



Image Analysis Enhanced Event Detection from Geo-Tagged Tweet Streams

Yi Han^(✉) , Shanika Karunasekera , and Christopher Leckie

School of Computing and Information Systems, The University of Melbourne,
Melbourne, Australia
{yi.han,karus,caleckie}@unimelb.edu.au

Abstract. Events detected from social media streams often include early signs of accidents, crimes or disasters. Therefore, they can be used by related parties for timely and efficient response. Although significant progress has been made on event detection from tweet streams, most existing methods have not considered the posted images in tweets, which provide richer information than the text, and potentially can be a reliable indicator of whether an event occurs or not. In this paper, we design an event detection algorithm that combines textual, statistical and image information, following an unsupervised machine learning approach. Specifically, the algorithm starts with semantic and statistical analyses to obtain a list of tweet clusters, each of which corresponds to an event candidate, and then performs image analysis to separate events from non-events—a convolutional autoencoder is trained for each cluster as an anomaly detector, where a part of the images are used as the training data and the remaining images are used as the test instances. Our experiments on multiple datasets verify that when an event occurs, the mean reconstruction errors of the training and test images are much closer, compared with the case where the candidate is a non-event cluster. Based on this finding, the algorithm rejects a candidate if the difference is larger than a threshold. Experimental results over millions of tweets demonstrate that this image analysis enhanced approach can significantly increase the precision with minimum impact on the recall.

Keywords: Event detection · Autoencoder · Tweet stream mining

1 Introduction

While social media, especially Twitter, has gained growing popularity over the past decade, it has also become a new source of news—events detected from social media streams often contain early signs of accidents, crimes or disasters. Therefore, they can provide valuable information for related parties to take timely and efficient responses.

Although event detection from tweet streams has been extensively studied, most existing methods still suffer from relatively high false positive and false

negative rates, especially for unsupervised machine learning approaches. These algorithms normally rely on semantic, spatial, temporal and frequency information. Images, on the other hand, have rarely been considered yet. Compared with text, especially short posts like tweets, images often provide richer information and potentially can help discover the occurrence of an event.

In this paper, we design an unsupervised event detection algorithm that utilises images in addition to textual and statistical information. The core idea is that when an event occurs, the images posted in the surrounding area are likely to be similar/correlated. Therefore, if we use part of them to train an autoencoder, and keep the rest as the test instances, the reconstruction errors of the training and test images should be close. However, when no event happens, the images posted in a certain region are likely to be more diverse, and hence the reconstruction errors of the test instances will be much higher than those of the training instances, as the autoencoder has not seen similar images before. Based on this idea, the algorithm uses the ratio between the mean reconstruction errors of the test and training images as an additional criterion to further decrease the false positive rate for event detection. Note that since image analysis is relatively expensive, it is only performed at the last step, after the semantic and statistical analyses are finished, which follow a similar approach to [9] with several improvements. In addition, considering that the posted images are normally limited, the algorithm randomly generates the same number of crops for each of them, and trains the autoencoder on the snippets.

In summary, the main contributions of this paper include:

- We analyse images posted in both event and non-event tweet clusters based on the reconstruction errors of autoencoders, and demonstrate that when an event occurs, the images are more coherent (Sect. 2.2);
- We utilise this finding and propose an image analysis enhanced event detection algorithm from tweet streams. It should be emphasised that although we integrate image analysis with a specific existing method [9], the analysis is generic and can be incorporated with other event detection schemes as well (Sect. 2.3);
- We conduct experiments on multiple tweet datasets, and demonstrate that this unsupervised, image analysis enhanced approach can significantly increase the precision without any impact on the recall (Sect. 3).

The remainder of this paper is organised as follows: Sect. 2 specifies the event detection problem, and introduces the image analysis enhanced algorithm; Sect. 3 presents the experimental verification; Sect. 4 overviews previous work on event detection; and Sect. 5 concludes the paper and gives directions for future work.

2 Image Analysis Enhanced Event Detection

In this section, we start with a brief definition of the event detection problem from geo-tagged tweet streams, then introduce in detail how image analysis is performed, and how it is integrated with semantic and statistical analyses.

