Hilait: Automatic Highlighting System Leveraging Facial, Audio, Text Sentiment AI

Tony Xia
tonyx717@stanford.edu

Danica Xiong
daxiong@stanford.edu

Abstract

In recent years, the popularity of live streaming platforms for gaming, such as Twitch, YouTube Gaming, and Facebook Gaming, has surged exponentially, establishing a burgeoning industry around live video game content. Within this landscape, the ability to effectively curate and highlight compelling moments from extensive streams has become increasingly valuable for content creators and fans. However, manual curation of highlights is time-consuming, relying heavily on human judgment digging through hundreds of hours of content. As far as we know, an automatic highlighting system does not currently exist. We propose a multimodal AI system that collects and analyzes visual, textual, and audio information from live streams and automatically proposes interesting timestamps. The source code can be found at https://github.com/danicax/AutoHilAIt

1. Introduction

1.1. Problem Statement

With the growing popularity of live streaming, the need to create highlights also increased. Streamers and content creators often struggle with efficiently pinpointing and showcasing key moments from extensive video footage. Manually searching through lengthy videos to find exciting clips is both time-consuming and labor-intensive. This challenge is pronounced for small streamers who may not have the resources to hire editors, limiting their capacity to create engaging short-form content for platforms like TikTok and YouTube.

To solve this issue, we developed Hilait, an Automatic Highlighting System that extracts video, audio, facial expressions, chat data, as well as API events for automatic clip recommendation. Our goal is to develop a comprehensive solution that alleviates the workload for streamers and editors while ensuring the quality and relevance of the generated highlights. Our core constraints are:

- **Ease of Use**: The system must be user-friendly, allowing streamers to easily integrate it into their workflows without extensive technical knowledge or setup.
- **High Clip Quality**: The tool must consistently produce high-quality clips that capture the most engaging and relevant moments. Similar to what an editor would choose for their video.
- **Time Efficiency**: The system should significantly reduce the time streamers spend searching for and editing highlight clips. This includes being able to process video in a reasonable timeframe.

2. Related Work

2.1. Automatic Video Highlighting

Automatic video highlighting has been extensively investigated by various research groups. Yang et al. [12] employed an unsupervised recurrent autoencoder to extract notable moments from video content. However, their approach solely relies on visual data, neglecting the chat and audio components, which are crucial indicators, particularly in the context of livestreams. Conversely, Fu et al. and Ping et al. [3, 8] explored the use of synchronized chat reactions as indicators for video highlights, but their methodologies were limited to textual data, thereby overlooking significant multimodal information. To the best of our knowledge, no prior work has successfully developed a comprehensive multi-modal system for automatic video highlighting.

2.2. Facial Feature Extraction

Face detection and facial expression recognition are two big areas of focus in computer vision in the early days of deep learning. Thus, a large amount of work have been done to tackle this problem.

Face Detection Zhang et al. [13] implemented a deep cascaded multi-task framework for predicting facial landmarks and alignment in a coarse-to-fine manner. Deng et al. [2] introduced RetinaFace, which employs a single-stage approach for face detection and alignment. YOLOv8 [7], developed as an enhancement of the original YOLO model

