Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 112 (2017) 2230–2239

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2017, 6-8 September 2017, Marseille, France

Enhanced Prediction of User-Preferred YouTube Videos Based on Cleaned Viewing Pattern History

Peter Braun[a], Alfredo Cuzzocrea[b], Lam M.V. Doan[a], Suyoung Kim[a], Carson K. Leung[a,*], Jose Francisco A. Matundan[a], Rashpal Robby Singh[a]

[a]*University of Manitoba, Winnipeg, MB, R3T 2N2, Canada*
[b]*University of Trieste and ICAR-CNR, 34127 Trieste (TS), Italy*

Abstract

In current era of big data, a wide variety of high-volume data having different veracity can be easily collected or generated at a high velocity. Social network data, as well as audio and video in social media and social networking sites, are examples of big data. Embedded in these big data are valuable information and knowledge. To discovery this implicit, previously unknown and potentially useful information and knowledge from these big data, some big data science solutions are in demand. In this paper, we explore big data mining techniques for detecting outliers or anomalies from YouTube video viewing history and data-cleaning this viewing log so that the user-preferred YouTube viewing patterns or trends can be recognized and the prediction of user-preferred YouTube videos can then be enhanced.

Peer-review under responsibility of KES International

Keywords: Data mining, outlier detection, anomaly detection, Youtube, video viewing history, trends, prediction, applications, knowledge engineering, large-scale systems

1. Introduction

In current era of big data, a wide variety of high-volume data having different veracity can be easily collected or generated at a high velocity (e.g., [1-3]). Social network data, as well as audio and video in social media and social

* Corresponding author.
E-mail address: kleung@cs.umanitoba.ca (C.K. Leung)

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of KES International
10.1016/j.procs.2017.08.129

networking sites, are examples of big data. Embedded in these big data are valuable information and knowledge. To discovery this implicit, previously unknown and potentially useful information and knowledge from these big data, some big data science solutions are in demand. In this paper, we explore big data mining techniques for detecting outliers or anomalies from a large-scale system—namely, YouTube video viewing history—for data cleaning viewing so that the prediction of user-preferred videos can be enhanced.

Specifically, data mining [4-13] discovers implicit, previously unknown and potentially useful information or knowledge from data. There are many different algorithms and methods for mining data in order to discover desired information and knowledge. Our *key contribution of this paper* is our algorithm that detects videos in users' YouTube history that do not represent their typical behavior, which we will refer to as *outliers*. Motivations to carry out this study are listed as follows:

1. Many algorithms rely on accurate users' history data;
2. The outliers in existing algorithms were regarded as noise instead of being addressed directly as we show in this project;
3. The outliers find their ways into users' history through many common means, thus leading to an incorrect model of users' behavior.

We asked participants tell us how much the different attributes of all YouTube video meant to them. We then compared every video in the participants' history to their preferences and gave each video a compatibility and certainty score. We presented many outlier videos to the participants and asked them whether or not the video is one they intended to watch. Our outlier detection found videos that deviate from users' normal behavior with a good success rate.

The remainder of this paper is organized as follows. Section 2 gives background and motivation of this work. Section 3 discusses related work. In Section 4, we present our outlier detection algorithm for cleaning YouTube user viewing history data. Evaluation and conclusions are given in Sections 5 and 6, respectively.

2. Background, motivation, and challenges

YouTube is currently a popular video-sharing website in the world, boasting over a billion users and hundreds of billions of views. YouTube allows billions of people around the world to discover, watch, and share originally-created videos. It also provides tools to connect, inform, and inspire people across the globe as it acts as a distribution platform for original content creators and advertisers.

With millions of videos available, YouTube users are faced with the problem of optimizing their searching experience. Few may have the time to spend hours looking for a video they might enjoy. To combat this, YouTube usually recommends videos to its users based on their viewing habits.

Users of YouTube can create an account and make personalized changes to their viewing experience on YouTube. They can subscribe to content creators they enjoy and rate videos that they watch. In addition, users have the option (enabled by default) to record the history of the videos they watch.

In order to continue making improvements to their platform, YouTube receives a large portion of their income from advertisements. These advertisements play throughout their site as banners and in videos. When users view or click the advertisements, YouTube and its partners (the content creators) earn money, more so when the latter is done.