2.1 Autoencoder Based Image Analysis

We study the event detection problem defined as follows: given a tweet stream $T = \{t_1, t_2, \dots, t_n\}$ from a certain region, and a query window $W = \{t_{n-m+1}, t_{n-m+2}, \dots, t_n\}$ (m is the number of tweets in W) that represents currently observed tweets, the aim is to identify a set of tweets $T_i \subseteq W$ that are associated with an event, e.g., an accident, a disaster or protest, as close to where and when the event occurs as possible.

A common type of solution to the above problem takes the clustering based approach [3, 10, 12, 25–28], which generates a list of event candidates by clustering the tweets according to their semantic, spatial and temporal information, and then removes non-event clusters via supervised or unsupervised methods. In this work, we focus on how image analysis can be used to enhance the second step.

Specifically, suppose that a set of images, $IM = \{im_1, im_2, \dots, im_k\}$, are extracted from an event candidate, *i.e.*, a cluster of tweets that are semantically coherent, and geographically and temporally close, IM is divided into two subsets $IM_{train} \subset IM$, $IM_{test} = IM \setminus IM_{train}$, which are the training and test datasets, respectively. For each image $im_i \in IM$, c random crops of the same size are generated, $\{im_{ij}, j = 1, 2, \dots, c\}$, and $\{im_{ij} \mid im_i \in IM_{train}\}$ are used to train a convolutional autoencoder, while $\{im_{ij} \mid im_i \in IM_{test}\}$ are kept as the test instances.

As mentioned in the introduction, when an event occurs the images in IM are likely to be similar, and hence the reconstruction errors of $\{im_{ij} \mid im_i \in IM_{train}\}$ should be close to those of $\{im_{ij} \mid im_i \in IM_{test}\}$. On the other hand, when there is not any event the difference in the reconstruction errors between the training and test instances should be much larger. Therefore, we propose to quantify the coherence of the images in a cluster, and use that as a metric to detect and remove non-event clusters.

2.2 Quantitative Study

In order to validate the above idea, we collected (part of) the posted images in the following three Twitter datasets:

- Dataset shared by the authors of [28], which includes 9.5 million geo-tagged tweets from New York between 1 August, 2014 and 30 November 2014—617K images are retrieved from it;
- All geo-tagged tweets from Los Angeles between 9 February and 22 February 2019, with a size of 13.2K—20K images are retrieved from it;
- All geo-tagged tweets from Sydney between 12 February and 5 April 2019, with a size of 28.4K—16K images are retrieved from it.

For each dataset, we first perform semantic and statistical analyses using the method in [9] (more details are given in the next subsection) to obtain a list of event candidates. If a candidate contains at least three images, we then (1) randomly generate 500 crops of size 32×32 for each image—there are