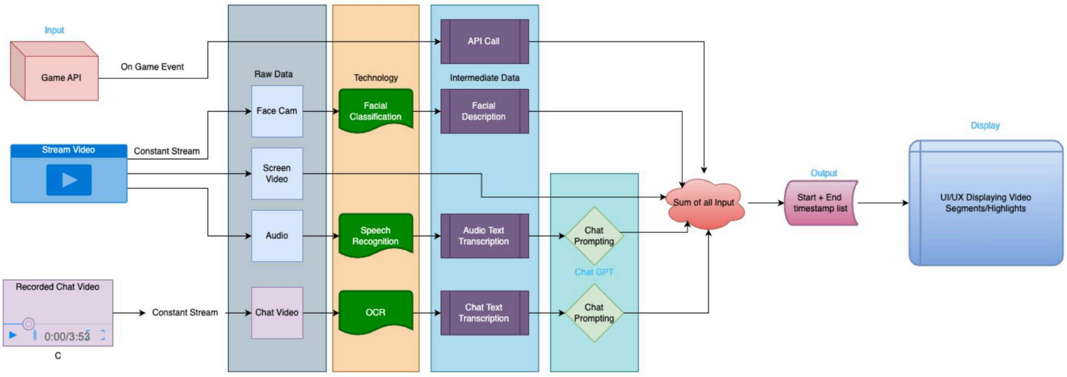

Figure 1. The following is our data processing pipeline. From the video Id, we get 3 inputs: Stream Video, Chat Video, and Game API. Eachis split into their raw data, ie. Face Cam, Screen Video, Audio, Chat video, be processed by their corresponding models, and then used as input to timestamp the video.

[9], offers significant improvements in speed and efficiency. In this study, we evaluated these models to determine the one most suitable for our objectives.

Facial Expression Recognition Goodfellow et al. [4] started the FER-2013 challenge. They constructed a dataset of 48×48 images of faces with emotions categorized into 7 classes: angry, disgust, fear, happy, neutral, sad, and surprise. Convolutional neural networks [6] have been the backbone of most of these approaches, as they showed exceptionally high efficacy in image classification with efficient training. After the initial success of CNN, He [5] proposed ResNet, utilizing residual connections to enable more efficient gradient flow, improving the performance of CNNs.

3. Approach

3.1. Overview

Our system processes a user-provided link to the Twitch VOD they wish to highlight. Using this link, the system retrieves the video recording and chat log via the Twitch API. Optical Character Recognition (OCR) is employed to identify the game being played, determining if a connection to a game API is necessary. After preliminary processing, the input data is organized into four modalities: video, audio, textual, and API-specific. Four parallel pipelines are then used to extract features from each modality. The final step is formulated as a regression problem, with the extracted features as inputs and a single excitement score as the output. This score is used to evaluate whether a clip should be recommended to the content creator. Figure 1 illustrates the entire system. For the purpose of this class, the focus of this report will mostly be on the facial expression recognition pipeline.

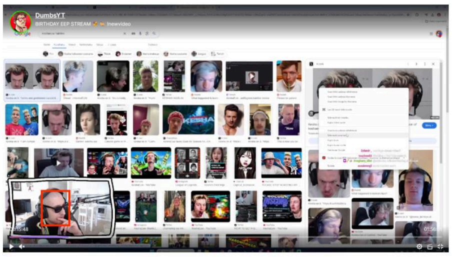

Figure 2. A challenging example of face recognition

3.2. Facial Expression Recognition Pipeline

The Facial Expression Recognition (FER) pipeline is designed to analyze the facial expressions of the streamer throughout the VOD. The goal is to classify the streamer's facial expressions at fixed time intervals to determine if their emotion is worth highlighting.

The input is video frames from the MP4 in 20 frame intervals. The pipeline consists of two steps:

1. **Face detection**: locating **all** faces that appear in the frame. Furthur filtering is done to eliminate faces that do not belong to the streamer.
2. **Facial expression classification**: identifying the dominant emotion based on the facial expression.

3.2.1 Facial Detection

To accurately recognize the streamer's face, we first employ YOLOv8 [7] to detect candidate faces within the entire stream window. The model outputs a list of detected faces. Next, using a pre-trained VGG network provided by DeepFace [10], we compute the similarity between each detected face and an identity image provided by the user. The face with the highest similarity score is selected and passed into the facial expression classifier. This ensures that the correct face is consistently identified throughout the video. Figure 2 shows the importance of such filtering.

