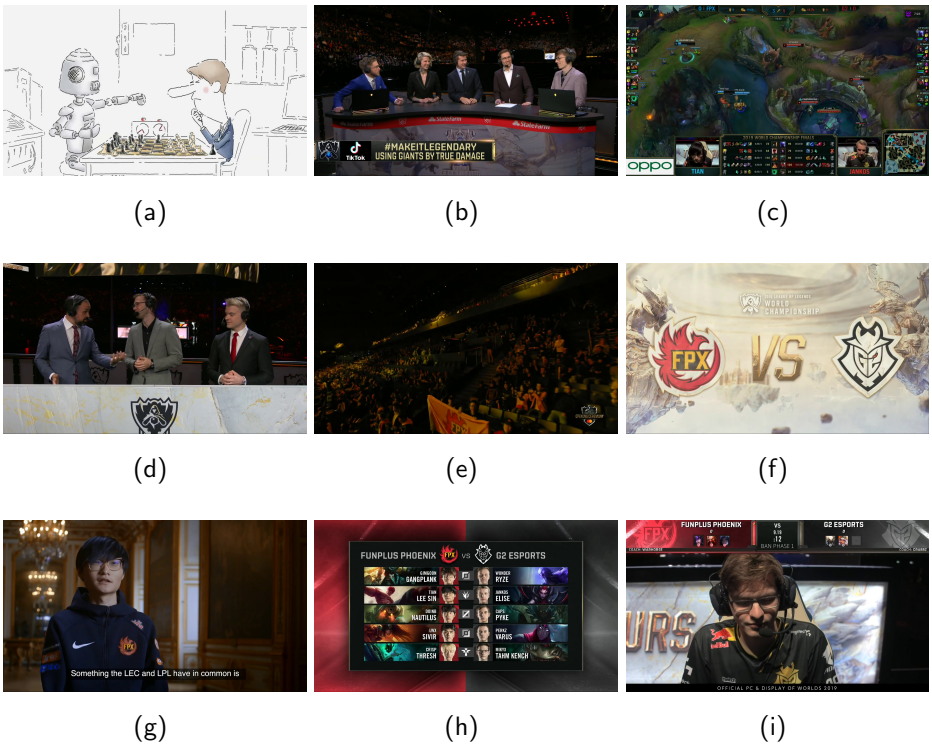
*Keywords:* Highlight Detection, Neural Networks, Livestreaming, Deep Learning, Ranking Networks, Weakly Supervised Learning

## 1. Introduction

Automatic sports highlight detection has been a popular area of research for some time. In recent years the rise of esports, competitive video games often broadcast on livestreaming services such as Twitch.tv, has generated a demand for esports highlight detection tools. Esports are often presented similarly to their traditional sports cousins, e.g. between games an 'analyst desk' discusses the results of previous matches as well as upcoming games. During a match, a set of commentators (known as 'shoutcasters') commentate over video footage of the gameplay. The visual feed shows a variety of scenes, e.g. gameplay, the analyst desk, advertisements etc., as shown in Figure 1. The critical difference to sports broadcasts is the format of the games. There are no constraints to the content of esports, bar the need for it to be digital. This variance results in significant differences in the visual appearance of different esports. Therefore we focus on a single game, *League of Legends*[1], for this work. *League of Legends* is a 'multiplayer online battle arena' game, where matches are contested between two teams of five players. Each team attempts to battle through the opposing defences to reach and destroy the opponents base. Simultaneously teams must protect their base from the other team. Unlike traditional sports where the rules and game mechanics tend to be simple, *League of Legends* is highly complex, e.g. players can choose from over 150 characters to play, each with unique abilities and gameplay.

Due to their video-game based nature, esports have the advantage that those who play them professionally often also livestream out-of-competition play sessions as an additional revenue source. A typical content cycle would be for a live broadcast to occur, either as part of an esports competition or from an individual streamer, then a highlight video would be manually created by a video editor. This highlight video shows the most exciting moments of the broadcast and can be shared through video-sharing platforms, e.g. YouTube. This dual

[1] https://www.leagueoflegends.com/




(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 1: The many different constituent elements which make up a *League of Legends* esports broadcast - (a) advertisements, (b) analyst desk segments, (c) gameplay, (d) commentators, (e) crowd reactions (f) transition shots, (g) feature pieces , (h) champion drafting, (i) player cameras.

content system allows fans to catch up on a broadcast if it was missed and mirrors traditional sports highlights broadcasts, although for esports the two streams are often on different platforms, rather than both being televised.

This work is motivated by the observation that automatic highlight detection offers additional avenues for streamers/broadcasters to reach their audience and for viewers to find content that suits them. An inordinate amount of data is produced on livestreaming and video-sharing platforms. For example, every minute, 500 hours of footage is uploaded to Youtube[2] and about 1900 hours

[2]https://www.statista.com/

of footage is livestreamed on Twitch[3]. Therefore it becomes next to impossible for small scale streamers to gain recognition and grow their audience. It is equally challenging for viewers to find the content they are most likely to enjoy. Automatic highlight detection can aid in several ways. For instance, reducing cost, both in terms of time and cost to an editor, to generate highlight videos is likely to increase the number of smaller streamers who can advertise their work through highlight videos. There is also the potential for personalised highlights, especially for larger channels with a diverse audience. These highlights could be tuned to the viewer's preferences, e.g. by biasing a detection system for the kinds of events a particular viewer enjoys. This type of content would be infeasible without automatic systems, as it would require a human editor to learn the preferences of a large number of viewers and then edit a unique video for each one. Therefore, research into automatic highlight detection for livestreamed esports has practical, real-world applications with benefits for many stakeholders.

This work tackles highlight detection through weakly supervised learning with positive-unlabelled data. This is motivated by this paradigm's apparent advantages over traditional supervised learning. Namely that we can leverage massive amounts of data due to the minimised human cost of data gathering. In particular, we focus on using data in its most raw format, i.e. unedited broadcasts and highlight videos, as these most closely mimic a real-world use case for such a technology. Therefore, videos often contain adverts, analyst desk segments, and other non-game footage. This decision is motivated by the desire to evaluate operationalisable techniques and the observation that human data processing is a bottleneck for such a system.

Determining what should be considered a 'highlight' is nebulous. Which moments are the most interesting is subjective. Different people enjoy different aspects of a broadcast. Additionally, despite the need for a binary highlight detection system, enjoyment intensity can be considered to exist on a continuous scale. Therefore, we gather highlight samples by looking at already generated

[3]https://twitchtracker.com/statistics

highlight videos made by humans and released on video-sharing platforms such as YouTube. We consider that a moment is a highlight if selected by these video editors and thus take a crowd-sourced approach to highlight definition, similar to Xiong et al. (2019). By selecting a range of highlight videos from multiple sources, we aim to generate highlight detection methods that can broadly recreate the style of already popular videos.

Additionally, we implement several state-of-the-art highlight detection methods to provide comparisons for our techniques. Models are trained to provide a highlight value for a given short video segment, such that concatenating these segment predictions produces a time-series 'highlight signal' for the video. This signal is used to determine which moments in a video are the most likely to be a highlight.

The key contributions of this work are summarised as follows:

- A multimodal 'hybrid-fusion' method is proposed to utilise highlight cues from both the visual and audio domains. This proposed method significantly outperforms existing state-of-the-art approaches in this task.

- The application of a smoothing convolution over the predicted time-series highlight signal is proposed, which models video context heuristically. Smoothing significantly improves performance for the hybrid fusion model.

- A thorough comparison of several methods for highlight detection is presented, including state-of-the-art approaches. These approaches consist of unimodal and multimodal systems and utilise various feature representations. This comparison confirms the intuition that modelling more than one modality improves performance and the belief that ranking outperforms classification.

The remainder of this paper is laid out in the following manner. Firstly, in Section 2 we discuss pertinent existing literature, including highlight and event detection, as well as related work in multimodal machine learning. Subsequently, in Section 3 we detail the datasets gathered for this work, both for training and

testing purposes. Next, in Section 4 we discuss the range of methods utilised, breaking down our models into feature extraction techniques, fusion mechanisms, and decision models. Afterwards, in Section 5 we present the details of and results from a series of experiments aimed to examine the effectiveness of these techniques. Finally in Sections 6, 7, and 8 respectively, we briefly discuss the implications of these results, present our conclusions, and suggest future work.

## 2. Related Work

### *2.1. Livestream Highlight Detection*

Livestreaming is a new technology and, as such, has not been widely explored. That said, there are some prior existing studies into detecting highlights in livestreams. Chu & Chou (2015, 2017) examined various facets of *League of Legends* esports broadcasts, building highlight detection models using hand-crafted features, such as the number of players on screen, and event detection, utilising text recognition on in-game messages. While this work is an excellent first work in this area, the reliance on hand-crafted features results in techniques that may not generalise well to other games. Additionally, Ringer & Nicolaou (2018) began to tackle unsupervised learning for highlight detection in the game *Player Unknown's Battlegrounds*, utilising measures of novelty as a proxy for a highlight signal. Although not strictly highlight detection, Ringer et al. (2019) explored supervised learning for joint in-game event and streamer emotion detection, which may act as highlight cues. More recently, Wang et al. (2019) experimented with rank-based highlight detection for the mobile title *Honor of Kings*. This particular work takes a multimodal ranking approach, similar to ours, and therefore we implement their architecture as a state-of-the-art reference point.

The works discussed above focus on audio-visual data (i.e. videos). However, esports are nearly always livestreamed in settings that facilitate viewer interaction via text chat. There is some emerging research into how cues from

the chat domain indicate when a highlight occurs in the game. Han et al. (2019) utilised a bidirectional GRU in a supervised setting to learn a highlight detection model from text chat only, with promising results which outperformed baseline models. Additionally, Liaw & Dai (2020) proposed an attention-based model for highlight detection, utilising the text present in crowd-sourced highlight clips as highlight cues.

### *2.2. Sports Highlight Detection*

Sports are a popular domain for event detection research. Ren et al. (2007) studied highlight detection in soccer games, studying four matches with good results, especially when detecting goals scored. Xu & Chua (2006) used audio-visual features and external text-based cues in their work towards the detection of highlights in team sports. A similar approach, applied to baseball games, was proposed by Chiu et al. (2012). Sun et al. (2010) analysed the excitement level of sports commentators using audio features, mainly Mel Frequency Cepstrum Coefficients (MFCCs) and pitch data, to detect highlights. Nguyen & Yoshitaka (2014) adopt a cinematography and motion-based approach, whereby they analysed the type of camera shots used to detect highlights, especially emotional events.

Spijkerman & van der Haar (2020) utilised convolutional neural networks trained on footage directly obtained from the Formula 1 video game to detect highlights in broadcasts from physical Formula 1 races. Such an approach is made possible because the video game is designed to look realistic. In this work, the models were trained to detect cars within a scene as the authors suggested that such signals are indicators for highlights. Additionally, Moodley & van der Haar (2020) also used CNN models for detecting events in Cricket matches. Specifically, an AlexNet was trained to detect when a 'stroke' occurred, i.e. the batsman hit the ball, with success. Additionally, it seems that approaches focused on Optical Flow calculations, the change in a scene between frames, show promise when summarising sports videos, for example Mendi et al. (2013). Finally, Zhu et al. (2007) utilised behaviour analysis alongside optical flow to

detect the behaviour of racket sport players and thus classify in-game events. While the vast majority of sports highlight detection methodologies are not suitable for our domain, e.g. they focus on supervised learning or use techniques that have since been superseded, they do serve as inspiration for this work, e.g. the application of convolutional networks and optical flow vectors.

### *2.3. Video Highlight and Event Detection*

Outside of sports, there has been some research into highlight detection and, more generally, event detection in video. Much event detection research has been focused on motion. Simonyan & Zisserman (2014) and Feichtenhofer et al. (2016) both utilise optical flow combined with object detection to detect actions performed by humans. Giannakopoulos et al. (2010) also considered motion in their work to detect violent scenes in films. Xu et al. (2015) used unsupervised learning to detect events partly based on motion when analysing scenes of pedestrians walking.

