國立臺灣大學管理學院資訊管理研究所

碩士論文

Department of Information Management

College of Management

National Taiwan University

Master Thesis

觀眾留言應用於直播影片精彩擷取之深度學習模型

A Deep Learning Model for Extracting Live Streaming Video Highlights using Audience Messages

韓宏光

Hung Guang Han

指導教授：陳建錦 博士

Advisor: Chien-Chin Chen, Ph.D.

中華民國 108 年 6 月

June 2019

ABSTRACT

Live streaming has become a ubiquitous channel for people to learn new happenings. Although live streaming videos generally attract a large audience of watchers, their contents are long and contain relatively unexciting stretches of knowledge transmission. This observation has prompted artificial intelligence researchers to establish advanced models that automatically extract highlights from live streaming videos. Most streaming highlight extraction research has been based on visual analysis of video frames, and seldom have studies considered the messages posted by the viewer-audience. In this paper, we propose a deep learning model that examines the messages posted by streaming audiences. The video segments whose messages reveal audience excitement are extracted to compose the highlights of a streaming video. We evaluate our model in terms of multiple Twitch streaming channels. The precision of our highlight extraction model is 50.1% and is superior to several baseline methods.

Keywords: Deep Learning, Recurrent Neural Networks, Bidirectional GRU, Live Streaming, Highlight Extraction, Crowdsourcing

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Due to the advance of streaming techniques, live streaming platforms like Facebook Live, YouTube Live, and Twitch have sprung up to help people access the development of new happenings. They also have surpassed traditional media (i.e., TV and radio) in terms of the audience magnitude. For instance, in 2017, the wedding of Prince Harry and Megan royal attracted 1.29 million people at the peak of the live broadcast on YouTube Live[1], even though the wedding was broadcasted by TV stations all around the world. Also, the live streaming of the 2018 League of Legends (LOL) World Championship Final earned 200 million viewers, which achieved a magnificent audience record of the world[2]. According to the Visual Networking Index[3] released in 2018, online video traffic will grow fourfold from 2017 to 2022, and live streaming will account for 17% online video traffic by 2022. There is much evidence and several surveys that indicate that live streaming will become one of the most important channels for people to receive information.

Live streaming videos are normally lengthy and contains few exciting part [26; 33]. In order to help audiences gather the gist of the streaming events and also to attract more viewers who are the major revenue source of streaming platforms, many streaming platforms offer highlight services which provide video highlights composed by platform officials or cooperating studios. Nevertheless, creating streaming highlights is tedious because highlight editors need to examine every part of the video. In the case of the LOL

[1] https://www.tubefilter.com/2018/05/24/youtube-royal-wedding-live-stream/

[2] https://esc.watch/blog/post/worlds-2018-final

[3] https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

live streaming on Twitch, the length of a match streaming is generally longer than one hour and the released highlights are only about 5 to 8 minutes. To reduce human labor, there is an urgent need of artificial intelligence models that automatically extract highlights from video contents [5; 18; 19; 23; 24; 31; 33].

Highlight extraction of video content is an active research topic. Current extraction methods are normally based on image processing of video frames [5; 18; 19; 23; 24; 31; 33], and presume that highlights contain rich visual effects or specific actions performed by characters. Advanced deep network architectures, such as AlexNet that is based on convolutional neural networks [11], are applied to the image frames of videos to learn visual features of highlights, which are used to label highlight sections of new videos.