To attract users into clicking these advertisements, the adverts must show something that the users may be interested in. This is very unlikely to be correct in the first time if the site chose these adverts randomly. What YouTube does instead is make an educated guess based on the user's behavior. They data-mine the user and process as much information that is available as possible.

However, the chosen advert is not guaranteed to be one that the user may be interested in. Thus, another way to increase the probability that an advert may appeal to a user is by having them refresh the page, thereby display a new advert. One of the goals of YouTube's recommendation system is to suggest the users a video that they may enjoy so they stay on the site for longer periods of time.

This data mining process enhances the relationship between YouTube and its users. YouTube plays advertisements for users so they can earn money and users can enjoy a free-to-use service with the occasional chance that they come across an item they might consider purchasing.

Choosing an advertisement is very difficult. It requires an educated guess and an automated process to apply it to as many users as possible. The task is made much more difficult due to anomalies in a users's behavior. A user's video watching history may contain outliers that affect the results of the recommendation and advertising systems. Outliers in YouTube may be caused by the following:

1. **Misclicks:** Users' dexterity is not always accurate and precise. They may intend to click on a video with their pointing device but instead click on another, unrelated one. Although the user can (frustratingly) return to the previous page and try again, the damage has already been done. Depending on the type of video clicked by them, their recommendation and catered advertisements may change significantly. The future YouTube experience of the user may always have remnants of this misclick.
2. **"Clickbait" videos:** One of the contributors to the outliers in users' history is the existence of clickbait videos, which are videos that mislead users into viewing them from sensationalized thumbnails or titles. For instance, a video title indicates that it is a full movie. Upon clicking the video, one can see the duration of approximately one hour, the typical length time for a movie. However, when being viewed, the video consists only of a slideshow of images in a loop accompanied by background music. The users were misled into clicking the video by the content creator. Consequently, the users did not get what they were looking for. The user might become frustrated because their time had been wasted. However, the content creator earns a small sum of earnings, and (most importantly) it shows up in users' history. By showing up in the history, the damage of the clickbait continues as it offsets the algorithms of the advertising and recommendation systems. Many videos like this exist, sporting misleading titles that do not agree with their video contents. This makes finding desired videos a daunting task.
3. **Group activities on a user's account:** Sometimes users may not be using their own accounts. They may share the account with someone else (e.g., with a family devices). Friends and family may request to use a user's device in order to play videos for themselves. It is far too cumbersome to sign-out of one's account, sign-in to their own accounts, and then plays the video. Instead, a user's viewing behavior retains a small portion of someone else's viewing behavior. What these new videos represent are outliers in a user's typical behavior. They present opportunities for the data-mining processes to lose accuracy. The user may never have wanted to watch videos of the kind presented by the group members. This makes it far more difficult to cater recommendations and advertisements to the user, which would ultimately add up in its damages.

Addressing these outlier generation processes represents a task that would ultimately promote YouTube profits. YouTube could then enhance their frameworks with the increase in profits and thus enhance user experience.

Our *goal* is to propose a solution to eliminate videos like these from being used in the search history because these videos are not representative of a user's actual preferences. A logical question to ask is: How to remove these outlier videos? A manual solution would be checking one's own watch history and removing the offending videos. However, this manual solution is not practical because it cannot be automated and there may be many hundreds of videos available to sift through, time that a user may not care to spend. As we cannot modify the YouTube algorithm, we sought to remove the offending videos by mining the user's history and using available metadata in each video to approximate a user's typical behavior. We then found videos in the history that deviated from the behavior and presented them to the user to see how accurate our educated guesses were. This paper reports our findings—i.e., a solution to remove outlier videos.

3. Related work

An *outlier* is an individual value that is inconsistent with the remainder of the set of data. Outlier detection has been used in variety of applications in real life [14-16] ranging from credit card fraud detection, telecom fraud detection, intrusion detection, customer segmentation, detection of criminal activities in E-commerce, etc.