3.2.2 Facial Emotion Classification

The facial expression classifier processes the selected face and outputs a score for each of the seven emotion labels defined in the FER 2013 dataset [4]. This score is used as an indicator of the streamer's emotional state, helping to identify moments in the video that are worth capturing. The emotions are classified into one of the seven categories: angry, disgust, fear, happy, neutral, sad, and surprise. Along with the dominant emotion, the model also outputs a score for each of the 7 emotions. This score is then passed to downstream components as a factor in determining whether a clip is worth recommending to the streamer.

To tackle this problem, we decided to test an off-the-shelf emotion classifier provided by DeepFace [11] against our own models trained on FER 2013. We experimented with 3 different model setups:

- CNN trained from scratch on FER 2013
- ResNet18 trained from scratch on FER 2013
- ResNet18 pretrained on ImageNet and finetuned on FER 2013

The training details and results are shown in later sections of the report.

4. Training and Evaluations

4.1. Training

We trained each of our vision model with 0.001 learning rate, Adam optimizer with CosineAnnealingLR learning rate scheduler. Since the training data consists of very small (48 × 48) images, combating overfitting was the top priority. Thus, we incorporated dropout with probability 0.5 in each of the networks we trained. Figure 3 shows the training and the validation accuracy over 20 epochs.

While all three models were able to achieve a very high training accuracy, they overfit to the training data too fast to be able to effectively generalize to the evaluation set. This is reflected in the validation accuracy over the training period. All three models plateaued at an accuracy lower than 0.64 despite our effort to mitigate overfitting with a learning rate scheduler.

4.2. Evaluations

To evaluate our system, we conducted both quantitative and qualitative assessments focusing on clip quality, the utility provided to the streamer, and a component-wise analysis to ensure each part contributes valid scores to the final output. For the sake of brevity, we omit evaluations of the other individual pipelines and concentrate solely on the Facial Expression Recognition (FER) pipeline and the overall system.

4.3. Evaluation of the FER Pipeline

Since our system's output is directly influenced by the quality of its individual components, it is essential to ensure that each pipeline produces results both quickly and accurately. We assessed the components of our facial expression recognition pipeline based on their speed, robustness, and accuracy.

We constructed a mini dataset, MiniFace, consisting of 61 random frames from streams of streamers that participated in our study. We manually labeled the facial expression on the streamers' faces.

4.3.1 Speed

We evaluated the speed and accuracy of different face detectors provided by DeepFace using MiniFace. We run the face detection model on 61 images with enforcing face detection on. This makes the model return with an error message if no face was discovered in the image. We discovered that YOLOV8 was able to detect the faces much faster than the other models (Table 2).

Method	Time (s)	Fails
opencv	49.67	25
mtcnn	45.64	0
fastmtcnn	4.58	0
retinaface	7.51	0
yolov8	**0.93**	0

Table 1. Performance comparison of different methods

Even though the sample size was small, we believed that YoloV8 demonstrated enough advantages over the other models, and thus we decided to use YoloV8 as our face detector.

4.3.2 Robustness

The robustness of the face detector is crucial to the pipeline's success. Given that streamers can display various content on their streams, it is essential to accurately distinguish the streamer's face from others.

Figure 2 illustrates our model's performance on frames containing multiple faces. As shown by the red box near the bottom left corner of the image, our pipeline successfully identified the correct face despite the streamer's best effort to confuse the model.

