Resource: A Framework for Community Prediction

Soroush Ziaeinejad
University of Windsor, Canada
ziaeines@uwindsor.ca

Saeed Samet
University of Windsor, Canada
ssamet@uwindsor.ca

Hossein Fani
University of Windsor, Canada
hfani@uwindsor.ca

ABSTRACT

Online user communities exhibit distinct temporal dynamics in response to popular topics or breaking events. Despite abundant community detection libraries, there is yet to be one that provides access to the possible user communities in future time intervals. To bridge this gap, we contribute SEERa, an open-source end-to-end community *prediction* framework to identify future user communities in a text streaming social network. SEERa incorporates state-of-the-art temporal graph neural networks to model inter-user topical affinities at each time interval via streams of temporal graphs and learns temporal vector representations for users. This all takes place while users' topics of interest and hence their inter-user topical affinities are changing over time. SEERa predicts yet-to-be-seen user communities based on the final positions of users' vectors in the latent space. Notably, our framework serves as a one-stop-shop to future user communities for Social Information Retrieval and Social Recommendation systems. While there are strong research papers on the community prediction problem, SEERa is the first framework to be publicly released for this purpose.

1 INTRODUCTION

Social networks are the prominent medium for communication and social interaction wherein communities of like-minded users who are interested in similar topics emerge due to homophily [24]. Identifying user communities finds immediate application in scalable item recommendation and marketing campaigns [5, 31]. Traditionally, user community detection methods were based on explicit links, e.g., followership. However, explicit links conflate topical similarity among users when connections are missing or due to kinship [4, 33]. To this end, *latent* community detection methods have been proposed based on users' textual posts and their topics of interest. Whether to find explicit or latent communities, existing methods primarily focus on identifying communities at the current point in time and fall short of extracting communities that will emerge in the future, except for Fani et al. [13] and Chimera [3] where temporal latent space embedding and shared matrix factorization are proposed respectively to predict the possible user communities in future yet-to-be-observed time intervals. In fact, online user communities are temporal since users enjoy varying behavior in response to popular topics or breaking events [11, 32, 35].

Meanwhile, myriads of temporal graph neural networks like [9, 26, 28] have been proposed for a range of social network tasks such as link prediction that can be employed for community prediction owing to their capability to model and learn from graph-structured data. Also, a dizzying array of topic modeling libraries [6, 18, 27, 30] and an increasing research in Social Information Retrieval (Social IR) and Social Recommendation that integrates social network contexts to enhance rankings and recommendations [20, 21, 36, 37] have been introduced to date. However, there is yet to be one end-to-end (no human in the loop) case-specific framework that unifies topic

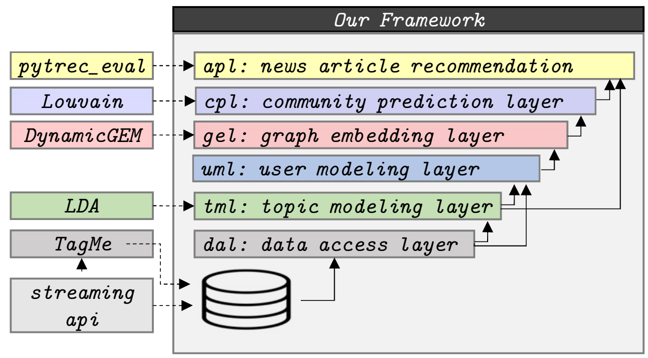

Figure 1: SEERa leverages layered architecture to add modularity, ease of extensibility, and stability against customization.

modeling approaches with temporal graph embedding methods for the task of topical community prediction in social networks.