Outlier detection [17-19] has been extensively studied in research and there are many well-known approaches to this problem. The existing approaches include the following:

1. Visual approaches,
2. Depth-based approaches (e.g., [20, 21]),
3. Cluster-based approaches (e.g., [22]),
4. Statistical-based approaches (e.g., [23]),
5. Deviation-based approaches (e.g., [24]),
6. Distance-based approaches (e.g., [25]), and
7. Density-based approaches (e.g., [26, 27]).

In contrast, we utilize a much simpler approach. We determine outliers through a *preference score* based on a number of factors and user preference.

Currently, YouTube seems to be relying on neural networks for their recommendation algorithm [28]. YouTube's core recommendation algorithm is based on the *efficient extreme multiclass*, which is very effective in handling large databases. In their current algorithm, YouTube converts video watches and search token into vectors, which are then converted to Rectified Linear Units (ReLUs) for processing by the neural network. They also take into account the "age" of data in order to recommend more "fresh" content to the users. Outliers are handled by a *softmax function* [28], which is a core element of neural networks. Specifically, the softmax function calculates video recommendation probabilities and stores them in a vector then passes it to the next function which performs a *nearest neighbor index search* for the videos [28]. Outliers sometimes slip through undetected and get recommended to the user. Given the "black box" nature of neural networks, it is not easy—if not near impossible—to determine how the neural network produced its results. In contrast, the aim of our algorithm is to find a more robust way of detecting outliers such that it is easy to trace which outliers are relevant and which are considered as *noise*.

4. Our outlier detection algorithm

In this section, we describe our outlier detection algorithm. This algorithm relies on a *compatibility score* based on metadata available from YouTube videos through the YouTube application programming interface (API). Compatibility is based on the video's similarity to other videos in the user's history. We compute compatibility by retrieving the metadata of each video and applying a custom analysis to each in order to produce a numerical result, which we will later average. We call these metadata attributes *factors*. Later, we will look at each factor and see how it played a role in determining compatibility.

A high-level abstract description, or pseudo code, of our algorithm can be found in Fig. 1. Here, every video is started with a compatibility value of 100%. Then, the impact of every factor's discrepancy is subtracted from this starting value. Some factors are not as important to a user, so we incorporate a third variable, an *internal weight* variable, to each factor. This *internal weight* variable comes into play when we look at the certainty variable.

```
Outlier:ComputeCompatibility(videos[i])
1       compatibility ← 100
2       initialize factor[] & userRatingOnFactor[]
3       for i ← 0 to numVideos do
4               compatibility ← compatibility – (factor[i] × userRatingOnFactor [i] × internalWeight[i])
5       return compatibility
```

Fig. 1.Pseudo code for the computation of the compatibility value.

Next, we will define the *userRatingOnFactor*, which is a certainty score, generated based on explicit feedback from the user. A user rates the factors involved in the compatibility score based on his ideal preferences where the ratings range from 0 to 10. Here, a score of 0 is the lowest and a score of 10 is the highest.

We then compute the average *internalWeight* value (aka *care value* in expressing whether or not the user cares) of each factor to determine the final certainty score. This value is important because we could not be more certain than the users regarding their preferences. However, if they are certain that they would not watch videos outside of their preferences, our compatibility score and certainty value (based on *userRatingOnFactor* and *internalWeight*) reflected this.

Many of the factors were non-numerical values, which required us to standardize and quantify them. Each factor has a corresponding X and Y value:

- X being the compatibility (between 0 and 1), and
- Y being the certainty (between 0 and 1).

Many of these factors also required a statistical average to compare each video to. In the remainder of this section, we will now look at each factor that was considered in the calculation and see how the X and Y values were generated.

4.1. Video quality definition

YouTube records the definition of a video in one of two variables, namely:

1. standard definition (SD), or
2. high definition (HD).

We asked participants how much this value meant to them, but it did not affect the determination of a video's compatibility. Instead, it affected how certain we were that we were correct.

Here, users are expected to spend a majority of the time watch videos that they were inclined towards. So if a user enjoyed HD videos, they would (the majority of the time) watch HD videos. So their preference became the most frequent of either of these two options. This became the statistical reference point.

When analyzing a video, we extract the video definition and compared it to the statistical reference point. If a video had the same video definition as the most frequent one in a user's history, we gave this factor's X value a 1 and a 0 otherwise.

