Video Review Analysis via Transformer-based Sentiment Change Detection

Zilong Wu[1], Siyuan Huang[2], Rui Zhang[1], Lin Li[1]
[1] *Wuhan University of Technology, Wuhan China, 430070, {zilongwu97@163.com, zhangrui@whut.edu.cn, cathylilin@whut.edu.cn}*
[2] *The George Washington University, Washington DC, USA, 20052, siyuanhuang@email.gwu.edu*

Abstract

With the popularization of video websites, huge amounts of reviews have been generated by the audience online. These reviews usually reflect people's sentiment changes which are valuable information for understanding the market and guiding social media. A number of studies analyzing sentiment changes have made great progress. However, most of them are not sensitive enough to detect sentiment changes thoroughly and subtly. To solve this problem, this paper proposes a transformer-based sentiment change detection model by using the Anomaly Clearing Cumulative SUM (ACCSUM) model to analyze review streams and distinguish sentiment change points therein. Our proposed model can detect sentiment change points precisely by excluding abnormal reviews from review streams. Experiment results show that our model can find sentiment change points with greater sentiment drop.

Keywords: sentiment change; transformer; stream analysis

1. Introduction

With the rise of the internet, video websites like YouTube and Netflix become valuable public opinion sources. Millions of users watch videos and express their views on video websites every day. Interestingly, reviews about videos may change over time and user attitudes may as well change due to unexpected reasons. Thus, analyzing sentiment changes is an important part of sentiment analysis.

A number of studies researching them have made great progress. Zhang et al. analyzed the sentiment evolution among reviews [10]. In their paper, a grained sentiment model is built to detect different sentiment evolutions on Microblog. Anastasia uses time series to track the general election through Twitter [3]. However, such models have deficiencies in detecting sentiment changes. For example, Zhang's and Anastasia's models both use lexicon dictionary to identify opinioned terms, which is inefficient in parsing sentences with complex structure or vocabulary in different semantic environments. These drawbacks of sentiment dictionary would affect the accuracy of traditional models.

In our proposed model, we use the transformer model [9] to avoid the deficiencies of lexicon-based model and transform sentences into sentiment vectors. Inspired by Devlin's pre-trained model [2], we utilize it to obtain reviewers' sentiments. The advantage is that its attention mechanism makes it possible for the words in a sentence to build a connection with others [1]. Even the last word in a sentence can get associated with the first so that the long-term dependency problem can be solved [1].

Besides, inspired by Sotiris's paper [8], we improve the Cumulative SUM (CUSUM) [5] algorithm to detect sentiment changes. CUSUM is a typical method of stream change detection and it can detect even a small shift around the whole sentiment stream. Although CUSUM performs well for typical streams, it is inefficient in dealing with outliers. We improve it by importing Streaming Peaks-over-Threshold with Drift (DSPOT) [7]. It removes binding reviews that deviate far from main sentiments. Since this paper aims to analyze the main sentiment of reviews, we eliminate the reviews containing irregular sentiments by DSPOT. Finally, reviews related to those sentiment changes are extracted by term frequency-inverse document frequency (TFIDF) [4], which helps us observe video-related events.

Our contributions are as follows：

- We successfully solved the semantic problems and long-term dependency problems by using transformer model.
- We improved CUSUM model to detect sentiment change points by removing outliers from streams with DSPOT algorithm.
- Experiments on real sports video reviews show that sentiment change points are relevant to unexpected news, which depicts a potential to predict them in the future.

The rest of the paper is organized as follows. Section 2 briefly reviews some related work and provides background knowledge. In Section 3, we describe our model in detail. Section 4 shows experimental results of our model. The whole work is concluded in Section 5.

978-1-7281-4272-2/20/$31.00 ©2020 IEEE
DOI 10.1109/MIPR49039.2020.00074

2. Related Work

2.1. Sentiment Change

Sotiris's research detects the changes of review stream by focusing on detecting real-time sentiment changes of tweets [8]. In his paper, the lexicon-based classification method is used to classify reviews' sentiments, and CUSUM algorithm is used to identify sentiment change points in the review stream. The experiment shows great result in finding the sentiment change points from thousands of reviews.

2.2. Attention Mechanism

Transformer model relies entirely on self-attention mechanism [1]. In the Transformer model, self-attention mechanism works by using

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V \tag{1}$$

where Q, K, V are three vectors created by the transformer model. Q denotes queries that represent all input words. K denotes keys that represent the currently selected word. V denotes values of currently selected word. d_K denotes the dimension of each vector in K.