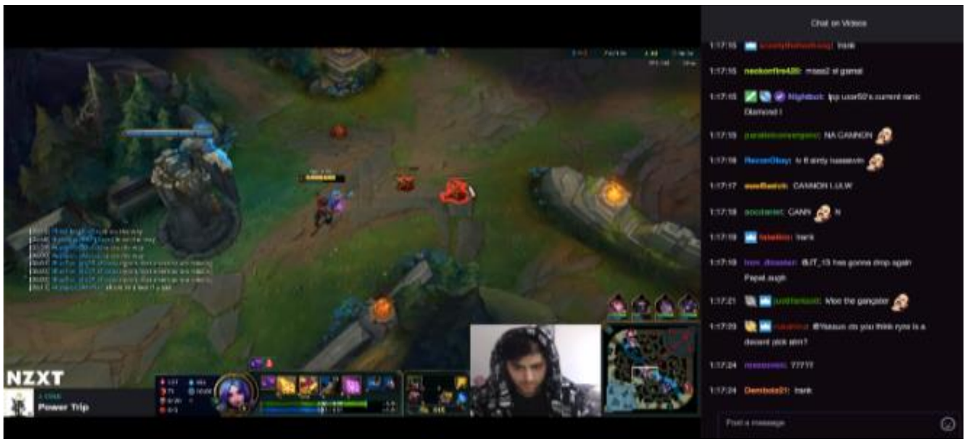

Figure 1. An example of a chat room for live streaming.

Highlights are supposed to entice an audience. It has been empirically shown that people are eager to share their feeling when they have strong emotions [4], and sentiment analysis on user messages have thus been evaluated to highlight the opinions of general public [7]. Most live streaming platforms now allow audiences to leave messages during live streaming sessions, as can be seen in Figure 1, which shows Twitch users intensely expressing their feelings during the streaming of a live game. The messages collaboratively reveal the audience's opinions regarding real-time happenings in the love streaming, and are therefore promising sources for extracting video highlights. However, few studies have applied audience messages on video highlight extraction. In this paper, rather than analyzing video frames, we examine the messages posted by live streaming audiences to extract video highlights.

A difficulty of using audience messages in the extraction of video highlights is that there exists a latency between the starting of a highlight and a corresponding audience response (e.g., messages). For example, in a video game streaming, before the winning attack, audiences would post celebration messages in advance. Here, we develop a bi-directional gated recurrent unit deep neural network (biGRU-DNN) that partitions a streaming video into a sequence of video segments and sequentially examines messages on these segments to label them as highlights. The reason that we employ the gated recurrent unit (GRU) is that GRU is a variant of a recurrent neural network [12] and is effective in processing sequential data [2]. GRU possesses a memory mechanism that helps retain the context of sequential user messages in resolving the latency difficulty.

Another difficulty of using audience messages is that audience messages often contain Internet slang, buzzwords, and emoticons [1]. To lessen the influence of the informal word usages on our model, we adopted the technique of word embedding [20] and train embedding vectors of message tokens with a huge amount of Internet messages. This way, tokens which are semantically relevant have a close embedding representation even if they involve informal word usages [13]. The evaluation results based on the dataset collected from multiple Twitch streaming channels show that the precision of our model is 50.1%. The precision is superior to several baseline methods and our user-message-oriented highlight extraction model is capable of identifying video highlights preferred by audiences.

2 RELATED WORKS

The goal of highlight extraction is to locate the key sections of a video that audiences are interested in [10]. These extracted highlights can serve as a summary of the target video [6; 8; 17; 28]. Traditional methods of highlight extraction generally focus on specific domains and use heuristic rules or domain knowledge to extract highlights [15; 16; 21; 34]. For example, [15] investigated sports videos and identified visual patterns of impressive scoring events to extract the highlights of a basketball game. However, the interests of the audience are normally so diverse that expert-defined rules can only cover a few highlight categories (e.g., scoring). Another approach of highlight extraction employs machine learning techniques to learn a scoring function that determines the importance of a video frame [8; 17; 25; 27; 32]. This scoring function examines various visual or audio features of a video frame to predict the probability that audiences would be interested in it. For instance, [15] uses the loudness of audio and the pan of visual field as features, and then customizes a heuristic model to determine which parts of a video are highlights. Some studies [6; 9; 30] examine the textual information of videos for highlight extraction. For instance, in [30] the real time webcasting text are extracted from webpages to discover video highlights. However, the textual information is only used as an auxiliary to enhance the results extracted by audio and visual features.

Recently, highlight extraction research started to adopt deep learning techniques due to their effectiveness in processing image data. In [18], the authors developed a deep action proposal model that divides the highlight extraction process into two steps. First, an action detection procedure is performed to extract candidate sections from a video, which contain important visual actions. In the second step, the candidates are evaluated by a long short-term memory (LSTM) network to predict their highlight confidence scores and to summarize (or highlight) the video by selecting representative sections.