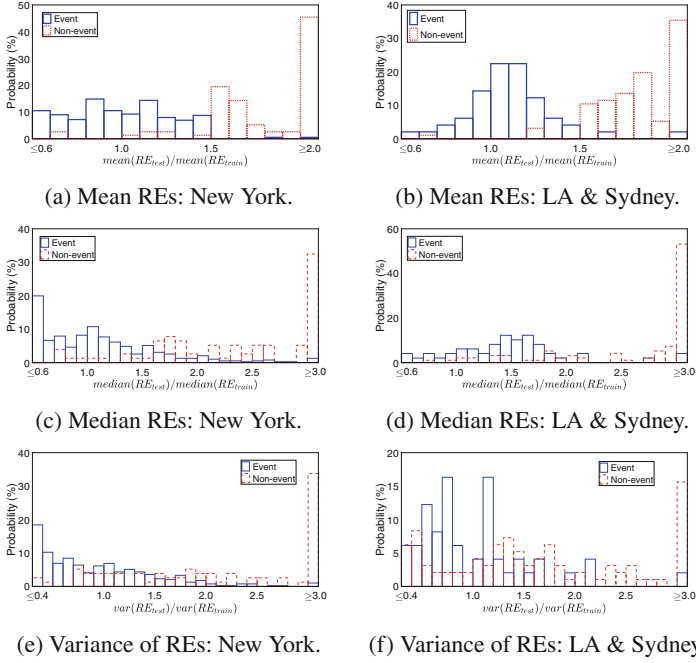


Fig. 1. Probability distributions of $\frac{\text{mean}(RE_{test})}{\text{mean}(RE_{train})}$, $\frac{\text{median}(RE_{test})}{\text{median}(RE_{train})}$, and $\frac{\text{var}(RE_{test})}{\text{var}(RE_{train})}$ for New York, Los Angeles and Sydney. Note that the results for Los Angeles and Sydney are combined due to a relatively smaller amount of data.

normally a limited number of images within each cluster, and they are insufficient for the training of an autoencoder; (2) use two-thirds of the crops to train a convolutional autoencoder, and keep the rest as the test data. Note that all the 500 crops of an image are either in the training or test dataset. In addition, we also notice that if a considerable part of an image is about human beings, the image is often quite different from the rest even if there is an event. For example, during a sports game or a concert, while the focus of most images is the court or the stadium, selfie images are likely to be very different and hence cause false negatives. Therefore, images of this type are excluded in the analysis (see Sect. 3.1 for more details), *i.e.*, each cluster needs to have at least three non-human images in order to be analysed; (3) compare the mean, median and variance of the reconstruction errors (REs, $RE(x) = \|x - x'\|^2$, where x and x' are the input and output of the autoencoder, respectively) for the training and test instances, and calculate the ratios of $\frac{\text{mean}(RE_{test})}{\text{mean}(RE_{train})}$, $\frac{\text{median}(RE_{test})}{\text{median}(RE_{train})}$, and $\frac{\text{var}(RE_{test})}{\text{var}(RE_{train})}$, where RE_{train} and RE_{test} represent the set of training and test REs, respectively.

Figure 1 shows the probability distributions of these three ratios for (manually labelled) event and non-event clusters obtained after the semantic and

statistical analyses. Note that the results for Los Angeles and Sydney are combined due to a relatively smaller amount of data. It is clear from these figures that when a candidate corresponds to a non-event, all the three ratios are distinctively higher in general, which indicates the images are more diverse. Specifically, we find that $\frac{\text{mean}(RE_{test})}{\text{mean}(RE_{train})}$ gives the best performance. Hence, it is selected in our experiment, and the threshold is set to be 1.5. More formally, denoting the reconstruction error of the autoencoder for input im_{ij} by $RE(im_{ij})$, we define the following metric to measure the coherence of the images in IM :

$$R = \frac{\overline{RE_{test}}}{\overline{RE_{train}}} = \frac{\sum_i \sum_{j=1}^c RE(im_{ij}), im_i \in IM_{test}/|IM_{test}|}{\sum_i \sum_{j=1}^c RE(im_{ij}), im_i \in IM_{train}/|IM_{train}|} \quad (1)$$

2.3 Algorithm Description

As mentioned earlier in the above section, for semantic and statistical analyses we adopt the similar method to [9], which works as follows (Algorithm 1):

- *Building a Quad-tree (QT)* [8,17] for the sliding windows. The root of QT represents the whole region, and if the number of tweets in the sliding windows is larger than a pre-defined threshold, the region is divided into four equally-sized sub-regions. The process continues until the number of tweets in each leaf node is smaller than or equal to the threshold, or the depth of QT reaches the maximum value. It should be emphasised that once the Quad-tree is built, **the detection will be run at all levels**, in order to mitigate the impact of the arbitrary division of space.
- *Embedding*. Entities and noun phrases from each tweet are extracted using the NLP tool [16] mentioned in [28]. These keywords are then embedded with the fastText algorithm [5], and each tweet is represented by the average value of all its keyword vectors. Note that the temporal and spatial information is not included in the embedding, as the similarities in time and space are ensured by the sliding window and the Quad-tree.
- *Clustering*. The generated vectors are clustered using the algorithm of BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [29].
- *Power-law Detection*. The study in [9] finds that when an event occurs, it is much more likely to observe power-law distributions in tweet streams. Based on this finding, we run power-law detection [6,21] within each cluster. Note that the clustering is only done at the root level of QT against all tweets in the sliding windows, but the power-law detection is run at all levels, so that the event can be identified as close to where it occurs as possible. For example, suppose that cluster A is formed at the root level (Level 0), it is divided into A_1, A_2, A_3, A_4 at Level 1, each of which is further divided into four sub-clusters at Level 2 and so on. Power-law detection is done in each of these clusters.
- *Verification*. For each remaining cluster that passes the power-law detection, we collect additional tweets from the verification window, which is set to 5 min in our experiment, and repeat the last three steps. The only difference