Xiong et al. (2019) recently published a key piece of work in video highlight detection. The authors utilised crowd-sourced videos under the assumption that a short clip would contain a highlight alongside ranking models for state-of-the-art highlight detection in a range of kinetic tasks, e.g. surfing. Ranking models, in general, have become a popular avenue for computer vision-based highlight detection. Sun et al. (2014) took a ranking approach to detect highlights in human action videos utilising crowd-sourced data. Additionally, Video2Gif (Gygli et al. (2016)) is a rank-based detection system trained to generate gifs from longer videos using human-generated gifs as training data. Finally, Yao et al. (2016) also utilised ranking for video summarisation in first-person videos.

### *2.4. Multimodal Machine Learning*

Given the presence of both audio and visual data in most videos, it makes sense to develop techniques that utilise both data sources to enable more robust modelling. Utilising multiple data sources originating from the same event is often referred to as multimodal or multi-view learning. A significant advantage

of utilising multimodal data, as Ngiam et al. (2011) note, is that exploiting information from multiple views can significantly aid learning from noisy or imperfect data. For this work, the term multimodal will describe the paradigm, with 'modality' used to refer to an individual data stream. For ease of discussion, we also extend the definition of modality to refer to different handling of a data stream, e.g. raw video frames and optical flow calculation for video frames are considered separate modalities. One of the biggest challenges for multimodal learning is that the modalities must be combined in a process known as fusion.

Katsaggelos et al. (2015) and Poria et al. (2017) show that there is no consensus on the most appropriate fusion technique, with many studies proposing different approaches. Some of the most popular techniques for fusion in 'end-to-end' systems include early fusion, feature fusion, and model fusion. While early fusion refers to concatenating raw features or representations (e.g. Ngiam et al. (2011); Wöllmer et al. (2013)), feature fusion is usually performed on bottleneck features (e.g. Shah et al. (2014)). Model fusion is where separate models are utilised for each view and subsequently aggregated (e.g. Kahou et al. (2016); Wang et al. (2019)). Multimodal learning, especially fusion, is an active research area. In the paper, we provide only a brief overview of the topic as it relates to this work. We refer the interested reader to Katsaggelos et al. (2015), Zeng et al. (2009) and Poria et al. (2017) for a more comprehensive summary of the area.

### *2.5. Learning from Positive-Unlabelled Data*

Unsupervised, weakly-supervised, and self-supervised learning have all become typical training paradigms due to the ability, as in this work, to leverage massive data without the human effort cost required to label and clean the data. However, this results in noisier training and thus research on how to compensate for this. In particular, our work takes a positive-unlabelled approach to data, where there are concrete labels for only positive training samples. This work implements a noise aware ranking model proposed by Xiong et al. (2019) which itself was inspired by Ilse et al. (2018). This approach can mitigate noise

by utilising a bagging approach alongside a heuristic noise prior. Additionally, we implement a binary classifier as a comparison model and use the weighting methodology proposed by Elkan & Noto (2008) to weight the output of this model by the performance on positive data.

Other attempts at learning from very noisy data include Natarajan et al. (2013) which focused on noisy labels in both positive and negative data and proposed probabilistic 'flipping' of labels to tackle this problem. Similarly, Sukhbaatar et al. (2015) proposed label flipping for computer vision tasks with convolutional neural networks. Further work into noise label vision tasks includes Li et al. (2017) where 'side' information, e.g. clean data, is used to improve learning under noisy conditions. Reed et al. (2015) applied bootstrapping to this problem, finding that it creates a robust classifier in multi-class scenarios. Additionally, Liu & Tao (2016) applied Importance Reweighting to noisy label classification. Finally, $\propto$-SVM (Yu et al. (2013)) is a Support Vector Machine that models the latent labels for unknown data alongside the proportions of known labels. This concept has been applied to event detection in video data, e.g. Lai et al. (2014).

## 3. Data

### *3.1. Training Data*

This work tackles highlight detection through weakly-supervised learning through the application of two datasets, one of only positive samples and one of unlabelled samples. This data-label formulation is similar to traditional semi-supervised learning, but labels only exist for positive samples. There is a considerable wealth of broadcast data and fan-made highlight videos. Broadcast data contains a mixture of highlight moments, which we aim to capture, and non-highlight moments, whereas highlight videos contains only highlight moments. In an ideal world, it would be possible to find the highlight moments in the broadcast through manual annotation. For example, by utilising a set of participants who would watch the broadcast and select their favourite moments.

However, manual annotation is extremely expensive and time-consuming, prohibiting leveraging the benefits of applying deep learning to massive datasets. Another reasonable approach would be to align the highlight video segments with those in the original broadcast, e.g. by detecting which frames from the broadcast are also in the related highlight video. However, while less time consuming than manual annotation, it still has a significant data gathering penalty as naming conventions for both broadcasts and highlights are inconsistent, requiring human preprocessing. Furthermore, the frame-matching process is computationally expensive. Such an approach also requires a corresponding highlight video for every broadcast video and a broadcast video for each highlight video in the dataset. These challenges make automated broadcast-highlight matching impractical for large-scale datasets.

Instead, the positive-unlabelled approach we use, inspired by Xiong et al. (2019), involves collecting two datasets, one of the mixed labels, from broadcasts and one of the positive labels, from highlight videos, but with no connection between the datasets. The broadcast dataset contains replays of *League of Legends* professional broadcasts from various competitions. The dataset of human-made highlight videos was gathered from multiple highlight makers who post their videos online. Some highlight videos contain footage from a single game. Others show highlights from multiple games, especially if the contest is in a 'best of n' format. Figure 2 demonstrates how our data differs from the two options for supervised learning and, in doing so, minimises the cost to gather data. In total, 257 broadcast videos and 676 highlight videos were gathered, resulting in more than 193 hours of broadcast video and more than 178 hours of highlight footage. Minimal preprocessing was applied to most closely mimic a real-world automated system. Each video was processed by converting frame dimensions to $224 \times 224 \times 3$. Additionally, the frame rate was set to 25 frames per second. No audio prepossessing was applied.

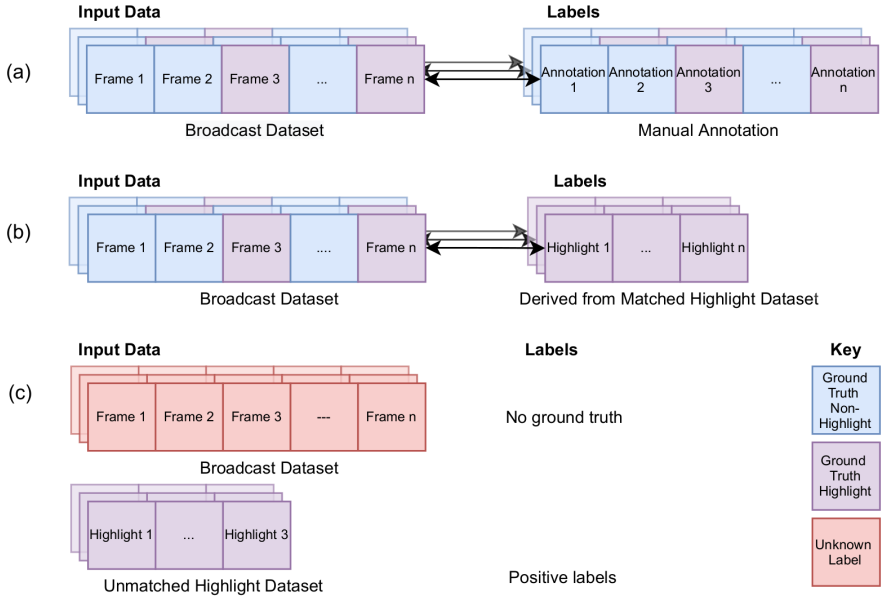



Figure 2: Comparison of the different potential dataset formulations. Notice that for traditional manual annotation (a) or frame matched annotation (b), each video in the broadcast dataset needs a corresponding set of labels, either through expensive manual annotation or a matched dataset of highlights. This is not the case with our data (c). The cost to gather training data is therefore minimised, and there is no requirement to have corresponding highlight videos for each broadcast video.

### *3.2. Test Data*

While our training structure follows the positive-unlabelled paradigm, highlight detection in esports broadcasts is an emerging area of research, and so there is currently a lack of test sets with which to evaluate models. Therefore we additionally constructed a test set of data with labels, used to evaluate the range of models presented in this work. To do this, we select 20 games, five each for the four major *League of Legends* esports leagues - LCS (North America), LEC (Europe), LPL (China), LCK (Korea). Following the frame matching annotation technique described above, we manually retrieve the human-made highlights from the two most popular highlight channels and match frames between the original broadcast and the highlight videos to find which points in

the main broadcast the highlights appear. These highlight channels each have $275,000$ and $356,000$ subscribers. In contrast, the official *League of Legends* esports channel hosting full-length games has fewer subscribers with $252,000$, and no other highlight channels have more than $40,000$ subscribers. This process results in a test set of 20 complete games, constituting 18 hours, 6 minutes and 51 seconds of broadcast data. It also contains a broad mix of broadcast settings, players, advertisement, and commentators from across the world, thus mimicking real-world challenges, as shown in Figure 1. This test set size is in keeping with prior works; Wang et al. (2019) used just four videos for evaluation and Chu & Chou (2015) used 24 games but from only one tournament, the 2014 World Championship.

To determine ground truth annotations for a given broadcast video, we compare frames from both highlight videos associated with the original broadcast. We select the broadcast frame that is the closest match, given a distance threshold as some highlight videos contain content absent in the original broadcast, e.g. the content creators personal brand. We sub-sample frames by a factor of eight for processing speed, resulting in a highlight annotation fidelity of 0.32 seconds. This process is still computationally expensive and took several days to calculate labels for our dataset, further highlighting the advantage of our positive-unlabelled approach, which deliberately minimises processing time. To capture the breadth of content which can be considered a highlight, we label segments of the original broadcast as a highlight if it appears in either of the two highlight videos. In total, this results in 203805 individual test set samples, with the shortest video contributing 4840 samples and the longest contributing 14498 samples. Overall each video in the test set contributed a median of 10003 samples.

## 4. Methods

Existing state-of-the-art highlight detection models primarily have similar components. The raw data is generally passed through feature extraction sys-

tems, e.g. a Convolutional Neural Network (CNN) or audio frequency analysis techniques. While it is theoretically possible to act on the raw data itself, there are generally many inputs in the case of both audio and visual data, making this challenging. Therefore, using a feature extractor tasked with finding a smaller number of salient features is preferred. These are then used in the downstream systems. Towards the end of the pipeline, there must be a decision-making mechanism that determines if the incoming features form part of a highlight or not. The output from this decision-making mechanism is usually the final output of the system and consists of a time-series signal over the length of a video where each point in the signal relates to the likelihood that a single video segment is a highlight. Additionally, at some point between raw input and final output, the data from each input modality, e.g. visual data, audio data, needs to be 'fused' to unify the system. Exactly how this fusion mechanism is implemented can vary between models. For example, it may be possible for particular data formats to fuse the raw inputs, e.g. concatenating an RGB and Optical Flow image together at the raw data level. Alternatively, fusion can occur after feature extraction so that the decision-making process has a single set of features to make a decision with, which we refer to as 'feature fusion'. Finally, perhaps each modality is trained entirely separately, including decision making, and then the decisions made for each modality are aggregated in some form. We call this aggregation 'model fusion'. For a particular penitent example of model fusion, see Wang et al. (2019).

There are very few state-of-the-art models that utilise data in a format similar to ours. The only existing model which approaches audio-visual highlight detection for esports titles using this approach is Wang et al. (2019), which we implement here as a state-of-the-art comparison model. Models such as Chu & Chou (2015, 2017) rely on hand-crafted features, and Ringer & Nicolaou (2018) is novelty-based in a one-shot setting and thus extremely computationally expensive owing to the need for training on each video. Therefore, these approaches are not suitable for inclusion in this paper as comparison models. We also implement the architecture from Xiong et al. (2019) as it heavily inspired

our approach, although it is unable to utilise the benefits of the multimodal nature of our data because it omits a fusion step. The system architecture proposed in this work is discussed in detail below and is additionally presented in Figure 3.