To establish a unified software platform for the task of topical community prediction, we contribute SEERa, the first open-source python-based framework for identifying the possible user communities in future yet-to-be-observed time intervals. SEERa models inter-user topical affinities in each time interval in user graphs whose nodes and edges are users and their topical affinities, respectively. Given the stream of graphs, SEERa applies a range of state-of-the-art temporal latent space embedding and graph neural networks to capture the users' temporal topics of interest and their inter-user affinities from the past up until now. Temporal embedding methods allow users to change their positions in latent space as their topics of interest evolve over time: the more similar temporal and topical behavior the users exhibit, the closer their final latent positions would be, upon which SEERa can accurately estimate user communities in the future whose members will *almost surely* share similar topics. Out of the box, SEERa includes a news article recommendation module to exhibit how future community prediction is effective in applications in which an accurate estimation of users' interests in the future is desired. Key contributions of SEERa include:

(1) SEERa predicts future user communities and can be configured for *any* online social platform with streaming content.
(2) SEERa is designed with extensibility in mind. While already hosts a wide variety of methods in each layer of its pipeline, it facilitates the addition of new topic modeling methods, temporal graph embeddings, graph clustering methods, and application-level use cases;
(3) SEERa facilitates reproducibility and repeatability of the research work on future user community prediction on a shared underlying application. Also, it helps practitioners with cross-comparing different topic modeling algorithms with temporal graph embedding methods and pick the most suitable one for their application. As a part of its release, it includes a news article recommendation application. Indeed, community-level

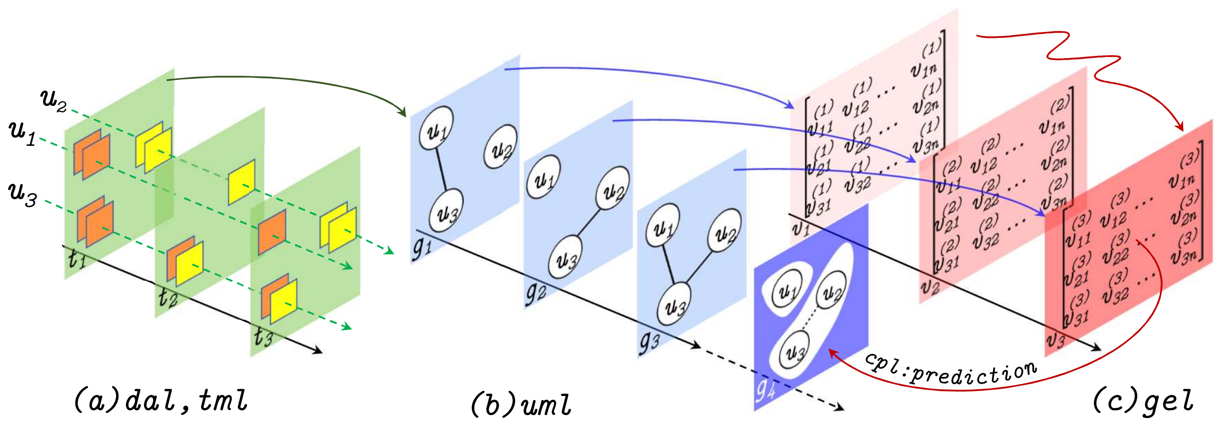

Figure 2: The overall flow of SEERa. g_4 is the predicted graph in the future whose communities are the future user communities.

```python
#./src/main.py
def RunPipeline():
    dataset = load_tweets(path, startDate, endDate, ...)
    processedDocs, documents = data_preparation(dataset, ...)
    topicOut = topic_modeling(processedDocs, numTopics, ...)
    userGraphs = user_modeling(documents, ...)
    embeddedMatrix = graph_embedding(userGraphs, method='DynAERNN')
    communities = graph_clustering(embeddedMatrix)
```

Figure 3: Entry point to SEERa's pipeline.

item recommendation in the future time intervals is a novel step forward to the recommendation systems.