is that when vectorising the tweet, the original text is directly embedded to make sure that both the keywords and texts are semantically close within a cluster. Each remaining event candidate is then checked against each cluster found in this step. If any two of them share more than half of the tweets, they are considered as a match. Otherwise the candidate is removed. The verification process is done twice.

While the above steps are similar to [9], we modify and add the following steps (see Fig. 2 for an illustration):

- *Pruning.* We extract all hashtags and mentions for each remaining cluster, and remove a tweet if it contains hashtags and/or mentions, but all of them either (1) only appear once in the cluster, (2) appear only in one tweet, or (3) are excluded keywords—including commonly used stop words, names of the city, state and country for the examined region, etc. Then we identify the top $X(= 5)$ hashtags and mentions, and reject an event candidate if less than half of the remaining tweets contain any of them.
- *Image Analysis.* If a cluster passes all the above tests and has at least three (non-human) images, we perform image analysis as described in Sect. 2.2 for each of them. One point worth noticing is that an image is only considered if it is posted in a tweet that contains at least one of the top $X(= 5)$ hashtags or mentions. It is found in our experiments that this can make the prediction more accurate. Finally, we calculate the ratio R as defined in Eq. (1) and reject a candidate if $R \geq 1.5$.

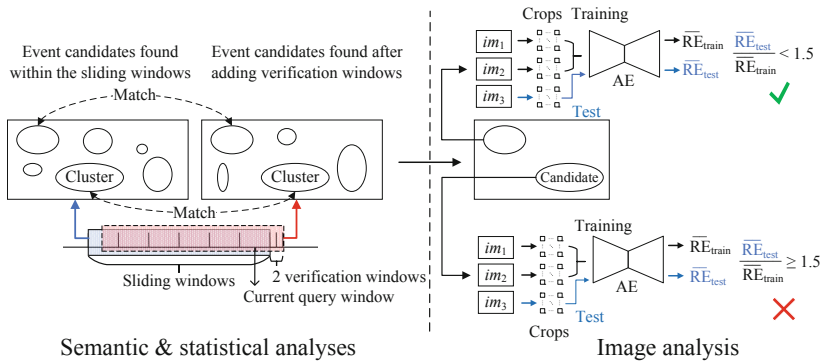


Fig. 2. An illustration of the image analysis enhanced event detection algorithm

3 Experimental Evaluation

In this section, we present the results on the three datasets as described in Sect. 2.2 to test the effectiveness of the image analysis enhanced event detection algorithm.

3.1 Experimental Setup

Baseline Algorithms. The following two methods are chosen as the baselines: (1) Geoburst [28], a widely cited event detection algorithm that considers temporal, spatial and semantic information. Although improved versions exist (Geoburst+ [26], TrioVec [27]), we do not use them as baselines in this work as they are supervised approaches, while both Geoburst and our method use unsupervised approaches; (2) Power-law advanced [9] that combines fastText, BIRCH, and power-law verification as introduced in the last section. Note that Power-law advanced is unsupervised as well.

Parameters. (1) All the parameters for Geoburst take the default values in the code shared by the author. (2) For Power-law advanced, (i) a pre-trained fastText model is used, and it is re-trained incrementally [15] with the new tweets in the last 24 h. Since the re-training is done in parallel, it does not delay the detection; (ii) the threshold of the cluster radius is the most important parameter in BIRCH. We do not set its value arbitrarily. Instead, we start with a value close to zero, and increase it by a small step size until either less than 5% of all items are in small clusters, *i.e.*, clusters with a size less than 10, or over half of the items are in the largest cluster, whichever occurs first; (iii) the Quad-tree has a maximum depth of 30, and each node can hold up to 50 tweets; (iv) the sliding windows keep the latest six query windows, each of which is 30 min.