### *4.1. Feature Extraction*

Our data is multimodal. Therefore, different pre-trained feature extraction models are required for each modality. While we have only two data sources - video frames and audio, we treat our input data as four separate modalities. Firstly we take a stack of RGB video frames. Secondly, we convert those RGB frames into optical flow frames to represent motion within a video segment. Thirdly, we extract features from only a single frame. Finally, we use audio samples. Precisely how these features are extracted is detailed below.

#### *4.1.1. Video Feature Extraction*

There is evidence that pretraining a network on a large, generic dataset outperforms training a network to specifically generate features for a particular task, e.g. Alwassel et al. (2020). To this end, we utilise a state-of-the-art inflated 3D convolutional network from Carreira & Zisserman (2017) which has been pre-trained for action recognition on the kinetics dataset (Kay et al. (2017)). This model is a two-stream model which considers both RGB video frames (RGB) and optical flow frames (Flow). It processes both modalities separately but using the same convolutional architecture. A 3D CNN is similar to a 2D CNN, e.g. as used in image processing, but with an additional convolution dimension that convolves over time, represented as a stack of frames. We use a 3D CNN rather than a CNN-RNN style architecture, where the spatial elements of a frame are modelled with a 2D CNN, while the temporal aspect is modelled using a recurrent neural network, e.g. Ringer et al. (2019). This is motivated by recent works which have shown that, at least for short video clips, a 3D CNN is generally more performant (Carreira & Zisserman (2017)). Because this model is pre-trained, we are limited to using an input dimension equal to

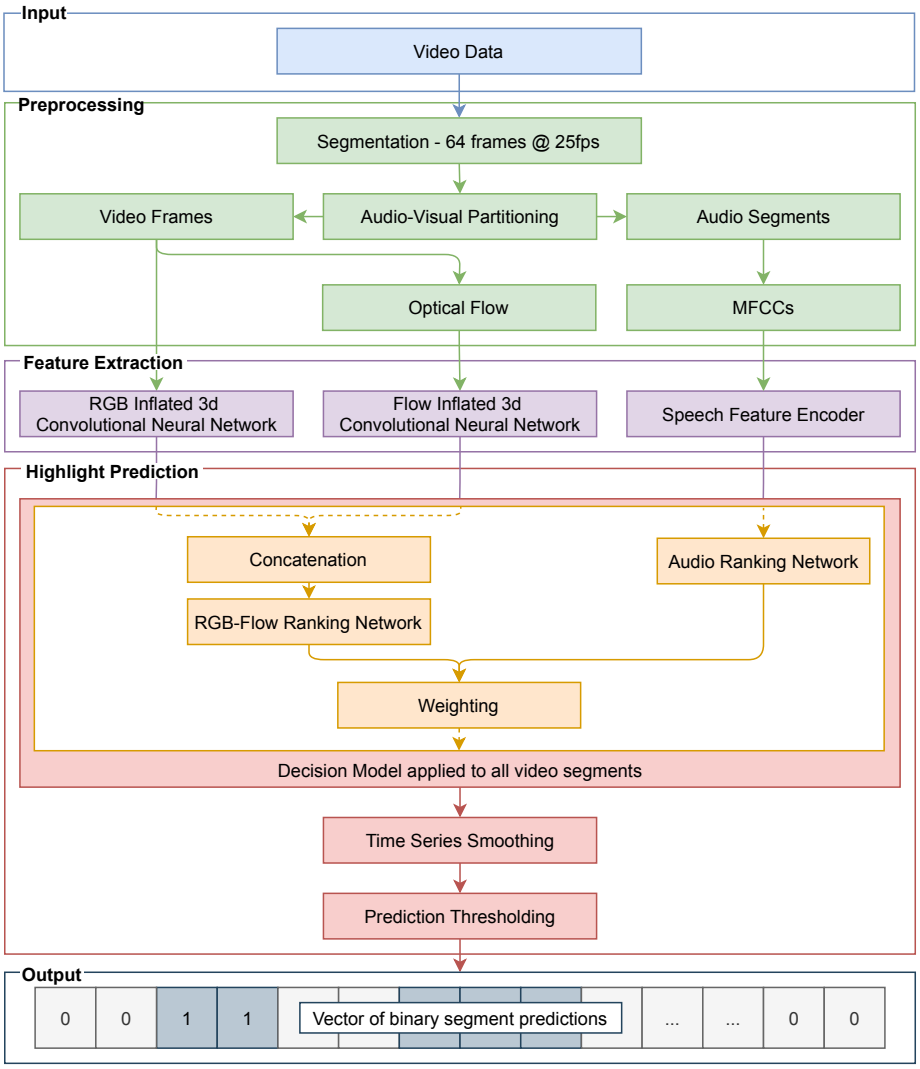



Figure 3: Full system architecture for Auto-Highlight model using multimodal hybrid fusion and smoothing. This figure details the end-to-end system, from input data, i.e. video data, through prepossessing, feature extraction, segment prediction and finally smoothing and thresholding operations. The output is a binary vector where each element relates to if a video segment is predicted as a highlight or not.

the pre-trained model. This requirement is doubly a limitation for 3D CNNs as it is a constraint that the input time dimension is fixed, unlike a CNN-RNN architecture. Therefore we use frame sizes of $224 \times 224 \times 3$ pixels and a stack of 64 frames total. Video data is sampled at 25 frames per second, and therefore 64 frames equate to 2.56 seconds of video per training sample. For both data streams, RGB and Flow, 1024 features are generated for each 64 frame video segment. The RGB network receives raw frames taken directly from the video. We use the optical flow method proposed in Farnebäck (2003) on greyscale frame data for the optical flow network.

#### *4.1.2. Audio Feature Extraction*

Audio is a crucial part of esports broadcasts because it represents a mixture of in-game audio and commentator speech, and therefore it is pertinent to include it in our models. Audio data is represented as a time-series vector of data sampled at $192,000$ hertz. To keep the length of the audio samples consistent with the video data, we group $491,520$ audio samples, i.e. 2.56 seconds, into a single audio training example. We then process these samples by calculating the mel-frequency spectrograms and passing these into a state-of-the-art pretrained speech feature encoder (Jia et al. (2018)). The encoder consists of several LSTM layers and generates 256 audio features for use in the downstream decision-making model.

#### *4.1.3. Single Frame Extraction*

One of the existing state-of-the-art approaches we implement, Wang et al. (2019), uses features from a 2D CNN that extracts features from only a single frame in addition to the video features detailed above. To offer a comparison to this model, we also implement the same feature extraction. An Alexnet (Krizhevsky et al. (2012)) network trained in object recognition on the ImageNet database (Deng et al. (2009)) is utilised. We extract features from the final bottleneck layer before classification is performed, resulting in a feature vector of 9216 features for a single frame.

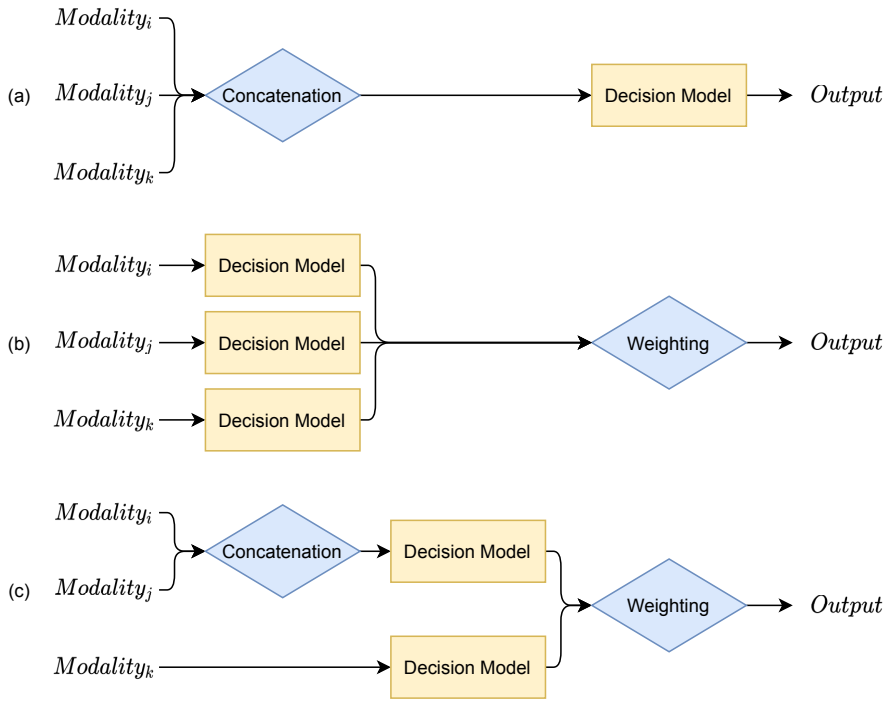



Figure 4: Comparison of Feature (a), Model (b), and Hybrid (c) fusion architectures. For simplicity three modalities are shown but in practice any number of modalities can be used.

### *4.2. Fusion*

Three fusion methods, detailed below, are presented in this work. For a visual overview of how these fusion methods differ, see Figure 4.

#### *4.2.1. Feature Fusion*

Feature fusion is perhaps the most intuitive method for fusion. After feature extraction, the modalities are fused, e.g. via concatenation, resulting in a single decision model provided with all modalities. However, multimodal feature fusion may not work well in certain situations. For example, if there is a significant disparity between the number of features from the modalities, the modality with more features will likely be over-represented in the decision-making model output. This is especially worth considering when comparing our audio feature extraction, which has 256 features, and our single frame feature extraction,

which has 9216 features. Therefore, we reserve this approach for the modalities which produce the same number of features, i.e. RGB and Flow.

#### *4.2.2. Model Fusion*

Model fusion utilises a decision-making model for each modality and then fuses them via weighted aggregation of the outputs. For a given video segment a highlight prediction is generated for each modality via separately trained decision-making models, $p_m$. The predictions for each modality are then weighted by $w_m$. This weighting allows the mechanism to be fine-tuned based on the performance of each modality. The weighted predictions are then summed to produce a single segment prediction $p$ (Equation 1). Past studies (e.g. Ringer et al. (2019), Katsaggelos et al. (2015), Poria et al. (2017)) have shown that sometimes fusing later in the system can improve performance. That said, model fusion is performed on decisions rather than features and thus loses fidelity in the fusion step, although it is unclear how much this impacts performance. In a positive-unlabelled setting such as ours, we need to heuristically determine the weights for each modality because we cannot evaluate the performance of our model on training data that mimics our test data, unlike in a supervised setting. In future works with a more extensive test set, it may be possible to reserve a portion of the test set for fitting hyper-parameters, thus reducing the reliance on heuristically chosen values. For all of our models, we use equal weighting for each modality. However, this is not a requirement, e.g. Wang et al. (2019) used uneven weighting, which put more focus on the video model. For their work, they selected the weights $\{0.7, 0.15, 0.15\}$ for RGB, Frame, and Audio outputs, respectively.

$$p = \sum_{m \in modalities} p_m w_m \tag{1}$$

#### *4.2.3. Hybrid Fusion*

Finally, we propose a mixed approach called hybrid fusion. Both the RGB and Flow feature extraction networks are applied to the same input data domain,

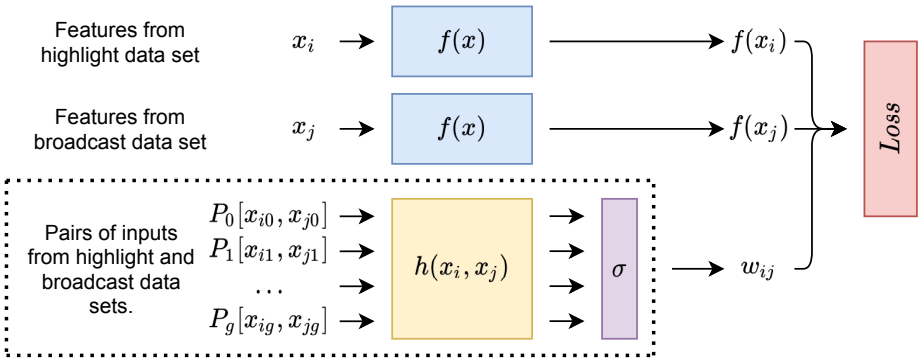



Figure 5: The rank network architecture. $f(x)$ and $h(x)$ are both multi-layer perceptrons.

visual data, and have the same number of features. Therefore it makes sense to perform feature fusion because it is performed on the rich domain features, thus providing more information to the decision model. However, feature fusion is not appropriate for cross-domain modalities, i.e. visual and audio data, due to the disparity in feature vector size. Therefore model fusion needs to be employed to fuse audio and visual modalities. To maximise performance, we apply feature fusion to two visual modalities and apply model fusion between the visual decision-making model and the audio decision-making model. Hybrid fusion has the added benefit that it equally weights visual and audio decision making. We hypothesise that this allows the system to utilise rich features from the same domain together, e.g. RGB and Flow data, while maintaining the ability for the model to work with other modalities, e.g. audio, without having a significant imbalance in terms of the number of features per feature set. Thus, we hypothesise that hybrid fusion will outperform both feature fusion of just the visual modalities or RGB, Flow, and Audio model fusion.