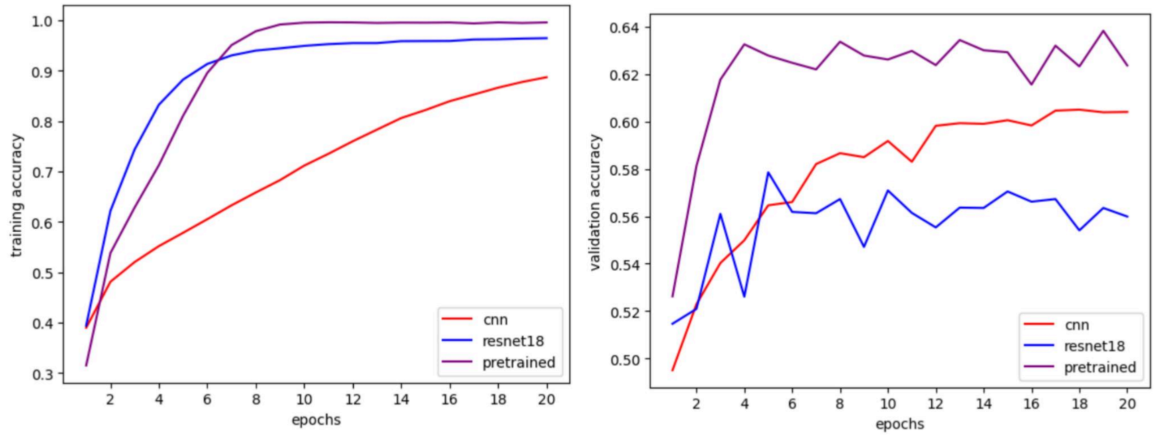

Figure 3. Training and Validation Accuracies

4.3.3 Accuracy

We conducted evaluations on several models: our self-trained CNN, ResNet18, a pre-trained and fine-tuned ResNet18, and the DeepFace facial expression classifier, using the FER2013 dataset. Although the hyperparameters are not yet perfectly tuned, the fine-tuned pre-trained ResNet18 achieved a validation accuracy of approximately 63

Subsequently, we assessed our FER pipeline using the MiniFace dataset. The best performance was achieved by the ResNet18 model, which was pre-trained on ImageNet [1] and fine-tuned on FER2013, attaining a 55% accuracy. In comparison, the CNN and DeepFace models each achieved 49% accuracy, while the ResNet18 trained from scratch correctly labeled only 45% of the samples.

Figure 4 displays the confusion matrices for each model when evaluated on MiniFace. The matrices reveal a tendency for all models to confuse neutral and sad faces. While pronounced features such as smiles and frowns are preserved, the subtler distinctions between sad and neutral expressions are often lost when the images are downscaled for input into the neural networks. This issue is currently mitigated by the fact that sadness is a less common emotion among streamers. As a temporary solution, we recategorized sad faces as neutral.

To address this problem in the future, we plan to train larger networks that can handle higher resolution inputs using more extensive facial expression datasets. By retaining more detailed information, we anticipate improved differentiation between sad and neutral faces by the classifiers.

4.4. Quantitative System Evaluation

The feedback on the quality of our generated clips was varied. To maintain privacy, the five streamers who provided feedback will be referred to as N, M, T, S, and E. Three of the five streamers (M, T, S) did not submit feedback through the rating forms, which may indicate a lack of interest or time constraints. Notably, none of the respondents provided specific details about the time required to view the clips or identify better alternative clips.

Method	**Accuracy**
CNN	49.2%
ResNet18	46.0%
Pretrained	**55.7%**
DeepFace	49.2%

Table 2. Performance comparison of different methods

High-Follower Streamers (T and M) The two streamers with the highest follower counts (2 million and 400k followers, respectively) expressed high satisfaction with our clips. They appreciated how our system effectively highlighted the exciting moments of the game, which aligned well with the type of content they aimed to share with their large audiences.

We analyzed the difference between the streamers' ratings and our system's ratings for the clips. While streamer M did not give a rating for each of the clips we suggested, they gave our system an overall rating of 9/10.

Streamer T filled out our form with the following ratings:

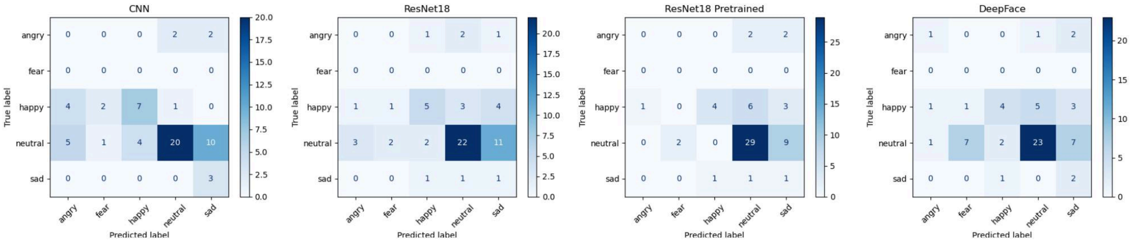

Figure 4. Confusion Matrices When Evaluated on MiniFace

Our Score	Streamer T's Rating
6	6
6	5
5	8
5	5
5	7.5
4	5
4	5.5
4	7
4	5
4	4

Streamer T gave an overall rating of highlight quality a 9/10 as well, and the difference in their rating vs our rating as 1.3 per clip, with a minimum difference of 0 and a maximum difference of 3.

High-Ranking Streamers (S and N): Conversely, the two streamers with the highest rankings (both top 200 players in North America, with approximately 100k and 200k followers each) were less satisfied with the generated clips. Streamer S noted that while the clips were exciting, they did not align with the educational content he typically produces, which focuses on in-depth gameplay strategies. He pointed out that a crucial clip was missed, where he executed a strategy for a full minute, resulting in a single kill. This oversight might be due to our interval system not accounting for clip contexts longer than a minute, as well as our scoring system potentially overemphasizing multiple kills and chat activity.

Streamer N felt that our system overemphasized moments where he "got loud," indicating that the combination of API and chat excitement indicators was overly sensitive. Interestingly, our system does not consider the loudness of the audio, only the transcribed audio and its sentiment.

Streamer S rated the overall clip quality at 3/10, while Streamer N gave a 5/10, with the following detailed ratings:

Our Score	Streamer N's Rating
10	10
8	0
8	2
7	2
7	5
6.5	3
6	0
6	0
6	7
6	5

This table shows an average difference of 3.25 with a minimum difference of 0 and a maximum difference of 8.

This feedback indicates that our system might have overvalued certain moments due to the convergence of multiple excitement signals, especially playing too much value on APIs and Chat reactions.

These mixed results reveal important areas for improvement. Different streamers have different content preferences, which suggests a need for more customizable and nuanced models. To better cater to diverse needs, we are considering developing separate models tailored to different types of content. This customization would enhance the relevance and quality of the highlights generated for each individual streamer.

4.5. Qualitative Clip Feedback

The most notable success was observed with a 14-hour VOD from streamer M, where 9 out of 10 generated clips were utilized in a highlight video by a streamer with 2 million followers. The sole error occurred when GPT erroneously identified the chat spamming "hello" at the start of the stream as a highly exciting moment. This feedback underscores the potential of our system while also highlighting areas that require improvement.

The most prevalent complaint from streamers pertained to the difficulty in viewing the generated clips. Initially, streamers were provided with a CSV file containing timestamps of their entire stream, ranked by score. However, they found it cumbersome to manually copy and paste the timestamps into their browser to view the clips. In response to this feedback, we developed hilait.com, a platform that automatically displays clips along with their respective timestamps. This website not only simplifies the viewing process but also prompts streamers to rate each clip.

The most significant quality-related feedback came from Streamers N and S, who thought that our system failed to capture nuanced conversations and educational content. Although our system is effective at identifying ”hype” clips, it struggles with subtler moments.

To address this, we propose expanding the range of parameters beyond basic emotions like “happy” or “sad” to include categories like ”educational” and ”sentimental.” Additionally, enhancing the role of facial expression analysis could be crucial, as nuanced conversations are often reflected in facial cues.

4.6. Workflow Integration

We further evaluated our system on how much time our tool would save. Unfortunately, our streamer friends did not give us a concrete time as to how much time they saved. However, they told us that their editors would often watch their VODs from start to finish to find the best clips. The consensus from all Streamers was that our system made the process much faster. We estimated the time spent on reviewing the clip by the time it took the streamers to fill out the ranking form on Google Spreadsheet. Since we only got ratings from streamers T and N, here are the results:

Streamer	VOD Length	Time spent with tool
T	8 hours	28 minutes
N	2.5 hours	53 minutes

These times were measured without our web UI. As a result, these estimates are likely inflated due to the streamers having to copy paste the clips manually into the browser. However, we could conclude from the numbers that such a tool would result in significant time save for the streamers and their editors.

