Event Embedding Learning from Social Media Using Graph Topic Model Autoencoder

Yihong Zhang and Takahiro Hara

Multimedia Data Engineering Lab, Graduate School of Information Science and Technology, Osaka University, Osaka, Japan
yhzhang7@gmail.com, hara@ist.osaka-u.ac.jp

Abstract. Event detection from social media has been researched intensively and applied to many social problems such as disaster monitoring, rumor detection, and product sales prediction. In this paper, we deal with the underlying task of event representation. Existing works use topic modeling or graph embedding to extract events from text corpora, but have not fully utilized the available information. We propose a novel model called Graph Topic Model Autoencoder (GTMA) for improving event representation quality. The model combines non-negative matrix factorization and graph autoencoder to take advantages of the document-word matrix and the word-word co-occurrence graph. The model outputs event embeddings that can be used for topic-based event detection. We test our approach with two real-world social media datasets. Compared to several embedding learning baselines, our model generally achieves better topic quality in event detection evaluation.

Keywords: event representation, social media mining, graph topic model

1 Introduction

In recent years, social media is shown to provide rich information on societal events. Twitter, a popular social media platform, for example, currently has 330M monthly active users, and produces 350K tweets per minute[1]. These tweets cover topics of all kinds of social activities which can be helpful for many social computational tasks [19]. Researchers have been trying to develop systems that can extract events from this massive data. Generally, an event can be seen as an unusual change in activity trends [18], sometimes with geographical information [20].

Topic modeling, the technique for generating topics from a text corpus, has been widely studied in data mining and information retrieval. Latent Dirichlet Allocation (LDA) [2], one of the most well-known topic modeling techniques, has spanned many variations [5]. Particularly, some online variations have been deployed in social media event detection [1, 13]. This technique models a document as a distribution of topics, and a topic as a distribution of words [10].

[1] https://www.bankmycell.com/blog/how-many-users-does-twitter-have

Non-negative matrix factorization (NMF) is a technique that can achieve a similar effect [15]. NMF uses the product of two low-rank matrices to approximate document-word data, and the two matrices can be seen as document-topic distribution and topic-word distribution. Because they are very similar in effect, several studies have compared the two techniques directly in various experiment settings [4]. Findings in these studies show that NMF is equivalent to and sometimes better than LDA in terms of topic quality. Because NMF is relatively new in topic modeling, and has a higher potential to be developed in a neural network framework, we choose this approach to further study in our work.

Our insight through analyzing social media data is that social media text streams can be represented as graphs. We can create a document-word graph, where the nodes are documents and words, and an edge indicates word frequency in a document. We can also create a word-word graph, where nodes are words, and an edge indicates the frequency two words co-occur in a document. The event information, which is represented by abnormal clusterings of words, is contained in these graphs. Once we represent social media data as graphs, we can use popular graph embedding techniques to encode event information in embeddings. For example, the well-known technique, TransE, and its variations, can learn graph embeddings through translation geometric [3, 14]. However, it does not cope with weighted edges. Graph convolutional network (GCN) [16] is a popular graph embedding technique that can work with weighted edges. However, GCN requires label data to learn the model, and thus is not unsupervised. Another graph embedding technique, graph auto-encoder (GAE) [6], on the other hand, is unsupervised, and can be used easily in topic modeling.

Using graph embedding techniques in topic modeling is largely unexplored. In this paper, we make an initial step by proposing a model that combines topic modeling and graph embedding techniques. Our model, called Graph Topic Model Autoencoder (GTMA), combines the advantages of both document-word graph and word-word co-occurrence graph, and thus can achieve better results than existing topic model and graph embedding techniques. We use two real-world social media datasets to test our model, evaluating event detection accuracy. As the main contributions, we first propose a method to convert social media data stream into graphs, which preserves critical event information. This conversion makes way for applying advanced graph embedding techniques in event detection. Then we propose an event embedding learning model that combines topic modeling and graph embedding techniques. While relatively simple, this model takes advantages of both document-word graph and word-word co-occurrence graph, and is expected to achieve better results. Finally, we test our approach in two real-world datasets and discuss the results.