### *4.3. Decision Making*

At some point, a decision needs to be made about if the extracted features constitute a highlight or not. This is a tricky task because our input data is extremely noisy. Data within the 'highlight' dataset are established as positive labels, as segments in this dataset are selected by a video editor to be a highlight.

However, we know with certainty that the broadcast dataset is noisy. It must contain a mixture of highlight and non-highlight moments because the highlight moments are sampled from the original broadcast data. Therefore, a noisy-label mitigation mechanism is likely to yield improvements.

We employ a ranking network for our decision-making mechanism due to their prevalence in literature. Concretely, a ranking network is a multilayer perceptron with a single output. The model is provided with two samples during training, one from each dataset. The expected output is that the sample from the highlight dataset is 'ranked' higher than the sample from the broadcast dataset, i.e. the output for the highlight sample is greater than the broadcast sample. Each sample uses the same set of layers such that weight updates affect the ranking of samples from both datasets. Often for such models, a Hinge loss is used for weight adjustment.

For our model, we follow the modification proposed by Xiong et al. (2019). Namely, we add a secondary network responsible for determining if an input pairing is 'valid' or not. Because our non-highlight labels are noisy, it is possible that a particular input pairing contains a sample from the broadcast dataset which should be considered a highlight. Therefore this secondary network takes a group of samples and, under the assumption that one sample is invalid, applies a weighting to all losses in that batch. For our work, we assume that one in eight samples is invalid because matches are often about 40 minutes to one hour long, and highlight videos tend to be between five and eight minutes. This modification performed well in Xiong et al. (2019) and given that their noisy data is similar to ours, although a different domain, such a modification is likely to yield good results in this work as well. Figure 5 details the architecture used. The model is optimised against the loss function in Equation 2 where $f$ is the ranking network, and $h$ is the validity network. For further detailed discussion regarding this modification and its motivation, we refer the interested reader to Xiong et al. (2019).

$$
\begin{aligned}
Loss = \sum_{g=1}^{m} \sum_{(s_i,s_j)\in P_g} \tilde{w}_{ij} max(0, 1 - f(x_i) + f(x_j)) \\
s.t. \sum_{(s_i,s_j)\in P_g} \tilde{w}_{ij} = \sum_{(s_i,s_j)\in P_g} \sigma_g(h(x_i, x_j)) = 1 \\
\tilde{w}_{ij} \in [0, 1]
\end{aligned}
\tag{2}
$$

*4.3.1. Ranking Model Weights*

Because each modality has a different number of input features, it makes sense to vary the number of layers in the ranking network accordingly. The architectures that we use are detailed in Table 1, alongside details about the number of weights and memory requirements of the models. Two values are given for the memory requirements of each model. Training memory refers to training both $f(x)$ and $h(x)$ whereas inference memory refers to just calculating a prediction value from $f(x)$. Accurate memory profiling with Keras is currently an open issue, and thus this work presents an approximation[4]. Note that the implementation for Wang et al. (2019) uses an architecture of shape $[64, 1]$ for the audio ranking network (Audio A). However, we noticed it performed poorly, so we additionally implemented a $[512, 128, 1]$ architecture (Audio B) which was used for all models except the comparison to Wang et al. (2019). All layers use a ReLU activation function.

*4.3.2. Binary Classifier*

In Section 5.4 we compare the ranking approach to a binary classifier. Ranking models are popular in literature, yet it is still valuable to compare ranking with binary classification on our data to examine if the assumed benefits of using a ranking network exist. We replace the ranking models used in our hybrid fusion model with binary classifiers. These classifiers are multilayer perceptrons with the same topology as a single instance of the ranking network, without the

[4]https://github.com/tensorflow/tensorflow/issues/36327. For this work, we use the solution proposed by James Mishra.

Table 1: Architectures for each ranking model. The architecture is detailed as a list of feedforward neuron layers, reading left to right. The weights of the ranking network, $f(x)$, and validity network, $h(x)$, are also detailed. Memory requirements are listed in MB and are an approximation. *Feature fusion, †Wang et al. (2019).

| Modality | Architecture | $f(x)$ Weights | $h(x)$ Weights | Training Memory | Inference Memory |
|---|---|---|---|---|---|
| RGB | [512, 128, 1] | 590,593 | 1,114,883 | 281.10 | 0.60 |
| Flow | [512, 128, 1] | 590,593 | 1,114,883 | 281.10 | 0.60 |
| RGB + Flow* | [1024, 512, 128, 1] | 2,688,769 | 4,785,923 | 572.08 | 2.70 |
| Audio A† | [64, 1] | 16,513 | 32,899 | 68.68 | 0.02 |
| Audio B | [512, 128, 1] | 197,377 | 328,451 | 78.60 | 0.20 |
| Frame | [4096, 1024, 512, 256, 128, 64, 1] | 42,645,507 | 80,394,243 | 2,639.04 | 42.71 |

additional validity network. They are trained on the same data by assuming that all samples in the highlight dataset are positive samples and all samples in the broadcast dataset are negative, although we know that the broadcast dataset contains mixed labels. Additionally, because the classifier network is performing binary classification, the activation for the neuron in the final layer is a sigmoid, not ReLU, and the loss is calculated using binary cross-entropy.

To mitigate the known deficits of this training paradigm, we implement the positive-unlabelled weighting technique presented by Elkan & Noto (2008). Simply, we reserve a small portion of the training datasets (in our case, 10%) as test data. Then we calculate the average output for all positive samples in the test data and use this value as a weighting for the output when determining if a sample is a highlight or not. Intuitively, the average output for all positive samples can be seen as a confidence score relating to how confident the network is that a known positive sample is positive. When using these classifier models in our system, we multiply the output by the weighting. This weighting reduces the output of the model and, as Elkan & Noto (2008) shows, shrinks the decision boundary, thus making positive classifications less likely but improving precision. This weighting does not affect average precision calculation, the main

metric used in our experiments, but has a noticeable effect on classification given a fixed decision threshold. These weighting constants were calculated as; RGB and Flow: 0.97, Audio: 0.71.

### *4.4. Time-Series Smoothing*

One apparent weakness of the ranking approach is that the model must decide if a video segment is a highlight based on only that segment. However, in reality, a human video editor would not do this. Instead, they would use the context of surrounding video clips to inform their decision. For instance, an uninteresting segment would likely be included if it links two exciting sections. Likewise, a short but exciting section may be omitted from the highlight video if clips before or after are uninteresting or contribute little to the match's narrative. However, influencing the segment decision making based on factors outside that segment is impossible during training because our positive label samples, i.e. those taken from the highlight dataset, are gathered without non-highlight context. The only way in which it would be possible to intrinsically model context would be if the dataset was fully labelled, e.g. scenarios (a) and (b) in Figure 2, although as discussed, this is infeasible for large-scale datasets. Therefore a technique for feature-agnostic modelling is required to consider context in the decision making process.

A solution to this challenge is to apply a smoothing convolution to the highlight signal across the whole video, which allows for context modelling in a heuristic manner. This smoothing increases the highlight signal for segments in a broadly interesting portion of the video and reduces the signal for moments in a generally uninteresting segment. Because the smoothing convolution is applied to the highlight signal rather than included as part of the rank model training, it can be applied to our positive-unlabelled training data. We use a convolution kernel of length $L = 101$ and a half-kernel length, used in kernel initialisation, of $n = \lceil L/2 \rceil$ because this roughly equates to 30 seconds of footage surrounding the sample, due to the sliding window segmentation approach described in Section 5. It is not clear which values would be most appropriate for this smoothing

kernel, so we implement three options:

- Linear: A uniform smoothing kernel where each kernel element has the same value. The kernel initialisation is described in Equation 3.
- Gaussian: A Gaussian with height 1 and the peak at the centre of the kernel. The standard deviation is set to $n/2$. The kernel initialisation is described in Equation 4.
- Power of 2: Each element is calculated as $2^{-x}$ where $x$ is the distance to the centre of the kernel. The kernel initialisation is described in Equation 5.

$$k = 1 \in R^n \tag{3}$$

$$k = Gaussian(i, 1, 0, n/2), \forall i \leq n \tag{4}$$

$$k = 2^{-i}, \forall i \leq n \tag{5}$$

All kernels are then normalised such that all values in the kernel sum to 1, Equation 6. The smoothed value $x'_i$ for a given point in the time-series highlight signal $x_i$ and kernel $k$ is calculated via Equation 7. Note that the method demonstrated here initialises only approximately half of the kernel weights. This is for practical reasons and is possible because the kernel is symmetrical. In theoretical terms, position one in a kernel vector represents the kernels centre, position two represents the two values on either side of the centre and so on. When calculating the resultant value, the absolute index is used to correctly index the right element of the kernel vector when calculating the value for time-series samples before the current time step.

$$k' = \frac{k_i / min(k) \forall k_i \in k}{\sum_{i=0}^{k} k_i} \tag{6}$$

$$x'_i = \sum_{j=-n}^{n} x_{i+j} \cdot k_{|j|} \tag{7}$$

## 5. Experimental Results

In total, four experiments are presented in this work. Firstly, we compare all single modality models, including Xiong et al. (2019). Secondly, we compare all multimodal models, including our Hybrid fusion approach as well as Wang et al. (2019). Thirdly, we experiment with the time-series smoothing step. Finally, we compare ranking models to binary classification, with and without noisy label mitigation. All models are implemented in Python using a range of popular machine learning libraries. PyTorch (Paszke et al. (2019)), Keras (Chollet et al. (2015)), and Tensorflow (Abadi et al. (2015)) were all used to compose and train models, with Numpy (Harris et al. (2020)), Scikit Learn (Pedregosa et al. (2011)), Librosa (McFee et al. (2015)) and OpenCV (Bradski (2000)) used for a range of auxiliary functions. All models were trained using 2 PCs. The first has a Ryzen 5 1600 CPU, 16GB of RAM and an Nvidia 2080ti GPU. The second had an Intel i7 9700 CPU, 64GB of RAM, and 2x Nvidia Titan RTX GPUs. These machines were deliberately chosen because they represent consumer-grade hardware and thus indicate if the techniques described in this work are likely to be operationalisable.

We have a highlight label fidelity of 0.32 seconds. However, due to the limitations of the feature extraction models, we must assess clips of length 2.56 seconds. We operate a sliding window system where a segment's annotation is calculated from the timestamp in the middle of the segment.