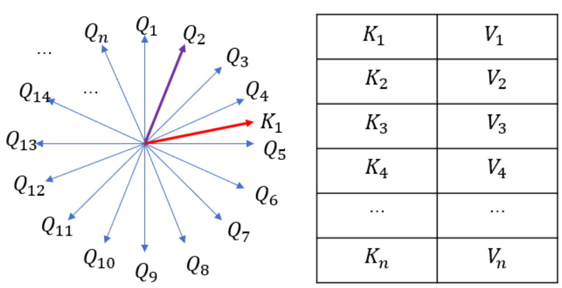

Figure 1. The illustration of self-attention mechanism.

Fig. 1 shows how attention mechanism works, where K_1 is the current input vector and Q_2 is a random word vector in the sentence. By calculating the dot product, the transformer learns how relevant two words are so that the attention of word vector can be easily calculated by multiplying the value of the word V_1. Here, a large value of d_K would largely increase the multiplication in magnitude and force the softmax into a narrow range. Therefore, all attention outputs are divided by $\sqrt{d_K}$.

2.3. CUSUM

CUSUM is a sequential analysis technique developed by E. S. Page [5], used for discrete random signal with i.i.d. samples. To describe CUSUM, we consider the input sequence as y, and y_i denotes the i^{th} review. The probability density $p_\theta(y)$ depends upon parameter θ:

$$p_\theta(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\theta)^2}{2\sigma^2}} \tag{2}$$

Here we suppose that the case follows $N(\theta, \sigma)$, and θ now refers to the mean value of y. If no change happens across all samples, θ will remain constant as the initial value θ_0. But in case of a change, the average of current samples will move around, which makes θ equal a different value θ_1. Thus, the problem is to detect changes of θ. Each time when a new sample is added to CUSUM, we will recalculate θ. Once θ reaches a specified ending average, it will be identified as a change point.

2.4. DSPOT

DSPOT is an approach modified from Peaks-Over-Thread (POT) [7]. POT determines thresholds in a static stream.

$$z_q \approx t + \frac{\hat{\sigma}}{\hat{\gamma}}\left(\left(\frac{qn}{N_t}\right)^{-\hat{\gamma}} - 1\right) \tag{3}$$

where z_q is the destination threshold, t is a designed threshold, q is a designed probability, n is the total number of observations and N_t is the number of peaks over t. $\hat{\sigma}$ and $\hat{\gamma}$ are parameters evaluated through observations [7]. DSPOT, as a modification of static POT, assumes that the thresholds of stream data change over time. Suppose we have n observations $X_1, \dots, X_n$, DSPOT uses the variable change

$$X'_j = X_j - M_j \tag{4}$$

where X'_j denotes the change of X_j,

$$M_j = \left(\frac{1}{b}\right) * \sum_{a=1}^{b} X^*_{j-a} \tag{5}$$

is moving average of j^{th} observation. $X^*_{j-1}, \dots, X^*_{j-b}$ denote the last b observations. Then by using the POT algorithm, we can calculate threshold z_q.

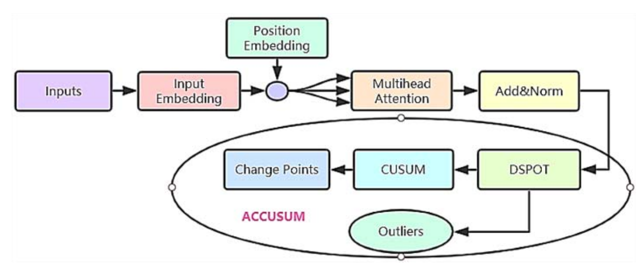

Figure 2. Architecture of our method.

3. Architecture

In this section, we describe our method in detail. Fig. 2 shows the whole architecture of our method and it can be divided into two parts: Transformer-based training and ACCUSUM algorithm. Transformer model translates input review streams into sentiment vectors, while ACCUSUM detects sentiment change points among the vectors processed by Transformer model.

3.1. Simplified Transformer model

In this section, we make the sentiment of reviews tractable by using transformer model. First, the reviews are transformed into word vectors using Word2Vec model [6]. Then, the transformer model is used to analyze review sentiments and transform those embedding vectors into 2-dimensional items. After labeling, we use Word2Vec to represent reviews as input embedding sequence whose max-length is set to 100.