2 Related Work

We focus on introducing existing works that are technically most related to our proposed model. They are divided into two groups, namely, topic models, and graph embedding techniques. LDA [2] is a well-known topic model that produces

document-topic distribution and topic-word distribution. It has many variations. For example, the bi-term topic model (BTM) [5] adds a special consideration of bi-terms to improve topic quality. The online BTM (OBTM) [5], proposed by the same authors, is an extension that stores only recent data when processing incoming new data. NMF [15] can achieve a similar effect as LDA. It also has several variations. For example, online NMF (ONMF) [8] uses sparse coding to make new latent factors converge to the past when processing new data. Evolve NMF (EvNMF) and Emerge NMF (EmNMF) are proposed together in the same work [12]. EvNMF uses a parameter to make NMF conform to a past model, while EmNMF is made to capture as many new anomalies as possible.

The representative graph embedding technique, as we mentioned, is GCN [16]. GCN also has several variations, some of which are suitable for continuous event detection. For example, Evolve GCN (EvGCN) [9] connects model parameter to past models through recurrent neural network, achieving an smooth effect. Frequency-weighted GCN (FW-GCN) [17] combines frequency changes with GCN embeddings to capture both node semantics and temporal anomalies. GCN-based methods usually perform pseudo-tasks such as link prediction when learning the model. Graph Autoencoder (GAE) [6], in contrast, reconstructs input data, and uses the difference for learning the model, and thus is fully unsupervised.

3 Preliminaries

Our method is based on two established graph embedding learning techniques, namely, non-negative matrix factorization (NMF) [15, 12], and graph autoencoder (GAE) [6]. In this section, we will briefly review these two techniques.

3.1 Non-negative Matrix Factorization

NMF is a technique that is widely used in document clustering. Usually, we have an input matrix $\mathbf{X}$ that represent document-term frequency, formulated as:

$$\mathbf{X}_{ji} = tf_{ji} \cdot \log\left(\frac{n}{idf_j}\right) \tag{1}$$

where tf_{ji} is the frequency of word j in document i, and idf_j is the inverse document frequency of word j. This matrix can be also seen as a weighted graph, where values are edge weights between each pair of document and word.

Given that the number of documents is N, and the size of the vocabulary is D, NMF then tries to find two low-rank matrices $\mathbf{W}$ and $\mathbf{H}$, whose product can approximate the input matrix. To learn these two matrices, the technique tries to minimize a reconstruction loss:

$$\mathcal{L}_{NMF} = \frac{1}{2}||\mathbf{X} - \mathbf{WH}|| \quad s.t. \quad \mathbf{W}, \mathbf{H} \geq 0. \tag{2}$$

Matrix $\mathbf{W}$ of size $N \times K$ can be seen as a latent representation or embeddings of documents, while Matrix $\mathbf{H}$ of size $K \times D$ can be seen as a latent representation or embeddings of the words.

3.2 Graph Autoencoder

GAE is a technique for learning graph embeddings based on the graph data structure [6]. A graph can be represented as an adjacency matrix $\mathbf{A}$ of size $D \times D$, where D is the number of nodes. In a weighted graph, $\mathbf{A}_{ij}$ is the edge weight between node i and node j. GAE incorporates an encoder-decoder structure. The encoder outputs a latent matrix $\mathbf{Z}$ through graph convolution [7]:

$$\mathbf{Z} = \mathrm{GCONV}(\mathbf{I}, \mathbf{A}) \tag{3}$$

where the graph convolution layer GCONV is defined as

$$\mathrm{GCONV}(\mathbf{I}, \mathbf{A}) = \tilde{\mathbf{A}}\mathrm{ReLU}(\tilde{\mathbf{A}}\mathbf{I}\mathbf{W}_0)\mathbf{W}_1.$$

where $\tilde{\mathbf{A}}$ is normalized $\mathbf{A}$.