In determining the certainty score, we divided the user's *internalWeight* value for this factor by 10 (the maximum *internalWeight* value). Thus, the Y value would be their definition *internalWeight* value (a value between 0 and 10) divided by 10, allowing it to fall in the range from 0 to 1.

4.2. Video duration

The duration of a video is the running time of the YouTube video. We considered the timings only in minutes, and rounded down to the nearest minute. For example, a video that was 2 minutes and 59 seconds is still a 2 minute video. We disregarded seconds in the calculation for the following reasons:

1. If we considered seconds, there would be too many subsets to consider. For example, 2 minutes 1 second vs. 2 minutes 5 seconds.
2. Dealing with fewer sets simplifies the determination of the mode of the video durations, which would be the user's preference.
3. We assumed the same pattern emerges in users choosing YouTube videos by duration as does pricing, in which users do not consider the ending-digits [29].
4. Videos that were less than 1 minute could be rounded up to 1 minute.
5. Hours would be easily considered as sets of 60 minutes.

Again, we looked at a user's history to find their preference. The majority of the video durations in a user's history should represent the ones that the user prefers, as we assume that outliers only occur some of the time. We found the most frequent duration in minutes from the user's history and called this their preference.

For each video, we compared its duration in minutes to the statistical mode. We found the X value by using the following equation:

$$X = (1 - (0.2 \times \text{thisUserDurationInternalWeightValue}))^{|\text{thisVideoDuration} - \text{modeVideoDuration}|} \quad (1)$$

Some interesting parts of Equation (1) are as follows:

- If the user does not care (i.e., *InternalWeight* value = 0), the X value = 1, which means every video is compatible
- Larger differences in units of time from the mode (e.g., 1 minute, 2 minute, 3 minutes, …) produce exponential differences in the compatibility. This models our assumption that videos 1 minute from the preference are still relatively acceptable, but become decreasingly acceptable the longer/shorter the video is.

Similarly, we define the Y value as being the *InternalWeight* value divided by the maximum *InternalWeight* value of 10.

4.3. Category IDs and channel IDs

A category ID is a unique ID that identifies a broad topic in YouTube. A channel ID identifies a unique user's channel in YouTube.

We applied these factors by counting the frequencies of each category ID in a user's history. We said that a video is *incompatible* (i.e., X = 0) if the frequency of this video's category ID is lower than the frequency of the least frequent category ID of a user's history. Thus, if it was greater, we gave the compatibility score 1.

In a similar fashion, we counted the frequencies of each channel ID in a user's history. We said that a video is *incompatible* (i.e., X = 0) if the frequency of this video's channel ID is lower than the frequency of the least frequent channel ID of a user's history. Thus, if it was greater, we gave the compatibility score 1.

Our focus in this paper was to fine-tune each variable and assume that each user was present in the experiment. We explicitly asked participants for their preferences in each factor and explicitly asked them whether or not an outlier was true. In the case where no explicit feedback is available from the user, we compute the certainty by obtaining the most watched videos based on the user's history. We define a frequency threshold based on how many videos the user watched. If the frequently watched videos reach the threshold then it is granted a rating of 10. In contrast, frequencies below the threshold will be granted a rating proportional to it. We then put all the video in a data structure and graph them according to their compatibility and certainty scores.

4.4. Comments to view ratio

The number of comments was available to incorporate into our analysis. When assessing the number of comments, we divided the comment count by the view count in order to better interpret the values. We wanted to see a ratio between the numbers of comments to views instead of just the comment count on its own, because it did not mean anything on its own. The ratio represents how often a viewer is inclined to comment on the video. Thus, a high comment-to-view ratio suggests a video with a lot of discussion.

In generating the compatibility value, we found the average comment to view ratio in the user's browsing history. We then assessed each video to see if it was below or above this ratio. If it was below, it would receive a compatibility value of 0 (X = 0). If it was above, it would receive a 1 (X = 1). The certainty (Y) was again the ratio of the user's *InternalWeight* value to the maximum *InternalWeight* value of 10.