In [33], the authors employed a pairwise learning strategy to train a deep convolution neural network. Instead of estimating the confidence scores of individual video sections, the authors paired each highlight section with a non-highlight section. The objective of the network is thus to minimize the negative difference between the confidence scores of the highlight sections and the scores of the non-highlight sections. In other words, the network aims to differentiate the highlight sections from the non-highlight sections. Again, visual features, such as the temporal dynamics (i.e., video clip) and spatial information (i.e., single frame) of the video are analyzed and input into the network to identify video highlights.

In [19], the technique of semantic segmentation was employed to discovery video highlights. Semantic segmentation is an important research subject of computer vision, and its task is to identify and classify specific objects in a picture. The authors considered a video as a picture whose pixels are video segments. The highlight extraction task is then modeled as a sequence labeling problem, and a fully convolutional network model is developed to label 1 to a video segment if it is a part of the highlight, otherwise the segment’s label is 0.

In addition to visual features, [6] also examines audience messages to enhance their highlight extraction model. The authors developed a 3-layer LSTM model that encodes a one-hot vector for each message character. Their experiment results demonstrate that audience messages are helpful in improving the performance of highlight extraction.

As shown by the above studies, research into highlight extraction mainly analyzes visual features of video frames. While Fu et al. examined audience messages, they also adopted a lot of visual features in their model and treated audience messages as auxiliary information. In this paper, we focus on the effect of audience messages on highlight extraction, so our extraction model depends only on the message sentences posted by viewers. Moreover, word embedding techniques are studied to examine the semantics of message tokens and to enhance our highlight extraction performance.