All experiments compare several models by evaluating their performance on the test set. The principal metric for performance is the mean average precision ($\bar{m}AP$). This metric is popular in highlight detection literature because it evaluates performance without determining a decision boundary for the model, which is a tricky task given that ranking networks are deliberately trained without one. $\bar{m}AP$ is calculated by taking the mean of the average precision ($AP$) for all videos in the test set. $AP$ is calculated by averaging the precision achieved at all recall values for all samples in a video. It is not clear if we would expect this average precision to be normally distributed, and as such, we additionally re-

Table 2: Results for Experiment One. Minimum Average Precision, Maximum Average Precision, Mean Average Precision, and Medium Average Precision are shown for four single modality models alongside a random baseline. [1]Xiong et al. (2019).

| Modality | Min. AP | Max. AP | $\bar{m}AP$ | $\tilde{m}AP$ |
|---|---|---|---|---|
| Random | 0.025 | 0.394 | 0.118 | 0.104 |
| RGB[1] | 0.049 | 0.454 | 0.328 | 0.325 |
| Flow | **0.064** | 0.513 | 0.335 | 0.332 |
| Audio | 0.041 | **0.574** | **0.366** | **0.343** |
| Frame | 0.031 | 0.371 | 0.151 | 0.147 |

port the median average precision ($\tilde{m}AP$). Precision, in general, is a reasonable metric for evaluating our models because the quality of a highlight video suffers if it contains many dull non-highlight moments. A high recall and low precision model would generate a very long video that would be unlikely to function as a 'highlight reel'. Additionally, accuracy is an inferior measurement because our positive class constitute a tiny amount of the test data.

### *5.1. Experiment One - Modalities Comparison*

The first experiment compares the performance of each modality if it is treated as the sole model for highlight detection. Here the four modalities, RGB video frames (RGB), optical flow video frames (Flow), audio (Audio), and single-frame features (Frame), are compared alongside a random baseline. The random baseline merely assigns a random value between $[0 - 1]$ for each video sample for all videos in the test set, and is included to demonstrate the baseline challenge of this task. Note that because Xiong et al. (2019) uses just RGB frames to determine highlights, the RGB modality here represents the implementation of that approach.

Table 2 details the minimum AP (Min AP), maximum AP (Max AP) as well as both averaged AP measures ($\bar{m}AP$ and $\tilde{m}AP$) across all videos in the test

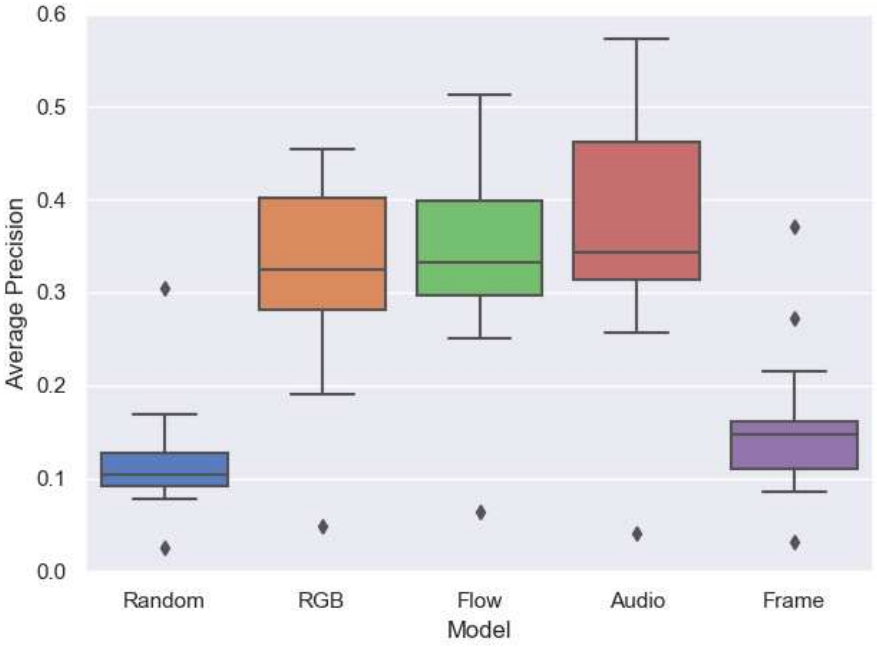



Figure 6: Box and Whisker plots for all models evaluated in Experiment One.

set for the five models tested in this experiment. To accompany this, Figure 6 presents box and whisker plots showing the distribution of results for each model across each video in the test set.

To test the statistical significance of our results, we perform a Wilcoxon signed-rank test on our models, the results of which can be found in Table 3. Assuming that a p-value of $< 0.05$ indicates statistical significance, we see that, unsurprisingly, the random model is significantly different to all of the other models. Likewise, the single-frame features are also significantly different from the modalities that capture temporal dynamics in some way (RGB, Flow, and Audio). Perhaps unsurprisingly, given that the feature extraction models are trained in the same manner following a similar architecture, the RGB and Flow models are not significantly different. The RGB and Flow models are close in $\tilde{m}AP$ performance to the Audio model, indicating the importance of all three modalities. However, the predictions from the Audio model are significantly different to the visual models, perhaps due to audio cues indicating different types of highlight to visual cues. This difference may further indicate the benefits of a multimodal approach.

Additionally, we calculate the A-measure (Vargha & Delaney (2000)), a statistical test for effect size, for each pair of models. An A-measure value of $a > 0.5$ indicates that one model is better than another, with values $a > 0.56$ indicating a small effect, $a > 0.64$ indicating a medium effect, and $a > 0.71$ indicating a large effect. The A-measure for a pair of models is symmetric around 0.5 and sum to one. This means that additionally $a > 0.44$, $a > 0.36$, and $a > 0.29$ and indicate small, medium and large effect, albeit negative effect. We see a large effect in the performance improvements of the temporal models, i.e. RGB, Flow, Audio, over the single-frame model. We also see a medium effect size when comparing Audio to RGB and Flow, indicating its performance benefits. These results are also found in Table 3.

Table 3: Statistical testing results for Experiment One. The upper right triangle details Vargha Delaney A measure. Values less than 0.5 indicate column outperforms row and values more than 0.5 indicate row outperforms column. † denotes a small effect size ($a < 0.44 \vee a > 0.56$). ‡ denote a medium effect size ($a < 0.36 \vee a > 0.64$). $\diamond$ and bold denotes a large effect size ($a < 0.29 \vee a > 0.71$). The lower left triangle details p-values calculated via a Wilcoxon signed-rank test. Bold denotes significant values, i.e $p < 0.05$.

| | Random | RGB | Flow | Audio | Frame |
|---|---|---|---|---|---|
| Random | - | **0.06**$^{\diamond}$ | **0.06**$^{\diamond}$ | **0.05**$^{\diamond}$ | 0.32$^{\ddagger}$ |
| RGB | $\mathbf{2 \times 10^{-6}}$ | - | 0.48 | 0.39$^{\dagger}$ | **0.91**$^{\diamond}$ |
| Flow | $\mathbf{2 \times 10^{-6}}$ | $5 \times 10^{-1}$ | - | 0.42$^{\dagger}$ | **0.91**$^{\diamond}$ |
| Audio | $\mathbf{2 \times 10^{-6}}$ | $\mathbf{3 \times 10^{-3}}$ | $\mathbf{3 \times 10^{-2}}$ | - | **0.92**$^{\diamond}$ |
| Frame | $\mathbf{3 \times 10^{-3}}$ | $\mathbf{2 \times 10^{-6}}$ | $\mathbf{2 \times 10^{-6}}$ | $\mathbf{2 \times 10^{-6}}$ | - |

### *5.2. Experiment Two - Comparison of Multimodal Models*

Experiment Two provides a comparison between several multimodal approaches. The system proposed by Wang et al. (2019) is presented alongside our hybrid fusion model and a selection of other models indicative of the performance gains from multimodal systems. Three model fusion models are presented, RGB + Flow, RGB + Audio, RGB + Flow + Audio, and one feature fusion model, RGB + Flow.

Similarly to Experiment One, Table 4 shows the minimum and maximum AP and both average AP measures for all multimodal models. Likewise, Figure 7 contains the corresponding box and whisker plots. The RGB + Audio + Frame architecture presented in Wang et al. (2019) is the worst performing model. Our proposed hybrid approach is the best performing system and produces significantly different predictions to all models except for the RGB + Audio model fusion model, Table 5. This may be due to the similarities in RGB and Flow prediction.

Table 5 also details the A-measure calculations for this experiment. We see a

Table 4: Results for Experiment Two. Minimum Average Precision, Maximum Average Precision, Mean Average Precision, and Medium Average Precision are shown for 6 multimodal models. [2]Wang et al. (2019).

| Modalities | Fusion | Min. AP | Max. AP | $\bar{m}AP$ | $\tilde{m}AP$ |
|---|---|---|---|---|---|
| RGB+Flow | Model | 0.064 | 0.514 | 0.375 | 0.379 |
| RGB+Audio | Model | **0.081** | **0.654** | 0.444 | **0.456** |
| RGB+Flow+Audio | Model | 0.075 | 0.616 | 0.436 | 0.431 |
| RGB+Audio+Frame[2] | Model | 0.025 | 0.647 | 0.264 | 0.280 |
| RGB+Flow | Feature | 0.070 | 0.567 | 0.418 | 0.424 |
| RGB+Flow+Audio | Hybrid | 0.072 | 0.642 | **0.462** | 0.452 |

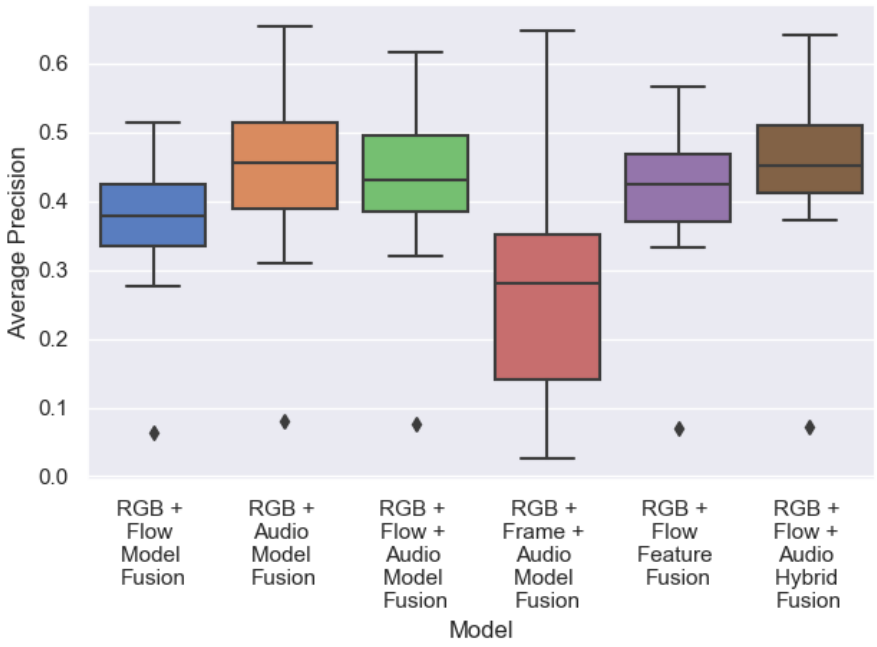



Figure 7: Box and Whisker plots for all models evaluated in Experiment Two.