4.5. Likes to dislikes

The number of likes a video received is the number of times a viewer manually clicked "Thumbs Up". The dislikes is the opposite, using a "Thumbs Down", and represents when a user did not enjoy the video. To users that care, the relative ratio tells viewers whether or not a video will be worth watching or controversial.

In this factor, we divide the number of likes by the total number of likes and dislikes to find the like to total likes (dislikes + likes) ratio. We did this for all the videos in a user's history to find the average likes to total likes ratio.

For each video, we compared its likes to total likes ratio to this average. On the one hand, if it was below this average, we gave it a compatibility score of 0 (i.e., X = 0). On the other hand, if it was above the average, we gave it a compatibility score of 1 (i.e., X = 1). Again, the certainty was the user's *InternalWeight* value over 10.

4.6. Total view counts

We took special considerations when assessing how users interpret view counts and how to incorporate it into our algorithm. This posed a similar problem as the video duration factor because there were potentially many subsets to deal with. We were able to consider it in a similar way, where videos above a certain threshold but below another could belong in the same set.

For example, a video with 5 million views would not be instinctively as different to a user as a video with 6 million views. However, a 500,000 view video would be. So, we grouped videos as follows:

- Videos less than 1,000 views were rounded to 100
- Videos less than 10,000 views were rounded to 1,000
- …
- Videos with less than 10^n views were rounded to 10^{n-1}, where $3 \leq n \leq 7$. Beyond 10^7, videos were all grouped together under 10^8 because we felt that videos past this threshold posed no instinctive difference to users, given the number of digits were overwhelming at first glance.

After categorizing videos in this way, we found the mode of the videos and considered this to be the user's preference. We then applied the following equation to find the compatibility:

$$X = (1 - (0.2 \times \text{thisUsersViewCountInternalWeightValue}))^{|\log_{10}(\text{thisVideoDuration}) - \log_{10}(\text{modeVideoCount})|} \quad (2)$$

Equation (2) is similar to Equation (1) applied in video durations, except that we take the logarithm of the video durations here because we expected the videos to vary by orders of magnitudes. The certainty was found by dividing the user's care value (i.e., *InternalWeight* value) by 10.

4.7. Relevant IDs

A relevant ID is a list of topics that a video covers. A video can cover multiple topics. We assumed that a user would be satisfied with seeing just one of their relevant topics on a video's list, so long as it is reoccurring in their viewing history above a threshold.

We first went through each video and recorded the relevant ID frequencies. We kept track of the smallest frequency and the second smallest. When assessing videos, we would check their relevant topic IDs to see if they fell below the second smallest. We found the second smallest to be important because the outlier would likely be in the smallest categories. We argued that the second smallest relevant ID would likely be intentional because the majority of a user's video history would contribute to this relevant topic ID.

If the video had any relevant topics above the second smallest frequency, we would give it a compatibility score of 1. Otherwise, it would receive a 0.

4.8. Summary

After considering the aforementioned seven factors (including video quality definition, video duration, category IDs and channel IDs, comments to view ratio, likes to dislikes, total view counts, as well as relevant IDs), we averaged the X and Y values. An outlier is one that has a high certainty score (Y) and a low compatibility score (X).

5. Evaluation

In this section, we present our evaluation. Specifically, we describe our way to obtain user data, as well as our methods and calculations for evaluation.

5.1. Obtaining user data

We asked 10 users to provide a hypertext markup language (HTML) file of their watch history. We obtained this data by navigating to YouTube, signing into the user's account, navigating to History on the right tab, scrolling down the user watch history page to load as many watched video as possible, and then right-click "Save as" to save the HTML file that will be later used to conduct the experiment.

We recorded at least 150 entries per user. By employing our own HTML parser, we obtained the unique video IDs and sent these to the YouTube API to extract video metadata such as duration, title, and date watched. We then plugged these values into the algorithm to reap the scores for each video. The scores are sorted on certainty first then compatibility. The top-N videos are displayed as outliers.

5.2. Methods and calculations

The participants were asked to rate the importance of different factors attributed to their watching decisions. This "care value" (i.e., the *InternalWeight*) represents how much the user is concerned with each factor. See Fig. 2.