3 HIGHLIGHT EXTRACTION SYSTEM

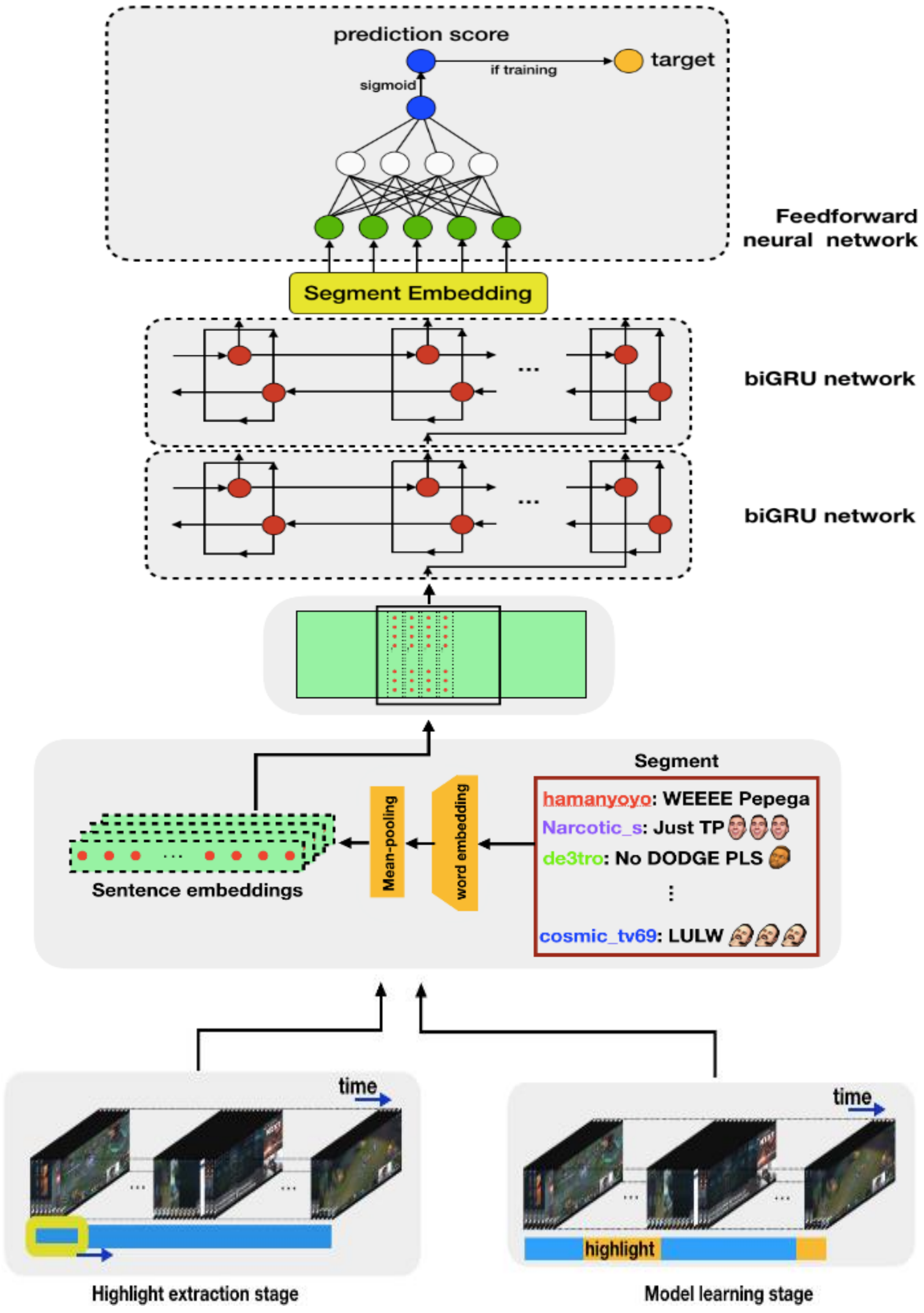

Figure 2. The system structure.

Figure 2 shows our highlight extraction system which consists of a model learning stage and a highlight extraction stage. In the model learning stage, a set of training videos together with their audience messages and labeled highlight sections are collected. We partition the videos' highlight/non-highlight sections into a series of overlapping segments, and convert their messages into a sequence of sentence embeddings. These embeddings are used to train our biGRU-DNN model by approximating the segments' highlight confidence scores. In the highlight extraction stage, the same segmentation and embedding procedures are applied to a testing video. The sentence embeddings of the video's segments are sequentially input into the learning model to predict the segments' confidence scores. The highlights of the testing video then are identified by selecting the top-confidence segments. In the following subsections, we detail the two system stages.

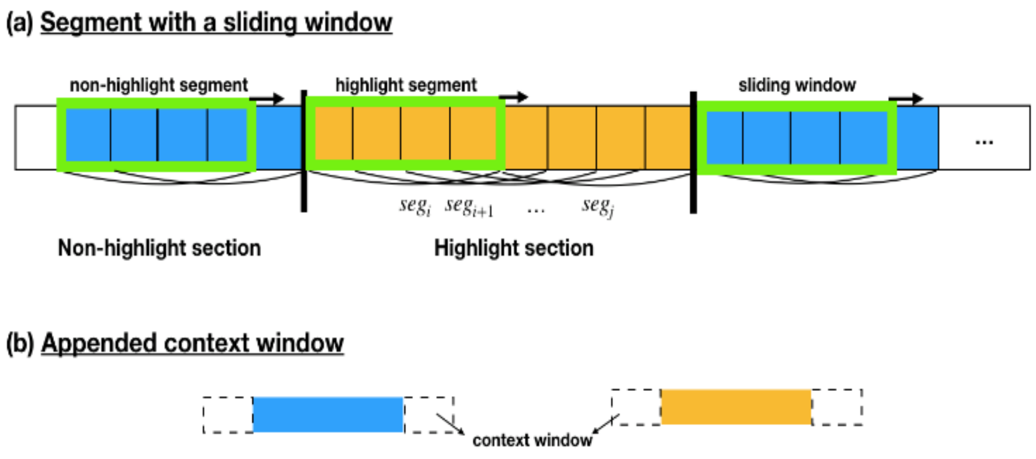

Figure 3. The segment processing procedure.