The codebase, installation instructions and video tutorials along with a case study on news article recommendation[1] are available and can be obtained under `cc-by-nc-sa-4.0` license at: `github.com/fani-lab/seera`

2 FRAMEWORK OVERVIEW

Figure 2 shows the workflow of SEERa. In Figure 2(a), SEERa streams raw posts within a time period and groups them into coherent semantic contexts (documents) where a user describes her topics of interest. In Figure 2(b), SEERa identifies topics of interest from user's documents and connects users who share strong similarities in their topics of interest in a graph g_t. In Figure 2(c), given the stream of inter-user similarity graphs, SEERa leverages a host of state-of-the-art temporal graph neural networks and temporal latent space modeling to embed users (nodes) into vectors as they pass through time. Finally, SEERa applies a clustering method such as Louvain [8] to users' vectors at last time interval to unfold densely connected users as the future communities. Figure 3 shows the entry point to the pipeline.

As shown in Figure 1, SEERa benefits from layered software design which adds modularity, maintainability, ease of extensibility, robustness, and stability with respect to customization and ad hoc changes. Our framework consists of a total 5+1 pipelined layers for data access, topic modeling, user modeling, graph embedding, community prediction, and news article recommendation. We detail each layer in the following subsections.

[1]https://colab.research.google.com/github/fani-lab/SEERa/blob/main/quickstart.ipynb

```python
# ./src/dal/DataReader.py
def load_tweets(path, startDate, endDate, stopwords, tagme_threshold):
    tweets = pd.read_csv(path, sep=';', encoding='utf-8')
    tweets = tweets[tweets.Score > tagme_threshold]
    tweets = tweets[~tweets['Word'].isin(stopwords)]
    tweets = tweets.loc[(tweets['CreationDate'] >= startDate)
                        & (tweets['CreationDate'] <  endDate)]
    return tweets
# ./src/dal/DataPreparation.py
def data_preparation(dataset, time, prep, tagme):
    documents = dataset.groupby(['UserId'])
    if time: documents = dataset.groupby(['UserId', 'CreationDate'])
    if prep: processed_docs = documents['Tokens'].map(preprocess)
    if tagme: processed_docs = documents['Tokens'].map(TAGME)
    return processed_docs, documents
```

Figure 4: Data Access Layer (`dal`).

2.1 Data Access Layer (`dal`)

The primary purpose of the data access layer (`dal`) is to *i*) load raw texts posted by users within a time period T (`load_tweets()`), and *ii*) group them into coherent contexts, namely documents, where a user describes one or a few topics of her interest (`data_preparation()`). A document is either *i*) a single post of a user (`time=False`), or *ii*) can be enriched by all posts of a user in each time interval (`time=True`), known as pooling [25], which is based on a prior assumption that whether users follow different topics of interest in each of their posts or remain consistent to a single or a few topics of interest within a time interval. Figure 4 shows that document's tokens are either raw words (`tagme=False`) or semantic entities (`tagme=True`). The choice of semantic entities is encouraged if posts are informal, short, and noisy (e.g., tweets) as opposed to long formal documents (e.g., weblogs). SEERa employs TagMe [15], a fast and accurate semantic annotator for short texts that uses Wikipedia as its background knowledge to disambiguate and identify semantic entities while filtering stopwords and uninformative tokens.

Once documents are formed, they are fed to the topic modeling layer (`tml`) for users' topics of interest detection.

2.2 Topic Modeling Layer (`tml`)

This layer identifies topics of interest from user's documents using latent Dirichlet allocation (LDA) [7]. It can also be easily extended to any other topic modeling method such as TwitterLDA [2] for short streaming content. This layer returns a list of topics as probability distributions over all unique tokens, and documents as distributions over topics. To this end, SEERa benefits from `gensim` [27] (python)

Table 1: Dominant topics on Twitter for Nov. 1 – Dec. 15, 2010.