Table 5: Statistical testing results for Experiment Two. The upper right triangle details Vargha Delaney A measure. Values less than 0.5 indicate column outperforms row and values more than 0.5 indicate row outperforms column. † denotes a small effect size ($a < 0.44 \vee a > 0.56$). ‡ denote a medium effect size ($a < 0.36 \vee a > 0.64$). $\diamond$ and bold denotes a large effect size ($a < 0.29 \vee a > 0.71$). The lower left triangle details p-values calculated via a Wilcoxon signed-rank test. Bold denotes significant values, i.e $p < 0.05$.

| | | Model | | | | Feature | Hybrid |
|---|---|---|---|---|---|---|---|
| | | RFl | RA | RFlA | RFlFr | RFl | RFlA |
| Model | RFl | - | $0.31^{\ddagger}$ | $0.32^{\ddagger}$ | $\mathbf{0.75}^{\diamond}$ | $0.35^{\ddagger}$ | $\mathbf{0.24}^{\diamond}$ |
| | RA | $\mathbf{1 \times 10^{-5}}$ | - | 0.53 | $\mathbf{0.84}^{\diamond}$ | $0.58^{\dagger}$ | 0.46 |
| | RFlA | $\mathbf{2 \times 10^{-6}}$ | $3 \times 10^{-1}$ | - | $\mathbf{0.84}^{\diamond}$ | 0.56 | $0.40^{\dagger}$ |
| | RFlFr | $1 \times 10^{-2}$ | $\mathbf{8 \times 10^{-5}}$ | $\mathbf{3 \times 10^{-4}}$ | - | $\mathbf{0.17}^{\diamond}$ | $\mathbf{0.12}^{\diamond}$ |
| Feature | RFl | $\mathbf{9 \times 10^{-6}}$ | $8 \times 10^{-2}$ | $5 \times 10^{-2}$ | $\mathbf{6 \times 10^{-4}}$ | - | $0.37^{\dagger}$ |
| Hybrid | RFlA | $\mathbf{2 \times 10^{-6}}$ | $9 \times 10^{-2}$ | $\mathbf{7 \times 10^{-4}}$ | $\mathbf{1 \times 10^{-5}}$ | $\mathbf{4 \times 10^{-6}}$ | - |

large effect size when comparing our Hybrid approach to the RGB+Flow model fusion model and a medium effect compared to the RGB+Flow feature fusion model. This large effect size indicates the performance benefits of using hybrid fusion across audio and visual modalities compared to visual data. Interestingly we see only little effect compared to RGB+Audio and RGB+Flow+Audio model fusion models, perhaps given that these all include the Audio domain, which appears to be the single strongest indicator of a highlight. The medium effect over the RGB+Flow+Audio model fusion model demonstrates the value of hybrid fusion over only using model fusion. The RGB+Flow+Frame model., i.e. Wang et al. (2019), performs very poorly in this test, likely due to the Frame model being a poor indicator of highlights, thus causing prediction noise.

### *5.3. Experiment Three - Time-Series Smoothing Convolutions*

For this experiment, smoothing convolutions were applied to our hybrid fusion model. All smoothing kernels are experimented with and compared to the model's performance without smoothing. Once again, key results are presented in Table 6 and Figure 8. While the 'Power of 2' smoothing kernel works poorly, both 'Linear' and 'Gaussian' outperformed the non-smoothing model. Furthermore, the 'Gaussian' smoothing performed better than the 'Linear' model.

Finally, all models produce significantly different predictions except for the Power of 2 model compared to No Smoothing. Gaussian smoothing is the only technique that produced an improvement with a notable effect size compared to not using smoothing. The full Wilcoxon signed-rank test and Vargha Delaney A measure results are presented in Table 7.

### *5.4. Experiment Four - Comparison of Ranking Network vs Binary Classifier*

Throughout the first three experiments all models use a ranking network to make predictions. However, this experiment compares ranking to binary classification, both using hybrid fusion and Gaussian smoothing. Two classifiers are presented, one which includes the weighting constant proposed by Elkan & Noto (2008) and one which does not. We also experiment with setting the

Table 6: Results for Experiment Three. Minimum Average Precision, Maximum Average Precision, Mean Average Precision, and Medium Average Precision are shown for the hybrid model with various smoothing kernels applied.

| Smoothing | Min. AP | Max. AP | $\bar{m}AP$ | $\tilde{m}AP$ |
|---|---|---|---|---|
| No Smoothing | 0.072 | 0.642 | 0.462 | 0.452 |
| Linear | **0.087** | 0.719 | 0.487 | 0.484 |
| Gaussian | 0.079 | **0.726** | **0.511** | **0.503** |
| Power of 2 | 0.073 | 0.642 | 0.442 | 0.438 |

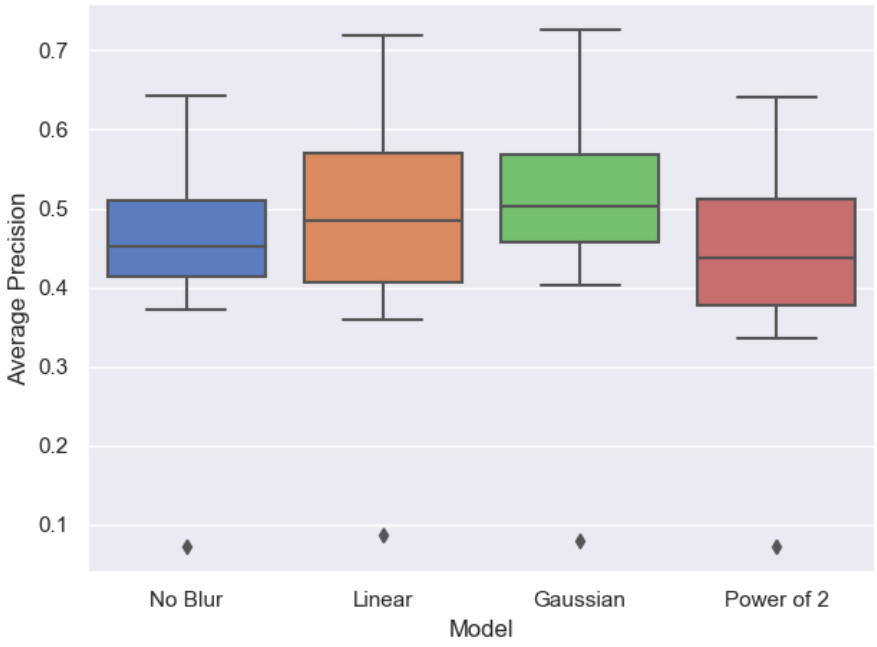



Figure 8: Box and Whisker plots for all models evaluated in Experiment Three.

Table 7: Statistical testing results for Experiment Three. The upper right triangle details Vargha Delaney A measure. Values less than 0.5 indicate column outperforms row and values more than 0.5 indicate row outperforms column. † denotes a small effect size ($a < 0.44 \vee a > 0.56$). ‡ denote a medium effect size ($a < 0.36 \vee a > 0.64$). $\diamond$ and bold denotes a large effect size ($a < 0.29 \vee a > 0.71$). The lower left triangle details p-values calculated via a Wilcoxon signed-rank test. Bold denotes significant values, i.e $p < 0.05$.

| | No Smoothing | Linear | Gaussian | Power of 2 |
|---|---|---|---|---|
| No Smoothing | - | 0.44 | $0.34^{\ddagger}$ | $0.58^{\ddagger}$ |
| Linear | $\mathbf{3 \times 10^{-3}}$ | - | $0.42^{\ddagger}$ | $0.63^{\ddagger}$ |
| Gaussian | $\mathbf{2 \times 10^{-6}}$ | $\mathbf{2 \times 10^{-3}}$ | - | $\mathbf{0.71}^{\diamond}$ |
| Power of 2 | $2 \times 10^{-3}$ | $\mathbf{2 \times 10^{-6}}$ | $\mathbf{2 \times 10^{-6}}$ | - |

Table 8: Results for Experiment Four. Minimum Average Precision, Maximum Average Precision, Mean Average Precision, and Medium Average Precision are shown for the hybrid model alongside a binary classifier.

| Decision Model | Min. AP | Max. AP | $\bar{m}AP$ | $\tilde{m}AP$ |
|---|---|---|---|---|
| Classifier | 0.046 | 0.719 | 0.499 | **0.542** |
| Ranker | **0.079** | **0.726** | **0.511** | 0.503 |

detection threshold for the models rather than relying solely on $\bar{m}AP$. Selecting a suitable threshold is vital for deploying such a model in a real-world situation. For the classifier, this selection is trivial because the model is trained with an exact threshold value, in our case, 0.5. In fact, the weighting constant requires the threshold to be 0.5 for it to function correctly. Selecting a threshold for the ranking model is less straightforward and needs to be determined heuristically. The ranking model has been trained under the assumption that 1 in 8 pairings are invalid, i.e. one in eight samples (12.5%) in the broadcast dataset are highlights. Following this, we apply a dynamic threshold where the 12.5% of samples with the highest ranks are selected as highlights.

Once again, we present the minimum AP, maximum AP, and both average

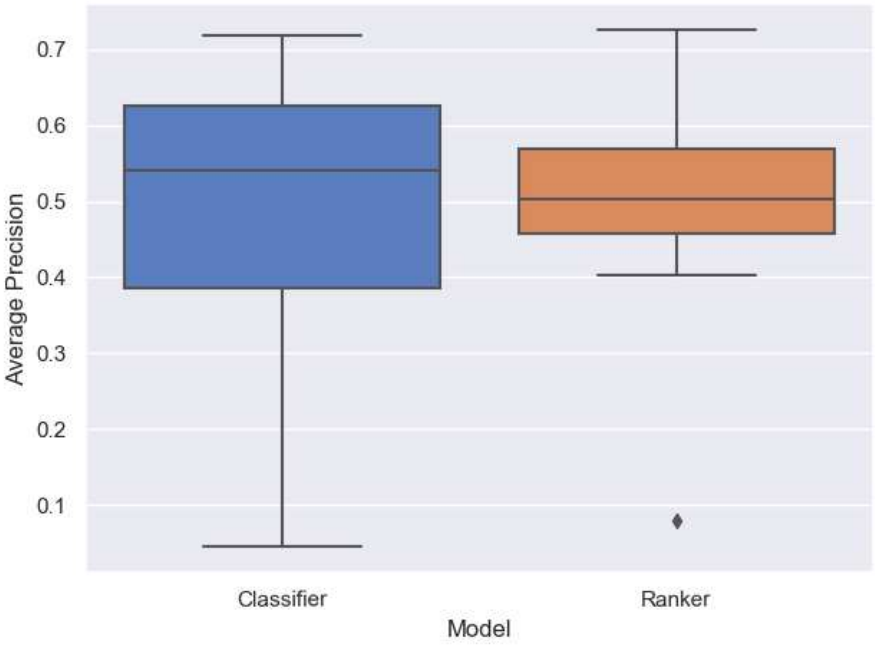



Figure 9: Box and Whisker plots for all models evaluated in Experiment Four.

Table 9: Accuracy, Precision, Recall and F1 score for Experiment Four.

| Decision Model | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|
| Classifier | 0.878 | 0.411 | 0.295 | 0.343 |
| Weighted Classifier | **0.893** | **0.559** | 0.074 | 0.131 |
| Ranker | 0.888 | 0.488 | **0.559** | **0.521** |

AP measures for the hybrid model and a binary classifier model in Table 8 and Figure 9. Only one set of scores for the binary classifier is presented because the weighting to account for uncertain labels does not affect average precision calculations.

When comparing thresholded models output, calculating the $\bar{m}AP$ becomes impossible, but more traditional metrics such as accuracy, precision, recall and F1 Score are useful. These are shown in Table 9. The metrics presented in this table rely on a set decision boundary. Therefore models which require a decision boundary to be set heuristically, i.e. ranking models, may appear worse as a poorly chosen threshold may artificially affect results. However, deploying a ranking model in a real-world application would require setting this threshold. Furthermore, for real-world settings, precision and recall based metrics are crucial. Measuring accuracy is not particularly useful for our task because there is a considerable class imbalance in our data, and accuracy does not weigh one class as more important than the other. In our data, many more video segments are non-highlights, but we are interested in detecting moments in the minority highlight class. As such, accuracy may lead to models which appear successful but, in practice, rarely detect the exciting moments and instead over classify the majority non-highlight class. Precision, recall and F1 are concerned only with the positive class, i.e. highlights, and therefore are more suitable. While precision measures are helpful for highlight detection evaluation, as evidenced by their prevalence in literature, recall is also valuable as it ensures that all of the interesting moments are included. Therefore the F1 Score, the harmonic mean of the two, is a reasonable metric for evaluating the ultimate performance of a model.