"Care value" (i.e., *InternalWeight*)	**User 1**	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**User 7**	**User 8**	**User 9**	**User 10**
Video quality definition	10	10	10	8	5	1	8	7	5	9
Video duration	0	6	10	7	2	1	5	7	5	5
Category ID	0	2	0	6	7	6	1	3	2	7
Channel ID	0	0	0	0	0	2	2	5	0	8
Comments to view ratio	0	8	9	1	1	7	6	6	1	4
Likes to dislikes	0	9	8	7	5	8	7	6	4	1
Total view counts	0	2	8	3	1	5	4	6	4	6
…	…	…	…	…	…	…	…	…	…	…

Fig. 2 Preference values for each user ranging from 1-10.

These values from the user ratings were then factored into the impact of the compatibility the equation to yield a score for each video. Compatibility value for each video is computed based on these user preference input using the Compatibility equation. 10 videos with the lowest compatibility value are considered the outliers.

After the algorithm generated 10 outlier videos for each participant, the participants were requested to submit their feedbacks. We used a yes-no question such as

"Did you actually want to watch this video?"

to obtain their feedback for each video. We recorded and converted their polar answers to 0 and 1, in order to calculate support for each user. A score of 1 for a video means that the video is an outlier since the participant does not want to watch it. Meanwhile, a score of 0 for a video means that the video is not an outlier and the participant actually wants to watch the video. Then we calculate the support for each user by calculating the average of their scores. Finally, we calculated the overall support for 10 participants by calculating the average of their support. The experimental results are summarized in Fig. 3.

Using the minimum support of 60% as a benchmark, our overall support of 63.0% indicates that our algorithm is a viable solution in pointing out outliers.

Outlier videos	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	
Video 1	0	1	1	1	0	1	1	0	1	0	
Video 2	0	1	1	1	1	1	0	0	1	0	
Video 3	1	0	0	1	1	1	1	1	0	0	
Video 4	1	0	1	1	1	1	1	0	0	0	
Video 5	1	0	0	0	0	1	1	1	1	0	
Video 6	1	1	1	1	1	0	1	0	1	0	**Minimum support**
Video 7	1	1	0	1	0	1	1	1	1	0	
Video 8	1	1	1	1	1	0	1	1	1	0	60%
Video 9	1	1	1	0	1	1	1	1	0	0	**Overall support**
Video 10	1	1	1	1	0	0	0	1	0	0	
Support	80%	70%	70%	80%	60%	70%	80%	60%	60%	0%	63%

Fig. 3 User feedback specifying the actual outliers from the experimental results of 10 candidate outlier videos.

6. Conclusions

In this paper, we presented an outlier detection algorithm for finding anomalies for YouTube viewing history log so that these anomalies (i.e., noise) can be removed. The resulting cleaned YouTube viewing history log can then be used to enhance prediction of user-preferred next YouTube video.

Acknowledgements

This project is partially supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and University of Manitoba.

References

[1] Cuzzocrea A. Accuracy control in compressed multidimensional data cubes for quality of answer-based OLAP tools. In: *Proceedings of SSDBM 2006*, IEEE; 2006, p. 301-310.
[2] Cuzzocrea A. Privacy and security of big data: current challenges and future research perspectives. In: *Proceedings of CIKM-PSBD 2014*, ACM; 2014, p. 45-47.
[3] Cuzzocrea A, Matrangolo U. Analytical synopses for approximate query answering in OLAP environments. In: *Proceedings of DEXA 2004*, Springer; 2004, p. 359-370.
[4] Braun P, Cuzzocrea A, Leung CK, MacKinnon RK, Tanbeer SK. A tree-based algorithm for mining diverse social entities. *Procedia Computer Science* 2014; **35**: 223-232.
[5] Braun P, Cuzzocrea A, Leung CK, Pazdor AGM, Tran K. Knowledge discovery from social graph data. *Procedia Computer Science* 2016; **96**: 682-691.
[6] Budhia BP, Cuzzocrea A, Leung CK. Vertical frequent pattern mining from uncertain data. In: *Proceedings of the KES 2012*, IOS Press; 2012, p. 1273-1282.
[7] Chowdhury NK, Leung CK. Improved travel time prediction algorithms for intelligent transportation systems. In: *Proceedings of the KES 2011, Part II*, Springer; 2011, p. 355-365.
[8] Cuzzocrea A, Han Z, Jiang F, Leung CK, Zhang H. Edge-based mining of frequent subgraphs from graph streams. *Procedia Computer Science* 2015; **60**: 573-582.
[9] Leung CK, Joseph KW. Sports data mining: predicting results for the college football games. *Procedia Computer Science* 2014; **35**: 710-719.
[10] Leung CK, Mateo MAF, Brajczuk DA. A tree-based approach for frequent pattern mining from uncertain data. In: *Proceedings of the PAKDD 2008*, Springer; 2008, p. 653-661.