In addition, as described in Sect. 2.2, an image is excluded in the image analysis if a considerable part of it is about human beings. In our experiment, we reject an image if a total of 40% of the area is detected as humans, or if a person takes up over 20% of the size. Note that since we are only interested in detecting humans in an image, the pre-trained models provided in [2] can be used directly and do not need to be re-trained. Specifically, “ssdlite_mobilenet_v2_coco” is chosen in our experiment.

3.2 Quantitative Analysis

Figure 3 presents the performance comparison between the three event detection algorithms. The result demonstrates that our image analysis enhanced approach can significantly increase the precision without any impact on the recall. One reason why the recall is not affected is that the detection is run at all levels of the Quad-tree, so even if an event candidate is rejected, the same event can be detected at a different level.

Note that when calculating the precision for Power-law advanced and our image analysis enhanced method, duplicated events—same events that are detected at different levels of the Quad-tree, or in consecutive query windows—are merged together. The precision will be much higher (over 10% higher) if we use the raw data directly.

Note also that since the ground truth of the three datasets are not given, it is difficult to calculate the true recall. Therefore, we adopt a similar approach as in [26, 27] and calculate the *pseudo recall* = N_{true}/N_{total} , where N_{true} is the number of true events detected by a method, and N_{total} is the number of true

Algorithm 1. Image analysis enhanced event detection algorithm

Input : Geo-tagged tweets in the query window, W ; Maximum depth of the Quad-tree (QT), D ; Threshold for splitting a node in QT , m_s

Output : Event list, E

- 1 **Build Quad-tree**
- 2 Create an empty Quad-tree QT ;
- 3 **for** tweet t in W **do**
- 4 **if** child nodes $\neq NULL$ **then**
- 5 Insert t into one of the child nodes based on t 's coordinates;
- 6 **else if** the number of tweets in the current node $\geq m_s$ & QT 's depth $< D$ **then**
- 7 Split into four nodes; move all tweets into one of them based on coordinates;
- 8 **else**
- 9 Insert t into the current node;
- 10 **Embedding**
- 11 Extract entities and noun phrases using the NLP tool [16] for each tweet;
- 12 Call fastText to embed the extracted keywords;
- 13 **Clustering**
- 14 Cluster the generated vectors using BIRCH;
- 15 **Power-law detection**
- 16 **for** Cluster C found in the last step **do**
- 17 $E \leftarrow$ Power-law detection at different layers of QT ;
- 18 **Verification**
- 19 **for** $i = 0$; $i < 2$ & E is not $NULL$ **do**
- 20 Call fastText to directly embed the text of each tweet;
- 21 Cluster the generated vectors using BIRCH;
- 22 **for** Cluster C' found in the last step **do**
- 23 $E' \leftarrow$ Power-law detection at different layers of QT ;
- 24 **for** Remaining event candidate $e \in E$ **do**
- 25 Remove e if there is no match in E' ;
- 26 **Pruning**
- 27 **for** Remaining event candidate $e \in E$ **do**
- 28 Remove a tweet if none of its hashtag/mention appears in other tweet, or is not an excluded keyword;
- 29 Remove e if $\geq 50\%$ tweets does not contain any top $X = 5$ hashtag/mention;
- 30 **Image analysis**
- 31 **for** Remaining event candidate $e \in E$ **do**
- 32 **if** e has at least three non-human images **then**
- 33 Train an autoencoder with 2/3 of the crops generated from each image;
- 34 Calculate the ratio R and remove e if $R \geq 1.5$
- 35 **return** E