In some versions of GAE, the graph convolution can take into account node features [7]. In our work, we assume that no node feature is available. In this case, the identity matrix $\mathbf{I}$ can be used instead of the node feature matrix.

The decoder reconstruct the input from the latent matrix:

$$\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^T) \tag{4}$$

where $\sigma(\cdot)$ is a logistic sigmoid activation function.

Learning the model involves minimizing both the reconstruction loss and the normalization loss:

$$\mathcal{L}_{GAE} = ||\hat{\mathbf{A}} - \mathbf{A}||^2 - \mathrm{KL}[q(\mathbf{Z}|\mathbf{I}, \mathbf{A})||p(\mathbf{Z})] \tag{5}$$

where KL is the Kullback-Leibler divergence, $q(\cdot)$ and $p(\cdot)$ are the probabilities based on a Gaussian distribution. We note that in the original paper of GAE [6], the authors proposed a more complex version called variational GAE (VGAE), which allows a probabilistic definition of the latent matrix $\mathbf{Z}$. After experimenting, however, we find that VGAE tends to generate less stable embeddings. So in this paper, we use the basic version of GAE.

4 Graph Topic Model Autoencoder for Event Embedding Learning

We propose a method to covert raw social media data stream into event embeddings. Because we follow a graph-based approach, we need to first convert social media data into graphs. Then we generate embeddings using a novel graph-based technique. In this section, we will present our method in detail.

4.1 Generating Event Graph from Social Media

We first convert raw social media data stream into graphs. The raw social media data stream is defined as the following. We first set a time unit length s. As we

collect data, we obtain data in a number of time units, $\mathbf{C} = \{C_1, ..., C_t\}$. Each C_t is a collection of short texts, $C_t = \{c_1, ...c_{N_t}\}$, collected in the time period $[t, t+s]$. The number of short texts, N_t, is variable in different time units. The vocabulary size, D, is assumed to be fixed for all time units.

We can construct two graphs from C_t. The first is document-word graph. We use the same approach as the NMF to construct the graph, which we described in Equation (1). This graph is denoted as G^{DW}, where each G^{DW}_{ji} is the tf-idf value of word j in document i.

The document-word graph comes naturally with a collection of short text documents. However, this graph may generate some noises by confining words to documents. For example, one account may spam some similar text in one time period. The document-word graph may capture this anomaly, but it is really not related to overall events in the period. In contrast, the word-word co-occurrence graph captures overall events by removing the confinement of documents. The word-word co-occurrence graph is simple to construct. This graph, denoted as G^{WW}, is a $D \times D$ matrix, where each value G^{WW}_{ij} is the number of documents word i and word j co-occur:

$$G^{WW}_{ij} = |C_{co}|, \forall (w_i, w_j) \in C_{co}, C_{co} \subseteq C_t \tag{6}$$

Above two graphs G^{DW} and G^{WW} can capture anomalies in word frequency and word co-occurrence in the time unit, thus the event information. After constructing these two graphs, we use a unified graph embedding technique to further refine the information present in the graphs. As a topic model approach, we manually define the number of topics as K.

4.2 Graph Topic Model Autoencoder

We propose a model called Graph Topic Model Autoencoder (GTMA) that takes advantages of both the document-word graph and word-word co-occurrence graph. Our model combines the approaches of NMF and GAE. Both approaches are unsupervised approaches based on data reconstruction, so our model is also unsupervised. Both methods reconstruct the graph by processing the original graph with latent factors. We let the NMF model process G^{DW}, and the reconstruction is done by calculating $\mathbf{WH}$, the product of the document latent factor and the word latent factor. The word latent factor $\mathbf{H}$ is usually used for topic modeling and event detection. We then let the GAE model process G^{WW}, reconstructing the graph using Equation (4). The latent factor $\mathbf{Z}$ in the equation is generated using the encoder defined in Equation (3). We find that this latent factor $\mathbf{Z}$, of the size of $K \times D$, can also be used in topic modeling and event detection.