3.1 Model Learning

3.1.1 Training Data Preparation

Each training video contains one or more highlight section. As shown in Figure 3 (a), we divide every highlight section into a series of overlapping segments using a sliding window mechanism. Specifically, for every video second in the highlight section, we construct a video segment whose length is $len_{segment}$ seconds. As the example shows in Figure 3 (a), the length of a video segment is 4 seconds. These segments are assigned a highlight confidence score 1 and are considered positive training instances. Note that there generally exists a latency between the starting (or ending) of a highlight and the response of audiences. Hence, in addition to the audience messages posted in a segment, we also use a context window whose length is $len_{context}$ seconds to extend the segment's content by including the message sentences posted before and after the segment (Figure 3 (b)). Regarding the negative training instances, we presume that the messages posted immediately before and after a highlight section provide valuable information to contrast the highlight. We therefore apply the above segmentation procedure to the sections 120 seconds before and after a highlight section to generate negative segments whose confidence score is 0.

Instead of modeling message sentences as a set of independent tokens (i.e., words, symbols, or emoticons), we transform message sentences into embeddings which associate tokens in terms of their semantics. Because Internet chatting contains slang expressions and emoticons, we do not use pre-trained embeddings, such as the Google pre-trained model[4], which are derived from formal sentences. Instead, we apply Word2Vec [20], a well-known word embedding technique that is effective in understanding the semantics of natural languages, to a huge number of streaming messages and represent a token as a high-dimensional embedding vector. The embedding vectors of a sentence's tokens are then averaged to construct the sentence's embedding.

[4] https://code.google.com/archive/p/word2vec/

Finally, the embeddings of all the sentences in a segment are input into the biGRU-DNN model to determine the highlight confidence score of the segment.

3.1.2 The biGRU-DNN Model

Our biGRU-DNN model consists of two types of networks: a two-layer bi-directional GRU network and a multiple-layer deep neural network (DNN) network. The reason we adopted GRU in processing audience messages is that GRU is a variant of recurrent neural networks and is effective in retaining contextual information when recognizing sequential data [12]. Audience messages are a sequence of sentences that can contextually reveal the intention of a crowd of people. It is therefore appropriate to analyze user messages using GRU. Although LSTM is also good at retaining context information of sequential data, GRU is superior to LSTM in terms of the training speed [2].

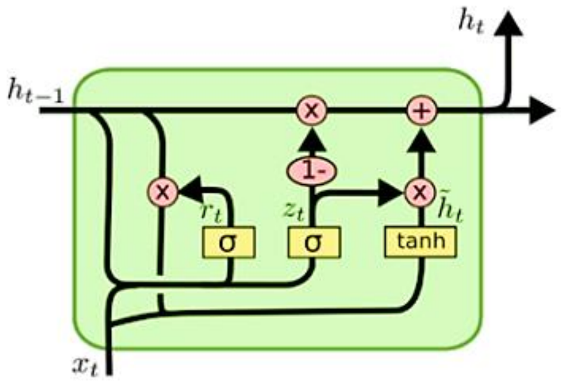

Figure 4. A GRU Unit.

For each segment, the first layer of the bi-directional GRU network recurrently examines the sentence embeddings occurring in a time unit. In this study, the unit length is 1 second and zero padding is adopted to ensure a fixed input size. The second layer of the GRU network then sequentially processes the embedding produced by the first layer to output the segment's embedding. Finally, the segment embedding passes through the multiple-layer deep neural network with a sigmoid function to predict a value between [0,1], which represents the highlight confidence score of the input segment. Conceptually, a bi-directional GRU layer is composed of two GRU units that respectively examine the forward and backward context information [22]. The structure of a GRU unit (Figure 4) computes the hidden state h_t of current input embedding x_t as follows:

$$z_t = \sigma(x_t U^z + h_{t-1} W^z + b^z) \tag{1}$$

$$r_t = \sigma(x_t U^r + h_{t-1} W^r + b^r) \tag{2}$$

$$\widetilde{h_t} = tanh\ (x_t U^h + (r_t * h_{t-1})\ W^h + b^h) \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t}) \tag{4}$$

where symbols z_t and r_t denote the update gate and the reset gate, respectively, and b indicates a bias. The candidate hidden state $\widetilde{h_t}$ is managed by the reset gate to proportionally combine the current input x_t with the previous memory h_{t-1}. The update gate described in Equation 4 determines how much previous memory should be kept when producing the current hidden state. Note that if the values in the reset gate are 1 and 0 for the update gate, a GRU unit turns into the vanilla RNN.