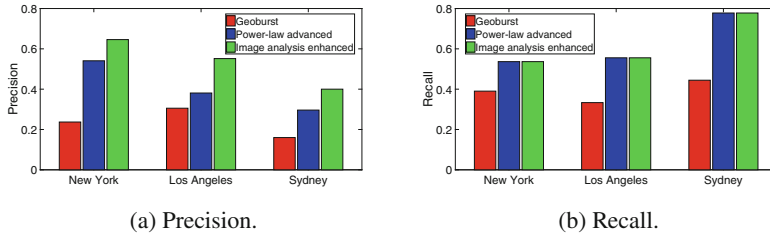


Fig. 3. Performance comparison of the three event detection algorithms

events detected by all methods, plus the events hand-picked by us that occurred during the query periods within the chosen cities, including protests, ceremonies, sport games, natural disasters, etc.

Discussion on Efficiency. The proposed image analysis mainly contains three parts: using the object detector to remove images of human beings, training a convolutional autoencoder, and feeding all the training and test instances to the autoencoder to obtain the reconstruction errors.

The following approaches are taken to minimise the time for image analysis: (1) it is performed only at the last step after the semantic and statistical analyses are finished. In over 95% of our experiments, less than 10 clusters/event candidates are able to reach the last step in one round of detection; (2) as mentioned in Sect. 2.3, an image is only considered if it is posted in a tweet that contains at least one of the top $X(=5)$ keywords. This largely decreases the number of images to be examined; (3) since the analysis of a cluster is independent of each other, it can be done in parallel.

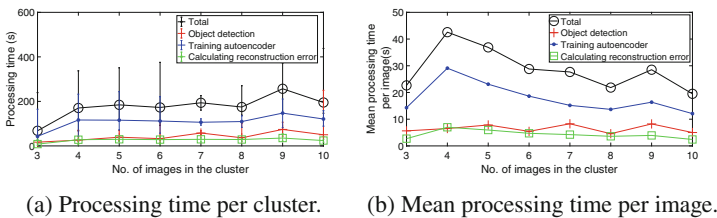


Fig. 4. Efficiency of the image analysis.

Figure 4 shows the processing time of the image analysis for around 240 event candidates in the Los Angeles dataset (results on the other two datasets are omitted due to similarity), including the total processing time over the entire cluster, and the time for each of three main operations. We can see that (1) the training of the autoencoder takes up more than half of the time, (2) the total processing time grows rather slowly with the number of images within the

cluster, and in the majority cases the image analysis can be finished within 200 seconds. Considering that the detection is run every 30 min, the image analysis for each cluster can be done in parallel, and that GPUs are not used in the experiment, the overhead is acceptable.

4 Related Work

This section briefly reviews the previous work on event detection from social media. We start with the work that has considered images for event detection, and then summarise two types of commonly used algorithms: clustering based and anomaly based [14].

4.1 Fusion of Text and Image for Event Detection

Although images have been used in domains such as event detection from videos and fake news detection, only a limited number of studies have used both text and images for event detection from social media streams. In addition, the image is also used in a very different way from ours. For example, Alqhtani *et al.* [13] extract three types of features from images, including Histogram of Oriented Gradients descriptors, Grey-Level Co-occurrence Matrix and color histogram, which are then combined with features extracted from text to train a Support Vector Machine for event detection. In another example, Kaneko and Yanai [11] propose a method to select images from tweet streams for detected events. Specifically, the images are clustered based on densely sampled speeded-up robust features (SURF) and 64-dimensional RGB color histograms. Visually coherent images are then selected according to the keywords extracted from the text.

4.2 Clustering Based Event Detection

This type of detection method takes a two-step approach [3, 4, 10, 12, 22, 23, 25–28]. First, tweets are clustered based on their temporal, spatial, semantic, frequency and user information. However, since the generated clusters may correspond to non-events, a second step is taken to eliminate false positives. For example, for each pair of tweets, Geoburst [28] measures their geographical and semantic impact based on the Epanechnikov kernel and the random-walk-with-restart algorithm, respectively. In this way, they obtain a list of clusters that are geographically close and semantically coherent, *i.e.*, event candidates. Finally, these candidates are ranked according to historical activities, and the top K events are returned. In terms of the improved versions: (1) Geoburst+ [26] adopts a supervised approach, and builds a candidate classification module, which learns the latent embeddings of tweets and keywords; then together with the activity timeline, the module extracts spatial unusualness and temporal burstiness to characterise each candidate event; (2) TrioVecEvent [27] learns multimodal embeddings of the location, time and text, and then performs online clustering using a Bayesian mixture model.