We combine the latent factor from two models to form the final event embedding of our model. Like existing studies on topic modeling, the latent factors are processed by softmax, so that the value in each topic is summed to 1. Formally, each row of our event embedding matrix $\mathbf{E}$ is calculated as:

$$\mathbf{E}_{[i,:]} = \text{softmax}(\mathbf{H}_{[i,:]}) + \text{softmax}(\mathbf{Z}_{[i,:]}) \tag{7}$$

This embedding matrix $\mathbf{E}$ is our final output, which can then be used for event detection and other downstream tasks.

4.3 Temporal Smoothing

Our model described in the above section captures events in each time unit, treating text messages in each time unit as an independent corpus. This model has some disadvantages. First, events are often continuous. Treating each time unit as independent cannot capture the evolution of events. Second, treating each time unit as independent makes the model too sensitive to the anomaly in the time unit. Particularly, the model will be vulnerable to spams in the time unit. As such, following existing approaches, we perform temporal smoothing on the model output. We use a loss function for minimizing the difference between current event embedding and event embedding of the past time unit:

$$\mathcal{L}_{smooth} = \gamma||\mathbf{E}_t - \mathbf{E}_{(t-1)}||_2 \tag{8}$$

where $||\cdot||_2$ is the L2 norm, and γ is a scaling factor. In this way, we enforce current event embedding to conform to past embedding, capturing event evolution and removing noises.

4.4 Model Learning

We follow the standard machine learning approach to learn the parameters in the model through gradient descent. Particularly, we calculate the final loss value by combining three loss functions presented in the above section:

$$\mathcal{L} = \mathcal{L}_{NMF} + \mathcal{L}_{GAE} + \mathcal{L}_{smooth} \tag{9}$$

4.5 Event Detection

The event embedding matrix $\mathbf{E}$ of the size $K \times D$ is considered containing information of K events. Each of the K events is represented by a number of topic words. The topic words can be obtained by ranking D words by their corresponding value in each row of $\mathbf{E}$. These top l topic words are considered most relevant for the topic and can be used for explaining and characterizing events. In the experiment, we use these top l words to represent the detected events, and compare them with ground truth labels, which are also some descriptive words representing the events.

5 Event Detection Evaluation

We conduct experimental evaluations to test the effectiveness of our method in the task of event detection. In this section, we will present the experiment setup and baseline methods, and discuss the results.

5.1 Datasets

In our experimental evaluation we use two real-world social media datasets. The first is called Reseed [11] dataset, which is currently publicly available[2]. The dataset contains about 437k Flickr posts, which are collected through Flickr API by filtering posts that have a social event tag. The posts are associated with textual titles, textual tags, and photos. Each post has a post date and a photo taken date, which are concentrated around 2006 to 2012. The second dataset is called Politics, which is collected by our own effort. First, we collect a list of Japanese politician Twitter accounts[3]. Next, we collect the follower accounts of these politicians. Then we collect from these follower accounts tweets that dated between Jun and September 2017. In total, this dataset contains about 2,464k Japanese tweets from 33,443 accounts.

We clean both datasets by removing less frequent words. For the Reseed dataset, we remove words with a frequency less than 100, resulting in a vocabulary of 2,438. For the Politics dataset, we remove words with a frequency less than 200, resulting in a vocabulary size of 6,955. A summary of these statistics about the dataset is shown in Table 1.