Different from regular transformer architecture for language translation, we use only the encoder part of transformer model to transform reviews into sentiment vectors rather than the whole sentences. The encoder part has two sub-layers. The first layer includes a simple position encoding (PE), which records the position of each word in a sentence. It uses sin and cos functions for different frequencies:

$$\mathrm{PE}_{(\mathrm{pos},2s)} = \sin(\mathrm{pos}/10{,}000^{\frac{2s}{d_{\mathrm{model}}}}) \quad (6)$$

$$\mathrm{PE}_{(\mathrm{pos},2s+1)} = \cos(\mathrm{pos}/10{,}000^{\frac{2s}{d_{\mathrm{model}}}}) \quad (7)$$

where pos is the position, s is the dimension and d_{model} is the dimensionality of input. Since the wavelengths vary from 2π to $10{,}000{*}2\pi$, each position will be encoded differently to represent a unique position.

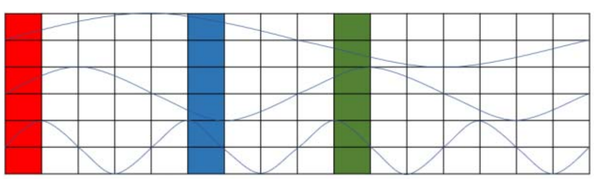

Figure 3. Position encoding illustration.

Fig. 3 explains how position encoding works. The red, blue, green columns represent different words in a sentence and their red, blue, and green position codes are (1,1,1), (1,0,1) and (0,1,1) respectively. In this case, transformer converts the position information to vectors.

In the second layer, we do self-attention mechanism multiple times.

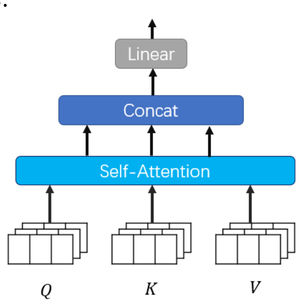

Figure 4. Multi-Head Attention consists multiple self-attention. Results of those attention will be concatenated together.

As is shown in Fig. 4, each head has different features. We concatenate all heads' results and condense them into a 2-dimensional vector after calculating each head's attention. Head size, mapping vector dimension, batch size and learning rate are set to 8, 16, 64 and 0.005, respectively.

3.2. ACCUSUM Algorithm

To detect sentiment changes, we adopt the ACCUSUM algorithm. ACCUSUM consists anomaly clearance part and change points detection part. DSPOT model is applied to clear the anomaly data in reviews and CUSUM model is applied to detect change points.

3.2.1. Anomaly Clearance

DSPOT is the model for anomaly clearance. X_j means the sentiment of j^{th} review. Since reviews have both positive and negative anomalies, in the DSPOT model we have two z_q to represent the upper bound and bottom bound. Once X_j is greater than the upper bound or smaller than the bottom bound, it will be recorded as outlier. The mechanism of DSPOT is shown in Fig. 5 where FLAG denotes a change point.

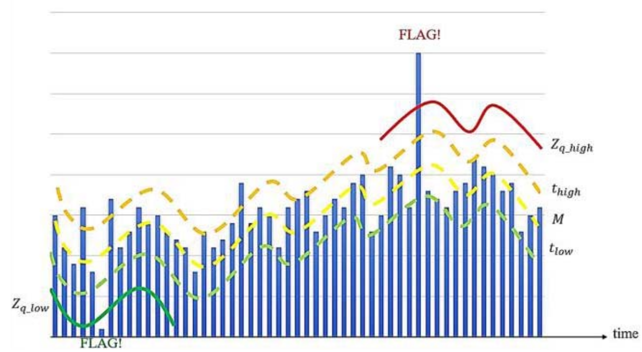

Figure 5. DSPOT mechanism: once the data exceeds z_{q_high} or below z_{q_low}, it will be recorded as outlier.

3.2.2. Change Points Detection

CUSUM is used to detect changes. We determine sufficient statistic s_v the instantaneous log-likelihood ratio at time n and use the magnitude of the changes δ to represent s_v. The advantage is that it avoids inaccuracy of picking specific value. We define s_v as

$$s_v = \frac{\delta}{\sigma^2}(y - \theta_0 - \frac{|\delta|}{2}) \quad (8)$$

To decide the time to reach the change points, we have formula as follow:

$$S_k = \sum_{u=1}^{k} s_u \quad (9)$$

$$g_k = S_k - m_k \geq h \quad (10)$$

$$m_k = \min_{1 \le j \le k} S_j \quad (11)$$

where k represents the current point we check, S_k represents the aggregate from s_1 to s_k, h refers to the threshold, and g_k refers to the decision for different time. Once it reaches terminating condition $g_k \ge h$, it picks point k as a change point. This is the process of one-side CUSUM. It is necessary to detect changes in both directions of sentiment as there are not only positive changes but also negative changes. A simple solution is to use two one-sided algorithms,

$$s^i = +\frac{|\delta|}{\sigma^2}(y - \theta_0 - \frac{|\delta|}{2}) \quad (12)$$

$$s^d = -\frac{|\delta|}{\sigma^2}(y - \theta_0 + \frac{|\delta|}{2}) \quad (13)$$

where s^i detects increment and s^d detects decrement. When a change is detected, the current location is reset to 0, and then θ becomes the current average.