To train our model, the network parameters (weights) are randomly initialized. Then, the gradient descent is deployed to iteratively refine the parameters and to minimize the following loss function $H_p(.)$:

$$H_p(Q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(p\left(seg_{embedding_i}\right)\right) + (1 - y_i) \cdot \log\left(1 - p\left(seg_{embedding_i}\right)\right) \tag{5}$$

where Q = {<$seg\_embedding_1$, y_1>, <$seg\_embedding_2$, y_2>, …, <$seg\_embedding_N$, y_N>} is the set of training segments and N is the number of the segments. The symbol $seg\_embedding_i$ represents the sentence embeddings of the i'th training segment and y_i is the segment's highlight confidence score. The function $p(seg\_embedding_i)$ produces the predicted confidence score of a segment under the network parameters. Basically, the loss function is the binary cross entropy [3; 14], in which the lower the cross entropy, the better the network parameters.

3.2 Highlight Extraction

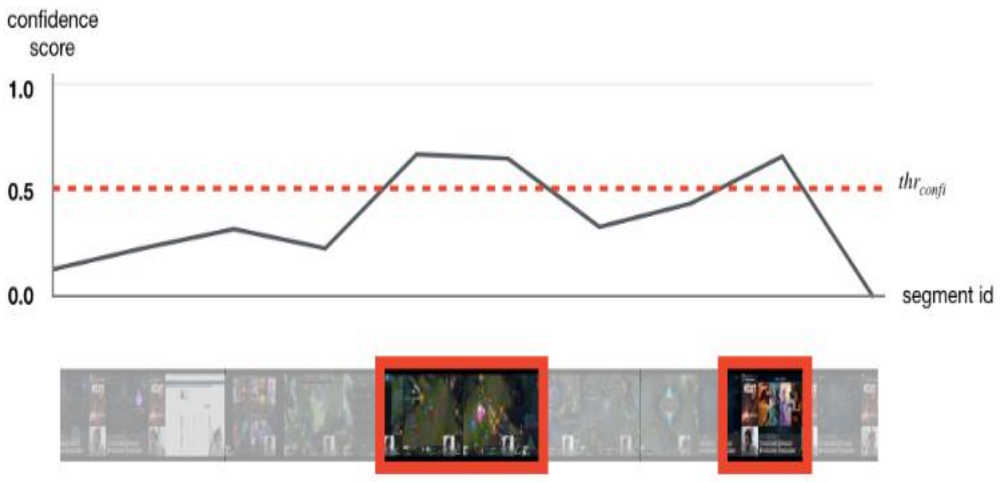

Figure 5. An Example of Highlight Extraction.

In the highlight extraction stage, the same video processing procedure is employed to partition a testing video into a series of overlapping segments. All the segments' sentence embeddings are sequentially fed into the learned network to predict their confidence scores. We identify the highlight of the video by specifying a confidence threshold (denoted as thr_{conf}). As illustrated in Figure 5, consecutive segments whose confidence scores are above the threshold are concatenated to create the video's highlight sections.

4 EXPERIMENT

4.1 Datasets and Evaluation Metrics

Due to the popularity of the online game League of Legends (LOL), Twitch contains a multitude of LOL live streaming videos. These videos also attract a lot of viewers and volunteer studios who contribute high-quality video highlights. We therefore downloaded the LOL live streaming videos and their highlights for experiments. However, some of the videos were not suitable for experiments because they were too short. We thus filtered out the videos whose lengths are shorter than 40 minutes and collected 491 videos for evaluation. Table 1 shows the statistics of the experiment data.

Table 1. Statistics of the experiment videos.