[11] Leung CK, Tanbeer SK, Cuzzocrea A, Braun P, MacKinnon RK. Interactive mining of diverse social entities. *International Journal of Knowledge-based and Intelligent Engineering Systems* (*KES Journal*) 2016; **20**(2): 97-111.
[12] Liu L, Özsu M. *Encyclopedia of Database Systems.* 1st ed. New York: Springer; 2009.
[13] Maimon, O, Rokach L. *Data Mining and Knowledge Discovery Handbook*. New York: Springer; 2005.
[14] Hao B, Leung CK, Camorlinga S, Reed MH, Bunge MK, Wrogemann J, Higgins RJ. A computer-aided change detection system for paediatric acute intracranial haemorrhage. In: *Proceedings of the C3S2E 2008*, ACM; 2008, p. 109-111.
[15] Leung CK, Thulasiram RK, Bondarenko DA. An efficient system for detecting outliers from financial time series. In: *Proceedings of the BNCOD 2006*, Springer; 2006, p. 190-198.
[16] Mateo MAF, Leung CK. Design and development of a prototype system for detecting abnormal weather observations. In: *Proceedings of the C3S2E 2008*, ACM; 2008, p. 45-59.
[17] Bansai R, Gaur N, Singh SN. Outlier detection: applications and techniques in data mining. In: *Proceedings of the Confluence 2016*, IEEE; 2016.
[18] Hodge V, Austin J. A survey of outlier detection methodologies. *Artificial Intelligence Review* 2004; **22**(2): 85-126.
[19] Khamis A, Ismail Z, Khalid H, Mohammed A. The effects of outliers data on neural network performance. *Journal of Applied Sciences* 2001; **5**: 1394-1398.
[20] Johnson T, Kwok I, Ng RT. Fast computation of 2-dimensional depth contours. In: *Proceedings of the KDD 1998*, AAAI Press; 1998, p. 224-228.
[21] Montes MC. Depth-based outlier detection algorithm. In: *Proceedings of the HAIS 2014*, Springer; 2014, p. 122-132.
[22] Duan L, Xu L, Liu Y, Lee J. Cluster-based outlier detection. *Annals of Operations Research* 2009; 168(1): 151-168.
[23] Rousseeuw P, Leroy A. *Robust Regression and Outlier Detection*, 3rd ed, John Wiley & Sons, 1996.
[24] Zhang Z, Feng X. New methods for deviation-based outlier detection in large databases. In: *Proceedings of the FSKD 2009*, ACM 2009, p. 495-499.
[25] Knorr EM, Ng RT, Tucakov V. Distance-based outliers: algorithms and applications. *VLDB J* 2000; **8**(3-4): 237-253.
[26] Ester M, Kriegel H, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise, In: *Proceedings of the KDD 1996*, AAAI Press; 1996, p. 226-231.
[27] Faloutsos C, Gibbons P, Kitagawa H, Papadimitriou S. LOCI: fast outlier detection using the local correlation integral. In: *Proceedings of the IEEE ICDE 2003*, IEEE; 2003, p. 315-326.
[28] Covington P, Adams J, Sargin E. Deep neural networks for YouTube recommendations. In: *Proceedings of the ACM RecSys 2016*, ACM; 2016, p.191-198.
[29] Bizer GY, Schindler RM. Direct evidence of ending-digit drop-off in price information processing. *Psychology and Marketing* 2005; **22**(10): 771-783.