In addition, we received enthusiastic feedback from Streamer E, a colleague of Streamer T, who indicated that our tool enabled him to generate short-form content for the first time. Prior to utilizing our tool, he had been unable to produce such content due to the absence of an editor and limited time resources. This feedback underscores the potential of our system to empower smaller streamers by significantly lowering the barriers to content creation.

Moreover, Streamers M, N, T, and E expressed interest in reusing our tool, as long as we could provide them with a website for navigating suggested clips. This interest suggests that the streamers valued the underlying concept and functionality of the tool.

5. Conclusion

Our study on different facial expression recognition models suggests that in order to train a better facial eomtion classifier, we would need a dataset with images that are bigger in scale. While FER2013 can capture bigger facial features, more nuanced expressions are often lost due to the 48×48 image size.

For future improvements, we would like to augment FER2013 with other newer and more diverse facial expression datasets. Moreover, more parameter tuning could be done to potentially improve the accuracy of the model.

On the other hand, the study around the entire system reveals the potential of our system. Through the use of models with different modalities of data, We developed a tool that makes identifying interesting moments from video streams easy and fast. The evaluations and feedback from streamers and editors suggests that the system we built, while still having plenty of room for improvements, already garnered a lot of interests. It is feasible to utilize different modalities of data, and such tool, once fully fleshed out, would fit into streamers' workflow conveniently.

While the hopes are high, we still need to tackle many technical challenges before this project can be launched to our potential users. In addition to the aforementioned improvements on the FER pipeline, we would also need to improve the run time of the software. As a large part of our code uses off-the-shelf libraries, parallelization has proven to be tricky, the run time of our software was quite long. Further more, we would like to implement a tree-like, multilevel alignment system to align and generate clips at different time granularity levels. Moreover, our current system in aggregating scores from the pipelines has weights manually tuned by us, resulting in suboptimal weights and therefore inaccurate scoring. To solve this issue, we would like to collect a labeled dataset and train the regression model using supervised learning.

We believe that we have built a system with a great potential. We plan to continue working on this project after the quarter ends, and would love to see the tool be used by streamers throughout the community.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 4

[2] Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-stage dense face localisation in the wild, 2019. 1

[3] Cheng-Yang Fu, Joon Lee, Mohit Bansal, and Alexander C. Berg. Video highlight prediction using audience chat reactions, 2017. 1

[4] Ian J. Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ramaiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Zhang Chuang, and Yoshua Bengio. Challenges in representation learning: A report on three machine learning contests, 2013. 2, 3

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. 2

[7] Ultralytics LLC. Yolov8: Real-time object detection and segmentation, 2024. Accessed: 2024-05-17. 1, 2

[8] Qing Ping and Chaomei Chen. Video highlights detection and summarization with lag-calibration based on concept-emotion mapping of crowd-sourced time-sync comments, 2017. 1

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016. 2

[10] Sefik Serengil and Alper Ozpinar. A benchmark of facial recognition pipelines and co-usability performances of modules. *Bilisim Teknolojileri Dergisi*, 17(2):95–107, 2024. 2

[11] Sefik Ilkin Serengil and Alper Ozpinar. Hyperextended lightface: A facial attribute analysis framework. In *2021 International Conference on Engineering and Emerging Technologies (ICEET)*, pages 1–4. IEEE, 2021. 3

[12] Huan Yang, Baoyuan Wang, Stephen Lin, David Wipf, Minyi Guo, and Baining Guo. Unsupervised extraction of video highlights via robust recurrent auto-encoders, 2015. 1

[13] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, Oct. 2016. 1