4. Experiments

4.1. Transform reviews

In this section, 61,514 reviews from Apple's App Store are used to train the transformer model with hidden size = 128, B = 64, and α = 0.005. Sentences with 1-2 stars are labeled as negative while 3-5 star reviews are labeled as positive. Cross entropy is used as polarity classification loss function. The accuracy of the transformer model is 89.33%, which proves its effectiveness. After the training, transformer indicates that there are 497 positive reviews and 214 negative reviews. We can see the whole review stream is dominated by positive attitude, because the video is related to the news that an NBA player won MVP in 2014.

4.2. Clear outliers

Reviews vectors are imported into DSPOT model and find binding reviews from the stream.

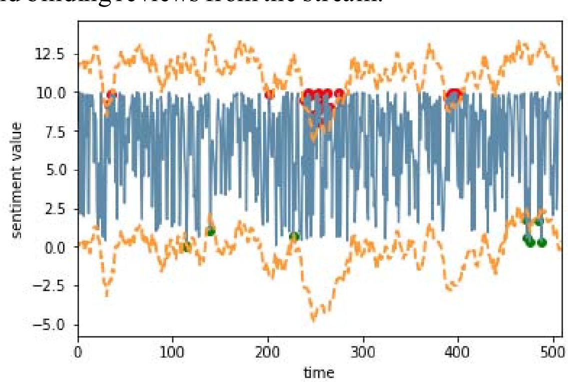

Figure 6. Anomaly detection: DSPOT detects outliers along the time.

Fig. 6 shows experimental result of DSPOT, where 711 reviews decrease to 511. The first 200 reviews are extracted as samples to detect outliers. The upper yellow curve represents the upper threshold and the lower curve represents the lower threshold. Red points are outliers that exceed the upper threshold while green are outliers lower than the lower threshold. The outlier of the 315th review is used as an example as shown in Table 1.

Table 1. Reviews around the 315th reviews.

id	review
312	fat and covered in gold chains glad she stayed humble
313	1:04 Youdarealmvp
314	only half a million views for something so real. you can tell that memes are the mainstream pop excrement of social media.
316	I never knew this was such a serious, emotional and meaningful speech. I feel bad laughing about the memes that were from this.
317	No KD, you the real MVP.
318	For as much shit as we give him for this, it really was an amazing speech
315	**Lebron does not even care when he gets a mvp**

Emotional differences are shown between the 315th reviews and others. Reviews around 315th mostly are positive towards Kevin, while the 315th review shows negative aggressive feeling toward Kevin for comparing him with another player. There are in total 38 outliers being detected. They are removed before CUSUM runs, and h = 3 and δ = 1 in this situation.

4.3. CUSUM

We use CUSUM to detect sentiment change points from clean data preprocessed by DSPOT.

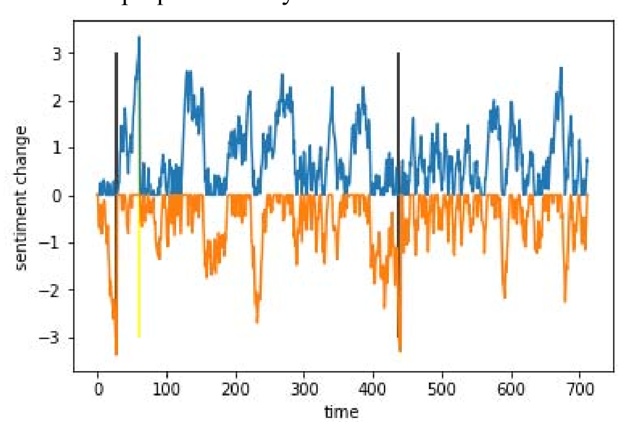

Figure 7. Detection of sentiment change points.

Fig. 7 shows the result of CUSUM, where black lines represent sentiment drops and yellow lines represent rises.

Positive value represents positive attitude and vice versa. There are in total two drops and two rises.

Figure 8. Word cloud of first drop.

Figure 9. word cloud of second drop.

Table 2. Keywords extract from changed stream.