Table 1: Dataset Statistics

dataset	Reseed	Politics
period	2006.01-2010.07	2017.06-2017.09
time unit length	14 days	1 day
number of units	120	120
number of posts	437k	2,464k
vocabulary size	2,438	6,955

For evaluating event detection accuracy, we prepare corresponding ground truth data. The Reseed dataset comes with a ground truth file that contains a list of events that fall in the same period as the posts. The list is constructed from the web service Upcoming, which provides information on musical events. The events are associated with textual descriptions and timestamps. We tokenize the textual descriptions using a standard tokenizer and assign them to time units as the ground truth label. For the Politics dataset, we obtain the ground truth from the pre-trained large language model, GPT-4[4]. GPT-4 is trained on the entire Web data up to 2021. It cannot generate answers for information related to a specific date, but is relatively accurate on a monthly level. We query GPT-4 with this prompt, "What are the top 10 Japanese political events that happened in *month* 2017?", where $month = \{June|July|August|September\}$, and use its answer, textual descriptions of top 10 events, as the ground truth label.

[2] https://qualinet.github.io/databases/image/ reseed_social_event_detection_dataset/

[3] An example list is provided by the website Meyou with the url https://meyou.jp/group/category/politician/

[4] https://openai.com/research/gpt-4

5.2 Evaluation Metrics

We measure the accuracy of event detection by comparing the output of various embedding approaches with the ground truth. As we mentioned earlier, the events are predicted by ranking the words according to their values in a latent dimension. So for each time unit t, we predict top l words in all K latent dimensions as P_t. The ground truth label is a collection of words L_t. As such, the true positive count of prediction is $TP = \sum_{t \in T} |P_t \cap L_t|$. Then we measure precision, recall, and the F-value, where precision is $\frac{TP}{\sum_{t \in T} |P_t|}$, recall is $\frac{TP}{\sum_{t \in T} |L_t|}$, and F-value is $2 \cdot \frac{Precision \times Recall}{Precision + Recall}$. For experiments with the Politics dataset, the ground truth is of a month, and the prediction is of a day. In this case, when measuring precision, we use a day as the unit, and repeat the ground truth for each day of the month. When measuring recall, we use a month as the unit, and aggregate the daily predictions to a month.

5.3 Baseline Methods and Implementation

We compare our model with seven baseline methods, some are topic model methods, and some are graph embedding methods. These include LDA [2], OBTM [5], ONMF [8], EvNMF [12], EmNMF [12], GCN [16], and EvGCN [9], all of which we have briefly introduced in the Related Work section.

We implement all models except OBTM as neural networks using Python and TensorFlow. For OBTM, we use the code provided by the authors[5]. We set the dimension of the latent factor, i.e., the number of topics, to 10 for all models. For the γ in GTMA, we set it to 1000, which seems to provide optimal results. We will make the code for our model available soon.

5.4 Results and Discussion

The accuracy results for the Reseed and Politics datasets are shown in Table 2 and 3, respectively. The best result among compared methods is highlighted in bold font.

For the Reseed dataset, we see that the best method is GTMA. When l is 20, the accuracy is improved by 3.7% compared to the best baseline. For the Politics dataset, we see that some methods tend to have higher precision while others tend to have higher recall. To have higher precision, it is better to predict critical and similar topic words, while to have a higher recall, it is better to predict words of higher variety. From the results, we see that OBTM, ONMF, and GTMA have high precision and low recall, while LDA, EmNMF, GCN, and EvGCN have high recall and low precision, separated by the cause we described. However, even though GTMA achieves higher precision by predicting fewer words, its recall is nonetheless relatively high compared to similar methods. As a result, it achieves the best F-value. This shows that GTMA has the best balance between picking a large number of words and pickingcritical topic words.