4.3 Anomaly Based Event Detection

This type of method [7, 18–20, 24] aims to identify abnormal observations in word usage, spatial activity and sentiment levels. For example, Vavliakis *et al.* [20] propose event detection for the MediaEval Benchmark 2012 [1] based on Latent Dirichlet Allocation. They detect peaks in the number of photos assigned to each topic, and identify an event for a topic if it receives an unexpectedly high number of photos. Another example is using a Discrete Wavelet Transformation [7] for the detection of peaks in Twitter hashtags, which are likely to correspond to real-world events. Specifically, only the hashtags are used, and all the remaining tweet text is discarded.

5 Conclusions and Future Work

In this paper, we propose an event detection algorithm that combines textual, statistical and image information. It generates a list of tweet clusters after the semantic and statistical analyses, and then performs image analysis to separate events from non-events. Specifically, a convolutional autoencoder is trained for each cluster, where a part of the images are used as the training data and the rest are kept as the test instances. When an event occurs, since the images posted in the surrounding area are more likely to be coherent, the reconstruction errors between test and training images will be closer. The algorithm utilises this as an additional criterion to further remove non-event clusters. Experimental results over multiple datasets demonstrate that the image analysis enhanced approach can significantly increase the precision without any impact on the recall.

For future work, we will improve the effectiveness of the image analysis. For example, currently each crop of an image is feed into the convolutional autoencoder independently, and we intend to find a way that can “stitch” them together. In addition, we will also explore other measurements of the reconstruction errors rather than the mean value to quantify the coherence of the images in a cluster.

Acknowledgements. This research is funded in part by the Defence Science and Technology Group, Edinburgh, South Australia, under contract MyIP:7293.

References

1. MediaEval 2012 (2019). <http://www.multimediaeval.org/mediaeval2012/>
2. Tensorflow/models (2019). <https://github.com/tensorflow/models>
3. Abdelhaq, H., Sengstock, C., Gertz, M.: EvenTweet: online localized event detection from twitter. *Proc. VLDB Endow.* **6**(12), 1326–1329 (2013). <https://doi.org/10.14778/2536274.2536307>
4. Becker, H., Naaman, M., Gravano, L.: Beyond trending topics: real-world event identification on twitter. In: *ICWSM 2011* (2011)
5. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. [arXiv:1607.04606](https://arxiv.org/abs/1607.04606) [cs] (2016)

6. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. *SIAM Rev.* **51**(4), 661–703 (2009)
7. Cordeiro, M., Frias, R.: Twitter event detection: combining wavelet analysis and topic inference summarization. In: *Doctoral Symposium on Informatics Engineering*, DSIE (2011)
8. Finkel, R.A., Bentley, J.L.: Quad trees a data structure for retrieval on composite keys. *Acta Inf.* **4**(1), 1–9 (1974). <https://doi.org/10.1007/BF00288933>
9. Han, Y., Karunasekera, S., Leckie, C., Harwood, A.: Multi-spatial scale event detection from geo-tagged tweet streams via power-law verification. In: *Accepted by IEEE Big Data 2019* (2019)
10. Hasan, M., Orgun, M.A., Schwitter, R.: Real-time event detection from the twitter data stream using the TwitterNews+ framework. *Inf. Process. Manage.* **56**(3), 1146–1165 (2019). <https://doi.org/10.1016/j.ipm.2018.03.001>
11. Kaneko, T., Yanai, K.: Event photo mining from twitter using keyword bursts and image clustering. *Neurocomput.* **172**, 143–158 (2016). <https://doi.org/10.1016/j.neucom.2015.02.081>
12. Li, R., Lei, K.H., Khadiwala, R., Chang, K.C.C.: TEDAS: a Twitter-based event detection and analysis system. In: *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, pp. 1273–1276. ICDE 2012, IEEE Computer Society (2012). DOI: <https://doi.org/10.1109/ICDE.2012.125>
13. Alqhtani, M., Luo, S., Regan, B.: Fusing text and image for event detection in Twitter. *Int. J. Multimedia Appl.* **7**(1), 27–35 (2015). <https://doi.org/10.5121/ijma.2015.7103>
14. Panagiotou, N., Katakis, I., Gunopulos, D.: Detecting events in online social networks: definitions, trends and challenges. In: Michaelis, S., Piatkowski, N., Stolpe, M. (eds.) *Solving Large Scale Learning Tasks. Challenges and Algorithms*. LNCS (LNAI), vol. 9580, pp. 42–84. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41706-6_2
15. QinLuo: Library for fast text representation and classification: ericxsun/fastText (2019). <https://github.com/ericxsun/fastText>
16. Ritter, A.: Twitter NLP tools. contribute to aritter/twitter_nlp development by creating an account on GitHub (2011). https://github.com/aritter/twitter_nlp
17. Samet, H.: The quadtree and related hierarchical data structures. *ACM Comput. Surv.* **16**(2), 187–260 (1984). <https://doi.org/10.1145/356924.356930>
18. Valkanas, G., Gunopulos, D.: Event detection from social media data. *IEEE Data Eng. Bull.* **36**(3), 51–58 (2013)
19. Valkanas, G., Gunopulos, D.: How the live web feels about events. In: *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 639–648. CIKM 2013, ACM (2013). DOI: <https://doi.org/10.1145/2505515.2505572>
20. Vavliakis, K.N., Tzima, F.A., Mitkas, P.A.: Event detection via LDA for the MediaEval2012 SED task. In: *MediaEval* (2012)
21. Virkar, Y., Clauset, A.: Power-law distributions in binned empirical data. *Ann. Appl. Stat.* **8**(1), 89–119 (2014)
22. Walther, M., Kaisser, M.: Geo-spatial event detection in the Twitter stream. In: Serdyukov, P., et al. (eds.) *Adv. Inf. Retrieval*, pp. 356–367. Springer, Berlin Heidelberg (2013)

23. Wei, H., Zhou, H., Sankaranarayanan, J., Sengupta, S., Samet, H.: Detecting latest local events from geotagged tweet streams. In: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 520–523. SIGSPATIAL 2018, ACM (2018). DOI: <https://doi.org/10.1145/3274895.3274977>
24. Xia, C., Hu, J., Zhu, Y., Naaman, M.: What is new in our city? a framework for event extraction using social media posts. In: Cao, T., Lim, E.-P., Zhou, Z.-H., Ho, T.-B., Cheung, D., Motoda, H. (eds.) PAKDD 2015. LNCS (LNAI), vol. 9077, pp. 16–32. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18038-0_2
25. Xie, W., Zhu, F., Jiang, J., Lim, E., Wang, K.: TopicSketch: real-time bursty topic detection from Twitter. *IEEE Trans. Knowl. Data Eng.* **28**(8), 2216–2229 (2016). <https://doi.org/10.1109/TKDE.2016.2556661>
26. Zhang, C., et al.: GeoBurst+: Effective and real-time local event detection in geo-tagged tweet streams. *ACM Trans. Intell. Syst. Technol.* **9**(3), 341–3424 (2018). <https://doi.org/10.1145/3066166>
27. Zhang, C., et al.: TrioVecEvent: embedding-based online local event detection in geo-tagged tweet streams. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 595–604. KDD 2017, ACM (2017). DOI: <https://doi.org/10.1145/3097983.3098027>
28. Zhang, C., et al.: GeoBurst: real-time local event detection in geo-tagged tweet streams. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 513–522. SIGIR 2016, ACM (2016). DOI: <https://doi.org/10.1145/2911451.2911519>
29. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An efficient data clustering method for very large databases. In: Proceedings of ACM SIGMOD 1996, pp. 103–114. ACM (1996). DOI: <https://doi.org/10.1145/233269.233324>