	Key words
Change Point 1	me, cry, tre, love, dislikes, disliked, everybody, son, todaylove, kidmade, out, KD, meme, laugh, Lebron
Change Point 2	his, KD, mom, but, respect, who, snake, ass, down, what, think, got, warrior, your, mad

According to the time, the first drop appeared 5 years ago and was exactly the time when Kevin won the MVP. In Fig. 8, the keyword "Lebron" can explain the sentiment drop. Fans who support Lebron to be MVP may show their negative feeling toward Kevin through their reviews. As for the second sentiment drop, in Fig. 9 and Table 2, we detect the valuable information like "dislike", "leborn", "snake" and "worrier". People call Kevin snake for his betrayal due to his leaving of home team and joining in enemy Golden State 3 years ago.

4.4. Performance of ACCUSUM

In this section, we would like to explain the performance of ACCUSUM. Here we set a rule to evaluate the model's performance of finding sentiment change points. We take 30 reviews before each change point and detect the sentiment changes by calculating the differences between prior 15 reviews and latter 15 reviews' average sentiment:

$$C = \left|\frac{\sum_{1}^{15} R_i - \sum_{16}^{30} R_i}{15}\right| \tag{14}$$

Here, C denotes sentiment change of the change point and R_i denotes the i^{th} review's sentiment. For C the greater the better.

Table 3. Performance of different models.

	1^{st} $sentiment$ change	2^{nd} sentiment change
T+A	2.69	1.6
T+C	2.69	1.39
L+A	1.79	0.73
L+C[7]	1.79	\

In Table 3, we evaluate the performance of 4 different models. Here, T represents transformer model, A represents ACCUSUM model, C represents CUSUM model and L represents lexicon-based model. In the first sentiment change, we can see there is no difference between ACCUSUM and CUSUM. The reason is that the first change happens among the first 200 reviews. ACCUSUM will extract the first 200 reviews as samples to detect outliers. In Table 3, transformer-based models achieve better performance in finding sentiment drops than lexicon-based models, and the T+A model has the best performance out of all four methods.

5. Conclusion

In this work, we propose ACCUSUM to detect sentiment changes in review streams. ACCUSUM has improved the lexicon-CUSUM model and identify the sentiment change points more accurately. Moreover, we successfully detect sentiment drops in the example stream and find the reason for the drops, which are jealousy from fans and betrayal from the player. The detection results make it possible to explain the unexpected events behind reviews' sentiment changes, and this topic shows great research prospects in review analysis. However, using polarity of reviews to detect sentiment changes still has limitations as it is hard to truly understand people's mind towards a video or a player. Opinion mining may be a good solution to this, because it is able to classify reviews more clearly. In the future, we will combine ACCUSUM with opinion mining to seek performance improvement.

References

[1] Bahdanau, Dzmitry, Kyunghyun Cho and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate." *ICLR*, (2015).

[2] Devlin, Jacob, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. "Bert: Pre-Training of Deep Bidirectional

Transformers for Language Understanding." *NAACL-HLT,* (2019).

[3] Giachanou, Anastasia and Fabio Crestani. "Tracking Sentiment by Time Series Analysis." In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 1037-1040, 2016.

[4] Martineau, Justin Christopher and Tim Finin. "Delta Tfidf: An Improved Feature Space for Sentiment Analysis." In *Third international AAAI conference on weblogs and social media*, 2009.

[5] Page, Ewan S. "Continuous Inspection Schemes." *Biometrika* 41, no. 1/2 (1954): 100-115.

[6] Rawte, Vipula, Aparna Gupta and Mohammed J Zaki. "Analysis of Year-over-Year Changes in Risk Factors Disclosure in 10-K Filings." In *Proceedings of the Fourth International Workshop on Data Science for Macro-Modeling with Financial and Economic Datasets*, 1-4, 2018.

[7] Siffer, Alban, Pierre-Alain Fouque, Alexandre Termier and Christine Largouet. "Anomaly Detection in Streams with Extreme Value Theory." In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1067-1075, 2017.

[8] Tasoulis, Sotiris K, Aristidis G Vrahatis, Spiros V Georgakopoulos and Vassilis P Plagianakos. "Real Time Sentiment Change Detection of Twitter Data Streams." *INISTA*, (2018): 1-6.

[9] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. "Attention Is All You Need." In *Advances in Neural Information Processing Systems 30*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, 30, 2017.

[10] Zhang, Lumin, Yan Jia, Xiang Zhu, Bin Zhou and Yi Han. "User-Level Sentiment Evolution Analysis in Microblog." *China Communications* 11, no. 12 (2014): 152-163.