topic1		topic2		topic3	
papers	0.017	new_york	0.006	aung_san_suu_kyi	0.018
secret	0.017	live	0.006	myanmar	0.014
usaid	0.017	cigarette	0.005	activist	0.011

```python
# ./src/uml/UserSimilarities.py
def user_modeling(documents, lda_model, timeInterval ...):
    day = documents['CreationDate'].min()
    end_date = documents['CreationDate'].max()
    while day <= end_date:
        c = documents[(documents['CreationDate'] == day)]
        users_topic_interests = topic_modeling.doc2topics(lda_model, ... )
        graph = networkx.from_numpy_matrix(cosine_similarity, ...)
        day = day + timeInterval
```

Figure 5: User Modeling Layer (`uml`).

as well as `mallet`[23] (java) libraries. The quality of output topics under different numbers of topics (`ntopics`) and other settings can be quantitatively tuned by coherence [10]. Table 1 shows dominant topics from Nov. 1, until Dec. 15, 2010, when *'Auug San Suu Kyi'*, a political activist, was among the users' topics of interest.

2.3 User Modeling Layer (`uml`)

Given the user's documents as distributions over identified topics from (`tml`), from those documents at time interval t, `SEERa` calculates pairwise inter-user topical similarities at t and connects users who share strong similarities in their topics of interest in the graph g_t, as shown in Figure 5. The final output of this layer is the stream of graphs for the entire time period $[g_t]_{t=1}^{\mathrm{T}}$. The time interval can be adjusted for, e.g., daily, weekly, biweekly, or monthly. We underscore that graphs are generated independently at this layer, and temporal relations between graphs are overlooked. `SEERa` employs dynamic graph embedding methods to address this issue in the next layer.

2.4 Graph Embedding Layer (`gel`)

Given $[g_t]_{t=1}^{\mathrm{T}}$, the graph embedding layer (`gel`) applies temporal embedding methods to embed users (nodes) into a low d-dimensional vectors $[v_t]_{t=1}^{\mathrm{T}}$ as they pass through time while capturing temporal and topical affinities from their initial position v_1 to a final position v_{T}, as shown Figure 6. Users' vectors at the final time interval, v_{T}, depend on preceding $[v_1...v_{t<\mathrm{T}}]$ via observation of $[g_1...g_{t<\mathrm{T}}]$ which is in contrast with static models that obtain v_{T} based solely on g_{T}. Our framework employs state-of-the-art temporal graph neural network methods including Dynamic Auto-Encoder, Dynamic RNN that uses LSTM, and Dynamic AERNN which uses AE followed by LSTM network [16], and temporal latent space modeling based on non-negative matrix factorization [17].

2.5 Community Prediction Layer (`cpl`)

`SEERa` uses users' vectors at the final time interval, v_{T}, as the proxy to predict the topical similarities of users in future time intervals, i.e., graph $\hat{g}_{t'>\mathrm{T}}$, based on pairwise cosine similarity $(v_{\mathrm{T},i} \cdot v_{\mathrm{T},j})$ of the users i and j. We are, additionally, developing more advanced prediction methods such as multivariate regression on $v_1...v_{\mathrm{T}}$ to predict $v_{t'>\mathrm{T}}$ as a better proxy for $\hat{g}_{t'>\mathrm{T}}$. Finally, a clustering method on $\hat{g}_{t'>\mathrm{T}}$ unfolds densely connected subgraphs as the future user communities. As shown in Figure 7, while we use Louvain [8],

```python
# ./src/gel/GraphEmbedding.py
def graph_embedding(method):
    # methods: ['Node2Vec', 'AE', 'DynAE', 'DynRNN', 'DynAERNN'] are available.
    if method == 'Node2Vec': N2V.main('/graphs', params.gel['EmbeddingDim'])
    else:
        embedding = GEMmethod(method=method, ...)
        emb, _ = embedding.learn_embeddings(graphs)
    return emb
def GEMmethod(dim_emb, lookback, method='DynAERNN'):
        if method == 'AE':       embedding = AE(d=dim_emb, ...)
      elif method == 'DynAE':    embedding = DynAE(d=dim_emb, ...)
      elif method == 'DynRNN':   embedding = DynRNN(d=dim_emb, ...)
      elif method == 'DynAERNN': embedding = DynAERNN(d=dim_emb, ...)
    return embedding
```