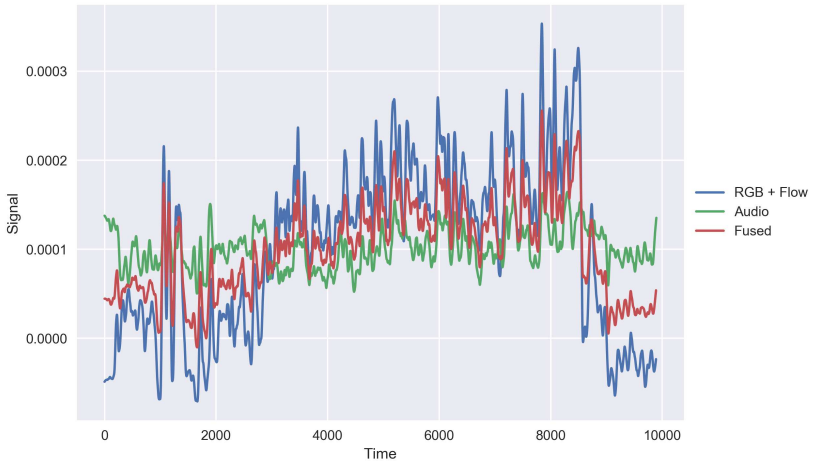
Additionally, we observe that the highlight signal produced by the two models is very different. Figure 10 shows the output for an indicative game for both the weighted classifier and ranking model. The signals from each half of the model (i.e. RGB + Flow and Audio) alongside the fused weighted signal are shown. Notice how the output from the models looks very different. Furthermore, we observed that the unweighted signals appeared to be more correlated

in the ranking model. We calculated the Pearson's correlation coefficient for these signals for all videos in the test set to confirm this. We found that, on average, the ranking networks decision models are more often correlated. The median correlation was 0.434 for the rank model and 0.291 for the classifier.

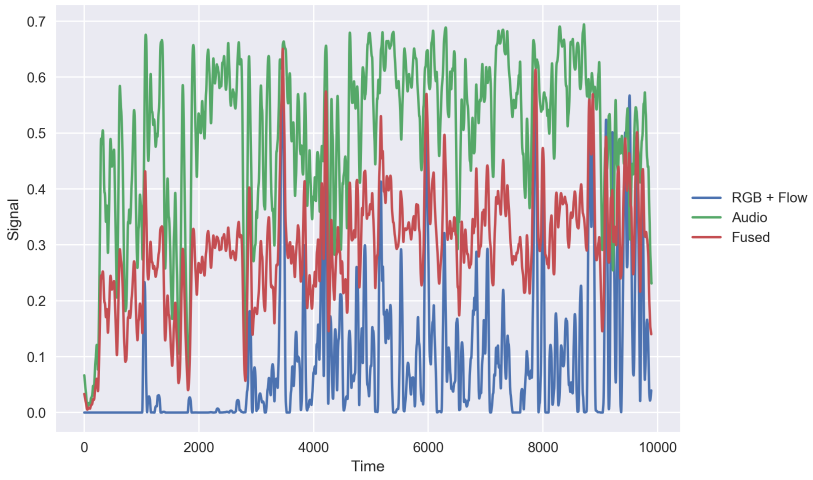
## 6. Discussion

These results show several important discoveries. Firstly, the proposed hybrid fusion model outperforms the current state-of-the-art. This performance improvement is evidenced both with and without the time-series smoothing. The performance benefit of the smoothing is also an important finding. It shows that modelling our video data as part of a time series, rather than merely individual 2.56 second segments, significantly improves performance. This is further evidence that context is essential for modelling the highlight moments in a broadcast.

We find that, somewhat surprisingly, audio appears to perform the best of all of the single modalities. This is an exciting result because we would expect that the visual data, especially in-game, would be more consistent between training examples, and thus, the ranker would be able to better model which moments are interesting. However, sports commentators are trained to make the audience 'feel' when moments are exciting by delivering their commentary with high arousal. The high levels of arousal in commentator speech appears to be a powerful indicator of highlight moments, mirroring Sun et al. (2010). Furthermore, we notice a broader range of results ($MaxAP - MinAP$) for the audio modality compared to both visual domains. Furthermore, the confidence value determined for the weighted classifier model is lower for the audio model. The findings indicate that audio is a potent indicator but that it also appears in some ways less reliable than visual data, perhaps due to the natural noise present in commentator speech, e.g. they may get animated when arguing about the effectiveness of a particular play, which would falsely appear to be an indicator of a highlight. Less surprising is that multimodal models generally outperform




(a)




(b)

Figure 10: Time-series highlight signal for the Ranker (a) and Classifier (b) models from Experiment Four on a single video in the test set.

unimodal models. This is expected as in other domains, e.g. affective computing, multimodal modelling is effective, but it is still helpful to demonstrate this empirically.

We see that the method from Xiong et al. (2019) generally performs worse on this domain than in the original paper. However, the method from Wang et al. (2019) performs better on our dataset. It is valuable to point out that Xiong et al. (2019) was evaluated on significantly different data, human action highlight recognition, but Wang et al. (2019) was evaluated on similar data, esports coverage of a similar game to *League of Legends*. We expected that this problem would be complex, first because, as discussed above, context is vital and secondly because the visual range of gameplay is small, e.g. the difference between a highlight and non-highlight frame is smaller than in human actions. This might also explain the generally poor performance of the single-frame model.

There does appear to be one video in the dataset where all models perform poorly, evidenced by the low performing outlier in most box plots and Figure 11, which shows the precision-recall curves used to calculate the average precision for test set videos using hybrid fusion with smoothing. Upon inspection, this particular video has much non-game content, especially highlights from previous matches and replays of celebrations with excited commentators speaking over the top. This extra content is unusual among the test set and explains the poor performance. A human editor would easily understand that these moments are outside of the game being played, but our models currently cannot.

Table 10 details the average execution time for various parts of the system. These times were calculated using the test set and the Nvidia 2080ti GPU equipped machine detailed in Section 5. Feature extraction directly from video data is the most computationally expensive part of the process. This is understandable due to the high complexity of video data. By comparison, calculating the initial segment decisions is very fast, with all models taking around 0.004 seconds. Therefore, even our most complex hybrid fusion model is capable of 'real-time' prediction, i.e. processing faster than data is generated, utilising consumer-grade hardware. However, such prediction is currently infea-

sible due to the need for thresholding and the benefits of smoothing, which are both applied to the entire broadcast. That said, these processes are extremely fast computationally, taking approximately 0.5 seconds per video, although this time varies slightly depending on video length. Therefore it would be possible to process segments in real-time and then apply the smoothing and thresholding techniques when the broadcast ends and thus generate and distribute the highlight video extremely quickly.

Calculating the memory usage of the models presented in this work is challenging due to the choice of implementation library for the ranking decision models. Keras has no native method for accurate profiling memory usage. However, approximate solutions exist, e.g. the one used for Table 1. Comparing weights also determines the relative complexity of models, if not the total memory cost. Table 1 details the number of weights for each ranking model used in this work. From this, it is clear that the Frame only model is extremely large, $\sim 42$ million weights, compared to all other models. This is approximately 15 times larger than the next largest ranking model in terms of weights, requiring nearly five times more memory during training and over 15 times more during inference. Thus Wang et al. (2019) has a much larger memory footprint compared to other approaches. Not only does the larger model size increase run-time and memory footprint, but it is also likely contributed to the poor performance of the frame model due to over-fitting.

To demonstrate the processing time differences between feature extraction and ranking, the RGB and Flow feature extraction models have a combined $\sim 25$ million weights, and each require over 1300MB of memory during inference while the RGB + Flow feature fusion decision model requires just 2.7MB. Therefore it is evident that the decision part of the network is extremely lightweight compared to feature extraction. Thus, while our three modality hybrid fusion model has approximately six times weights and nearly five times the memory requirements compared to Xiong et al. (2019) it still operates quickly and adds little total operating time to the system. However, utilising this approach provides significant performance benefits, which we argue are worth the additional time

Table 10: Average Execution Time per sample in seconds for both Feature Extraction (including preprocessing) and the Segment Decision Process.

| Component | Average Time (seconds) |
|---|---|
| RGB Feature Extraction | 0.215 |
| Flow Feature Extraction | 0.298 |
| RGB + Flow Feature Extraction | 0.495 |
| Audio Feature Extraction | 0.013 |
| Segment Decision Process | 0.004 |

and memory requirements. Additionally, we observe that all decision models other than the Frame model, owing to its size, are lightweight enough to train quickly, with each model taking just one to two hours to train on consumer hardware.

Finally, it is worthwhile discussing the implications of Experiment Four, specifically Table 9. While we see that the performance between the two models is similar when calculating $\bar{m}AP$ and $\tilde{m}AP$, when a threshold is selected, the ranking model performs very well, despite the threshold being chosen heuristically, compared to the, in theory, 'perfectly' selected threshold for the binary classifiers, i.e. the threshold with which the model was trained. The ranking model has similar precision to the weighted classifier and is better than the unweighted classifier. However, it has far better recall and thus a better F1 score. This suggests that even with a heuristically decided threshold, a ranking approach is more suitable for deployment than the corresponding classifier model. Furthermore, we see the classification noisy label weighting effect as it is the most precise model. However, it seriously affects the recall of the model. This results in an ineffectual model for our domain. The highlights are intended to entertain but also present an abridged narrative of the match. Such a low recall score would, in practice, lead to a model which fails in this respect.

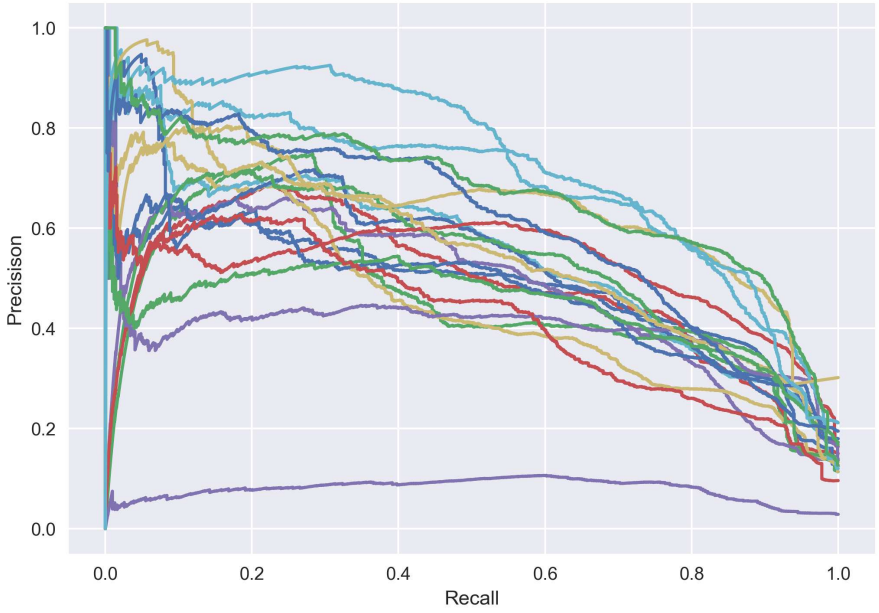



Figure 11: Precision Recall curves for all videos in the dataset with the hybrid fusion model with gaussian smoothing.

## 7. Conclusion

This work presents a multimodal hybrid fusion model for automatic highlight detection in esports broadcasts. It shows significant improvement over existing techniques, in some cases achieving a $\bar{m}AP$ nearly twice as high as existing state-of-the-art multimodal approaches in this domain. In doing so, we also show the effectiveness of multimodal models compared to using a single modality. Furthermore, we propose applying a smoothing kernel convolution to the output signal of the model, which further significantly improves results. Lastly, we examined the assumption that ranking approaches would outperform a corresponding classification approach. While classification performance is comparable when averaging across all potential thresholds, we see that when we select even a heuristically chosen threshold, the ranking model outperforms the classifier model on critical metrics such as F1 Score. While this approach has some limitations, e.g. context is only modelled crudely, it provides a solid foundation to operationalise automatic highlight detection. The results are an improvement over other state-of-the-art models, and additionally, the model can be run quickly, just a few minutes per video on consumer-grade hardware, which would allow esports broadcasters to quickly edit and release highlight videos after a match has finished.

## 8. Future Work

The hybrid fusion model performs well compared to state-of-the-art models. However, like them, it selects highlights based only on short segments and does not explicitly model events outside of these segments, which could provide additional context. While applying smoothing does introduce consideration for the context of a given sample, this is not modelled in the original ranking model. It is challenging to model context in a weakly-supervised setting. While we have the context for the broadcast dataset, as it is the original unedited broadcast, this is not the case for the highlight dataset because it has been edited. Careful consideration is required to best include context data into these models as doing

so is likely to yield positive results. Additionally, rather than using pre-trained feature extractors, it is also possible that using self-supervised learning may yield performance benefits. For instance, a training scheme similar to the successful 'jigsaw' scheme, but using temporal stacks of frames would potentially result in a better set of features and improved short-term context modelling.