Number of experiment videos	491
Number of training videos	353
Number of validation videos	88
Number of testing videos	50
The length of the videos (sec.)	11,351,902
The length of highlight sections (sec.)	77,313
The number of user messages	37,118,352
The number of message tokens	197,169,890

When processing the experiment videos, we noticed that many of them were viewed by more than one studio and different studios produced different highlight sections. To test the generality of our method, the ground truth of a video was constructed by merging all the highlight sections edited by different volunteer studios. The conventional holdout evaluation procedure [29] was adopted to obtain fair evaluation results. Under the procedure, 70% of the collected videos (i.e., 353 videos) were selected to train our GRU-based highlight extraction model and 88 videos were used as the validation set to adjust our system parameters. The remaining videos were used for testing. For each of the 50 testing videos, our method estimated the highlight confidence scores of its segments. We set the extraction threshold thr_{conf} at the top 2% confidence score of the segments and selected all high-confidence segments to produce the predicted highlight sections. Then, the following precision (P) and recall (R) were applied to the predicted highlight sections and the ground truth highlight sections to estimate the highlight extraction performance.

$$P = \frac{1}{|D_{test}|} \sum_{i \in D_{test}} \frac{|highlight_{truth}^{i} \cap highlight_{predicted}^{i}|}{|highlight_{predicted}^{i}|} \quad (6)$$

$$R = \frac{1}{|D_{test}|} \sum_{i \in D_{test}} \frac{|highlight_{truth}^{i} \cap highlight_{predicted}^{i}|}{|highlight_{truth}^{i}|} \quad (7)$$

where $|D_{test}|$ is the number of testing videos, and the symbols $|highlight_{predicted}^{i}|$ and $|highlight_{truth}^{i}|$ respectively denote the length (sec.) of the predicted highlight section and the length of the ground truth highlight section of the ith testing video. Note that our GRU-based model was implemented by using the Keras neural network package[5] which adopts an enhanced stochastic gradient descent to acquire the model parameters. The hyperparameters of the gradient descent are set as follows: batch size as 500 (500 samples in one batch), learning rate as 10^{-4}, epoch size as 20, and decay rate as 10^{-6}. Also, the embedding dimension for each message token is 50. Our model contains 3 layers of bi-directional GRU whose output dimension is 256, and 3 layers of a fully connected

[5] https://keras.io/

network whose output dimensions are 128, 32, 1, respectively. To prevent overfitting, we inserted four layers of dropout, with a drop rate of 0.5.

4.2 Effects of System Parameters on Model Training

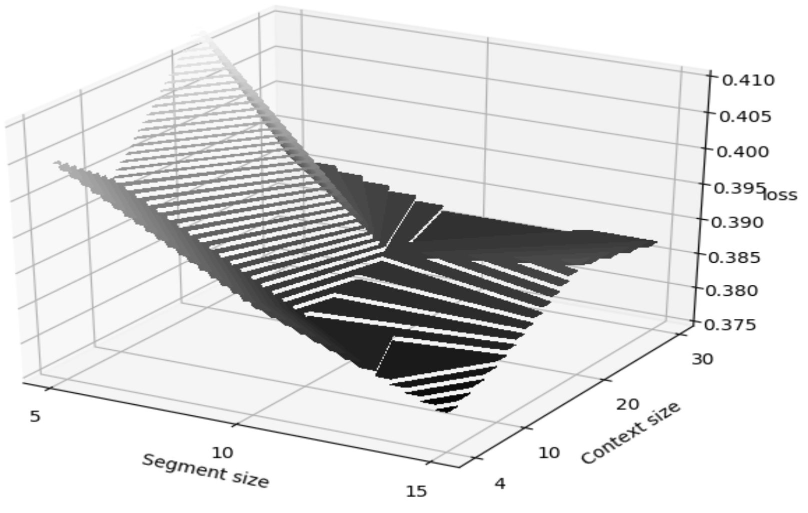

Figure 6. The effect of parameter settings.