[5] https://github.com/xiaohuiyan/OnlineBTM

Table 2: Event detection accuracy results on the Reseed dataset

	P@20	P@50	R@20	R@50	F@20	F@50
LDA	0.129	0.107	0.149	0.308	0.138	0.158
OBTM	0.169	0.146	0.195	0.421	0.181	0.217
ONMF	0.156	0.147	0.180	0.423	0.167	0.218
EvNMF	0.158	0.157	0.182	0.453	0.169	0.233
EmNMF	0.158	0.158	0.182	0.457	0.169	0.235
GCN	0.146	0.136	0.168	0.392	0.156	0.202
EvGCN	0.152	0.121	0.176	0.350	0.163	0.180
GTMA	**0.175**	**0.160**	**0.202**	**0.462**	**0.188**	**0.238**

Table 3: Event detection accuracy results on the Politics dataset

	P@20	P@50	R@20	R@50	F@20	F@50
LDA	0.110	0.081	0.469	0.589	0.178	0.143
OBTM	0.158	0.109	0.214	0.340	0.182	0.165
ONMF	0.128	0.102	0.226	0.334	0.163	0.157
EvNMF	0.117	0.094	0.357	0.563	0.177	0.161
EmNMF	0.113	0.091	**0.491**	**0.674**	0.183	0.161
GCN	0.127	0.103	0.454	0.606	0.199	0.176
EvGCN	0.138	0.102	0.451	0.617	0.211	0.175
GTMA	**0.233**	**0.177**	0.300	0.469	**0.262**	**0.257**

6 Conclusion

Event detection from social media is an important and challenging problem. In existing works, topic models and graph embedding methods have been used for learning event embedding from social media. In this paper, we propose a model to combine the advantages of topic models and graph embeddings for better event embedding learning. Our model consists of an NMF module, a GAE module, and a temporal smoothing module. Experimental evaluation with two real-world social media datasets shows that each module contributes positively to the model performance. We also compare our model with a number of existing approaches and show that our model has a superior performance. In the future, we plan to further develop our model to include a variable vocabulary.

References

1. L. AlSumait, D. Barbará, and C. Domeniconi. On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *2008 eighth IEEE international conference on data mining*, pages 3–12. IEEE, 2008.
2. D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.
3. A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, 26, 2013.

4. Y. Chen, H. Zhang, R. Liu, Z. Ye, and J. Lin. Experimental explorations on short text topic mining between lda and nmf based schemes. *Knowledge-Based Systems*, 163:1–13, 2019.
5. X. Cheng, X. Yan, Y. Lan, and J. Guo. Btm: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2928–2941, 2014.
6. T. N. Kipf and M. Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.
7. T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the Fifth International Conference on Learning Representations ICLR*, 2017.
8. J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(1), 2010.
9. A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, and C. Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5363–5370, 2020.
10. I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 569–577. ACM, 2008.
11. T. Reuter, S. Papadopoulos, V. Mezaris, and P. Cimiano. Reseed: social event detection dataset. In *Proceedings of the 5th ACM Multimedia Systems Conference*, pages 35–40, 2014.
12. A. Saha and V. Sindhwani. Learning evolving and emerging topics in social media: a dynamic nmf approach with temporal regularization. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 693–702, 2012.
13. Y. Wang, E. Agichtein, and M. Benzi. Tm-lda: efficient online modeling of latent topic transitions in social media. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 123–131, 2012.
14. Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28, 2014.
15. W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273. ACM, 2003.
16. S. Zhang, H. Tong, J. Xu, and R. Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.
17. Y. Zhang, X. S. Fang, and T. Hara. Evolving social media background representation with frequency weights and co-occurrence graphs. *ACM Transactions on Knowledge Discovery from Data*, 17(7):1–17, 2023.
18. Y. Zhang, M. Shirakawa, and T. Hara. Generalized durative event detection on social media. *Journal of Intelligent Information Systems*, pages 1–23, 2022.
19. Y. Zhang, M. Shirakawa, Y. Wang, Z. Li, and T. Hara. Twitter-aided decision making: a review of recent developments. *Applied Intelligence*, 52(12):13839–13854, 2022.
20. Y. Zhang, C. Szabo, and Q. Z. Sheng. Sense and focus: Towards effective location inference and event detection on twitter. In *Proceedings of the 16th International Conference on Web Information Systems Engineering Part I*, pages 463–477, 2015.