While text chat data was outside of the scope of this work, it does appear to be an exciting avenue of investigation. Therefore, should appropriate techniques for highlight detection from chat data emerge, it would be pertinent to develop models which leveraged audio, visual and textual information for highlight classification.

Finally, it is arguable that these models create highlights as a pastiche of the video editors who made the original highlight videos in the training data. It would be potentially valuable to explore highlight detection through the lens of computation creativity, researching algorithms that learn to generate highlight videos with unique styles. For instance, it may be possible to utilise the active community of League of Legends fans on social media sites to improve our models by posting highlight clips from our model to social media channels and judging the responses it receives as a form of fine-tuning. This may also lead to personalised highlight systems where viewers are shown a highlight video conditioned on the kind of content the viewer has historically enjoyed. Alternatively, techniques such as the ones presented in this paper may serve as a jumping-off point for human-in-the-loop co-creativity systems where the system aides a human editor, rather than replacing them.

### Author Statement

**Charles Ringer**: Conceptualisation, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualisation, Formal analysis. **Mihalis A. Nicolaou**: Conceptualisation, Writing - Review & Editing, Supervision, Project administration. **James Alfred Walker**: Conceptualization, Resources, Writ-

ing - Review & Editing, Supervision, Project administration

**Acknowledgements**

The authors would like to thank Peter York and Daniel Hernandez for their helpful and insightful comments.

Funding: This work is funded by the EPSRC Centre for Doctoral Training in Intelligent Games and Game Intelligence (IGGI), EP/L015846/1 and the Digital Creativity Labs funded by EPSRC/AHRC/Innovate UK, EP/M023265/1.

**References**

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., & Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Alwassel, H., Mahajan, D., Korbar, B., Torresani, L., Ghanem, B., & Tran, D. (2020). Self-supervised learning by cross-modal audio-video clustering. In *Advances in Neural Information Processing Systems 34* (pp. 1–18). Curran Associates, Inc.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, .

Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4724–4733).

Chiu, C. Y., Lin, P. C., Li, S. Y., Tsai, T. H., & Tsai, Y. L. (2012). Tagging webcast text in baseball videos by video segmentation and text alignment. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*, 999–1013. doi:10.1109/TCSVT.2012.2189478.

Chollet, F. et al. (2015). Keras. `https://keras.io`.

Chu, W. T., & Chou, Y. C. (2015). Event detection and highlight detection of broadcasted game videos. *HCMC 2015 - Proceedings of the 2nd Workshop on Computational Models of Social Interactions: Human-Computer-Media Communication, co-located with ACM MM 2015*, (pp. 1–8). doi:10.1145/2810397.2810398.

Chu, W. T., & Chou, Y. C. (2017). On broadcasted game video analysis: event detection, highlight detection, and highlight forecast. *Multimedia Tools and Applications*, *76*, 9735–9758. doi:10.1007/s11042-016-3577-x.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). Ieee.

Elkan, C., & Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD '08 (p. 213–220). New York, NY, USA: Association for Computing Machinery. doi:10.1145/1401890.1401920.

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In J. Bigun, & T. Gustavsson (Eds.), *Image Analysis* (pp. 363–370). Berlin, Heidelberg: Springer Berlin Heidelberg.

Feichtenhofer, C., Pinz, A., & Zisserman, A. (2016). Convolutional two-stream network fusion for video action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Giannakopoulos, T., Makris, A., Kosmopoulos, D., Perantonis, S., & Theodoridis, S. (2010). Audio-visual fusion for detecting violent scenes in videos. In S. Konstantopoulos, S. Perantonis, V. Karkaletsis, C. D. Spyropoulos, & G. Vouros (Eds.), *Artificial Intelligence: Theories, Models and Applications* (pp. 91–100). Berlin, Heidelberg: Springer Berlin Heidelberg.

Gygli, M., Song, Y., & Cao, L. (2016). Video2gif: Automatic generation of animated gifs from video. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1001–1009). doi:10.1109/CVPR.2016.114.

Han, H.-K., Huang, Y.-C., & Chen, C. C. (2019). A deep learning model for extracting live streaming video highlights using audience messages. In *Proceedings of the 2019 2nd Artificial Intelligence and Cloud Computing Conference* AICCC 2019 (p. 75–81). New York, NY, USA: Association for Computing Machinery. doi:10.1145/3375959.3375965.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'ıo, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*, 357–362. doi:10.1038/s41586-020-2649-2.

Ilse, M., Tomczak, J., & Welling, M. (2018). Attention-based deep multiple instance learning. In J. Dy, & A. Krause (Eds.), *Proceedings of the 35th International Conference on Machine Learning* (pp. 2127–2136). PMLR volume 80 of *Proceedings of Machine Learning Research*. URL: `https://proceedings.mlr.press/v80/ilse18a.html`.

Jia, Y., Zhang, Y., Weiss, R., Wang, Q., Shen, J., Ren, F., Chen, z., Nguyen, P., Pang, R., Lopez Moreno, I., & Wu, Y. (2018). Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances*

*in Neural Information Processing Systems 31* (pp. 4480–4490). Curran Associates, Inc.

Kahou, S. E., Bouthillier, X., Lamblin, P., Gulcehre, C., Michalski, V., Konda, K., Jean, S., Froumenty, P., Dauphin, Y., Boulanger-Lewandowski, N., Chandias Ferrari, R., Mirza, M., Warde-Farley, D., Courville, A., Vincent, P., Memisevic, R., Pal, C., & Bengio, Y. (2016). Emonets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, *10*, 99–111.

Katsaggelos, A. K., Bahaadini, S., & Molina, R. (2015). Audiovisual Fusion: Challenges and New Approaches. *Proceedings of the IEEE*, *103*, 1635–1653.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., & Zisserman, A. (2017). The kinetics human action video dataset. *CoRR*, *abs/1705.06950*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 1097–1105). Curran Associates, Inc.

Lai, K., Yu, F. X., Chen, M., & Chang, S. (2014). Video event detection by inferring temporal instance labels. In *2014 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2251–2258).

Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., & Li, L. (2017). Learning from noisy labels with distillation. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 1928–1936). Los Alamitos, CA, USA: IEEE Computer Society. URL: `https://doi.ieeecomputersociety.org/10.1109/ICCV.2017.211`. doi:10.1109/ICCV.2017.211.

Liaw, C., & Dai, B. (2020). Live stream highlight detection using chat messages.

In *2020 21st IEEE International Conference on Mobile Data Management (MDM)* (pp. 328–332).

Liu, T., & Tao, D. (2016). Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*, 447–461. doi:10.1109/TPAMI.2015.2456899.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*. volume 8.

Mendi, E., Clemente, H. B., & Bayrak, C. (2013). Sports video summarization based on motion analysis. *Computers & Electrical Engineering*, *39*, 790 – 796. doi:https://doi.org/10.1016/j.compeleceng.2012.11.020. Special issue on Image and Video Processing Special issue on Recent Trends in Communications and Signal Processing.

Moodley, T., & van der Haar, D. (2020). Scene recognition using alexnet to recognize significant events within cricket game footage. In L. J. Chmielewski, R. Kozera, & A. Orłowski (Eds.), *Computer Vision and Graphics* (pp. 98–109). Cham: Springer International Publishing.

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., & Tewari, A. (2013). Learning with noisy labels. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 26* (pp. 1196–1204). Curran Associates, Inc.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., & Ng, A. Y. (2011). Multimodal deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning* ICML'11 (pp. 689–696). USA: Omnipress.

Nguyen, N., & Yoshitaka, A. (2014). Soccer video summarization based on cinematography and motion analysis. *Multimedia Signal Processing (MMSP)*, (pp. 1–6).

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Poria, S., Cambria, E., Bajpai, R., & Hussain, A. (2017). A review of affective computing: From unimodal analysis to multimodal fusion. *Information Fusion*, *37*, 98–125.

Reed, S. E., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., & Rabinovich, A. (2015). Training deep neural networks on noisy labels with bootstrapping. In Y. Bengio, & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*. URL: `http://arxiv.org/abs/1412.6596`.

Ren, R., Jose, J., & Yin, H. (2007). Affective sports highlight detection. *European Signal Processing Conference*, (pp. 728–732).

Ringer, C., & Nicolaou, M. A. (2018). Deep unsupervised multi-view detection of video game stream highlights. In *Proceedings of the 13th International Conference on the Foundations of Digital Games* FDG '18 (pp. 15:1–15:6). New York, NY, USA: ACM.

Ringer, C., Walker, J. A., & Nicolaou, M. A. (2019). Multimodal joint emotion and game context recognition in league of legends livestreams. In *2019 IEEE Conference on Games*. IEEE.

Shah, M., Chakrabarti, C., & Spanias, A. (2014). A multi-modal approach to emotion recognition using undirected topic models. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)* (pp. 754–757).

Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*.

Spijkerman, R., & van der Haar, D. (2020). Video footage highlight detection in formula 1 through vehicle recognition with faster r-cnn trained on game footage. In L. J. Chmielewski, R. Kozera, & A. Orłowski (Eds.), *Computer Vision and Graphics* (pp. 176–187). Cham: Springer International Publishing.

Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., & Fergus, R. (2015). Training convolutional networks with noisy labels.

Sun, M., Farhadi, A., & Seitz, S. (2014). Ranking domain-specific highlights by analyzing edited videos. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014* (pp. 787–802). Cham: Springer International Publishing.

Sun, Y., Ou, Z., Hu, W., & Zhang, Y. (2010). Excited commentator speech detection with unsupervised model adaptation for soccer highlight extraction. *ICALIP 2010 - 2010 International Conference on Audio, Language and Image Processing, Proceedings*, (pp. 747–751). doi:10.1109/ICALIP.2010.5685077.

Vargha, A., & Delaney, H. D. (2000). A critique and improvement of the "cl" common language effect size statistics of mcgraw and wong. *Journal of Educational and Behavioral Statistics*, *25*, 101–132. URL: `http://www.jstor.org/stable/1165329`.

Wang, L., Sun, Z., Yao, W., Zhan, H., & Zhu, C. (2019). Unsupervised multi-stream highlight detection for the game "honor of kings". *CoRR*, *abs/1910.06189*.

Wöllmer, M., Kaiser, M., Eyben, F., Schuller, B., & Rigoll, G. (2013). Lstm-modeling of continuous emotions in an audiovisual affect recognition framework. *Image and Vision Computing*, *31*, 153 – 163. Affect Analysis In Continuous Input.

Xiong, B., Kalantidis, Y., Ghadiyaram, D., & Grauman, K. (2019). Less is more: Learning highlight detection from video duration. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1258–1267). doi:10.1109/CVPR.2019.00135.

Xu, D., Ricci, E., Yan, Y., Song, J., & Sebe, N. (2015). Learning deep representations of appearance and motion for anomalous event detection. In X. Xie, M. W. Jones, & G. K. L. Tam (Eds.), *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015* (pp. 8.1–8.12). BMVA Press. URL: `https://doi.org/10.5244/C.29.8`. doi:10.5244/C.29.8.

Xu, H., & Chua, T.-S. (2006). Fusion of AV features and external information sources for event detection in team sports video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *2*, 44–67. doi:10.1145/1126004.1126007.

Yao, T., Mei, T., & Rui, Y. (2016). Highlight detection with pairwise deep ranking for first-person video summarization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 982–990).

Yu, F. X., Liu, D., Kumar, S., Jebara, T., & Chang, S.-F. (2013). $\propto$ svm for learning with label proportions. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28* ICML'13 (p. III–504–III–512). JMLR.org.

Zeng, Z., Pantic, M., Roisman, G. I., & Huang, T. S. (2009). A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*, 39–58. doi:10.1109/TPAMI.2008.52.

Zhu, G., Huang, Q., Xu, C., Xing, L., Gao, W., & Yao, H. (2007). Human behavior analysis for highlight ranking in broadcast racket sports video. *IEEE Transactions on Multimedia*, *9*, 1167–1182.