Here, we examine the parameters $len_{segment}$ and $len_{context}$, which respectively denote the length (sec.) of a segment and the length of the segment context. We tested different parameter settings and measured the corresponding binary cross entropy loss (i.e., Equation 5) on the validation videos. As shown in Figure 6, the smaller the $len_{segment}$, the higher the entropy loss. This is because the messages in a short segment are too few for our model to correctly detect highlights. Intuitively, a large $len_{context}$ is helpful for our model to acquire context user messages. However, entropy loss increases as context length increases; because a large $len_{context}$ (e.g., 30 seconds) contains too much context information, it distracts our model from detecting highlights. In the following experiments we set len_{window} to 15 seconds and $len_{context}$ to 4 seconds due to their superior entropy loss.

4.3 Comparisons with other Methods

In this section, we compare our model with the following baseline methods. To obtain fair evaluation results, the same holdout evaluation procedure was applied to these baseline methods.

Message density model. It has been observed that people are eager to share their feeling when they have strong emotions [4]. The highlights of an online video would therefore receive a lot of discussion. The message density model sorts the segments of a testing video according to their message numbers. The top 2% message-filled segments were selected to form the predicted highlight sections for comparisons.

L-Char-LSTM Model. Fu et al. developed a 3-layer LSTM model that examines audience messages for video highlight extraction [6]. The authors encoded a one-hot vector for each message character, and concatenated all the vectors of a segment to construct the input vector which was fed into the LSTM model to predict the highlight score. Again, the top 2% segments were selected for comparisons. We implemented the method by using the program released by the authors on GitHub. These authors also suggested two model settings: the $L_Char_LSTM_{100\%}$model that uses all the annotated highlight sections of training videos for model training, and the $L\_Char\_LSTM_{25\%}$, which is only trained with the last 25% of highlights. Their evaluation results show that the last 25% model is more effective than the 100% model in extracting video highlights.

Table 2. The comparison results.

Model	*P*	*R*
L-Char-LSTM$_{100\%}$	0.376	0.034
L-Char-LSTM$_{25\%}$	0.245	0.022
The message density model	0.372	0.033
biGRU-DNN	0.501	0.066

Table 2 shows the performance of the compared methods. While the precision of our method is only 50.1%, our method still outperforms the compared methods. The recall scores of all the methods are low. This is because the highlights of the testing videos are generally edited by more than one studio, and since the studios have different tastes for highlights, the ground truth has substantial variation. The highlights predicted by the methods therefore cover a small subset of the ground truth that leads to an inferior recall performance. Nevertheless, our recall performance still emerges superior to those of the baseline methods.

The message density model simply examines the amount of messages posted by the viewers. Its performance, however, is comparable to that of the L-Char-LSTM model. This finding validates the argument in [4] that people are generally willing to share their feelings when they have strong emotions (e.g., excitement). The number of messages posted by the audience is thus helpful in identifying highlights. However, for very popular live streaming videos, there are too many viewers, and the crowds keep posting messages so that the distribution of message densities is too uniform for the model to detect highlights. L-Char-LSTM and our method, in contrast, further examine the message contents and therefore achieve better extraction performances.

5 CONCLUSION

Live streaming offers an accessible manner for people to learn new events. However, the length of a live streaming video is generally too long for audiences to capture the gist of the happenings. In this paper, instead of analyzing video frames and their visual features, we examine the responses of live streaming audience members to extract live streaming video highlights. We developed a deep neural network model that evaluates the messages posted by audiences to detect live streaming highlights. The evaluation results based on real-world live streaming data show that the proposed biGRU-DNN model is promising and it outperforms several baseline methods. In future work, we will incorporate other audience behavior into our model. Features like the message density in a segment will be further investigated to enhance the extracted results by using message tokens. Also, different baseline methods and evaluation metrics will be evaluated to demonstrate the effectiveness of our model.

6 ACKNOWLEDGMENTS

This research was supported by the Ministry of Science and Technology of Taiwan under grant MOST 106-2628-H-002-002-MY3.

7 REFERENCES

[1] BARBIERI, F., ANKE, L.E., BALLESTEROS, M., SOLER, J., and SAGGION, H., 2017. Towards the understanding of gaming audiences by modeling twitch emotes. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 11-20.