Figure 6: Graph Embedding Layer (`gel`).

```python
# ./src/cpl/GraphClustering.py
def graph_clustering(embeddedMatrix):
    adjMTX = cosine_similarity(embeddedMatrix)
    louvain = sknetwork.clustering.Louvain(resolution=1, n_aggregations=200)
    lbls_louvain = louvain.fit_transform(adjMTX)
    return lbls_louvain
```

Figure 7: Community Prediction Layer (`cpl`).

the canonical method in community detection, this layer can be seamlessly extended to any clustering methods [22, 29]. We are currently extending `SEERa` to clustering methods to be applied directly on users' d-dimensional vectors at the final time interval v_{T} to obtain the future user communities, dispensing the intermediate graph representation $\hat{g}_{t'>\mathrm{T}}$.

To quantitatively evaluate the quality of the predicted future user communities, our framework includes two methods based on the availability of golden standard: (1) intrinsic evaluation using well-known metrics such as rand index and mutual information, and (2) extrinsic evaluation, that is, to what extent an extra knowledge about users' membership to future communities will provide synergy to the efficacy and efficiency of an underlying application.

2.6 Application Layer (`apl`): News Article Recommendation

Future community prediction helps applications with prior insight into the future and to excel at effectiveness and efficiency. Out of the box, we implemented an important use case in Social Recommendation as suggested in [1, 14]: news article recommendations. Future user communities can signal a recommender system to target users based on topics of the communities to which they belong. Therefore, community prediction not only increases the efficiency by targeting communities, as opposed to individual users, but also improves the efficacy of the recommendations via prior knowledge about topics of interest in the future. For instance, in Figure 2(b), g_4 is the predicted graph in the future and u_2 and u_3 are members of the same community. They will receive the same set of news articles whose topics are different from the news articles that u_1 receives.

To this end, we define community-level topics of interest as the sum of all its members' predicted topics of interest in the future. Top-k news articles are recommended to each community (i.e., all of its members) according to cosine similarity scores between the news articles' topics and community-level topics. The recommendation table is then evaluated against users' mentions of news articles in their posts. We use `pytrec_eval` [34] to report information retrieval metrics such as `MRR` and `nDCG`.

```python
# ./src/params.py
RunID = 1

general = {
    'Comment': '', # Any comments.
    'RunId': RunID,
}
dal = {
    'path': '../data/tweets_all.csv',
    'userModeling': True, # document: all tweets of a user
    'timeModeling': True, # document: all tweets of a specific day
    'start': '2010-11-01', # First date of system activity
    'end': '2010-12-14', # Last day of system activity
    'timeInterval': 1, # Time interval (days) for grouping documents
    'preProcessing': False, # Applying pre-processing methods on corpus
    'TagME': True # Apply Tagme on the dataset.
}
tml = {
    'num_topics': 25, # Number of topics
    'library': 'gensim', # Library for topic modeling: ['gensim', 'mallet']
    'mallet_home': 'C:/Users/[User]/mallet-2.0.8', # Path to mallet dir
    'filterExtremes': True, # Filtering very rare and very common terms.
    'JO': False, # 'Just One' topic is being selected.
    'Bin': True, # 'Binary' topic score instead of weighted score.
    'Threshold': 0.2 # Quantization Threshold.
}
uml = {
    'UserSimilarityThreshold': 0.5, # Filtering low user similarity scores
}
gel = {
    'EmbeddingDim': 40, # Graph embedding dimension
    'method': 'Node2Vec' # Graph embedding method.
    #Available options: ['Node2Vec', 'AE', 'DynAE', 'DynRNN', 'DynAERNN']
}
evl = {
    'EvaluationType': 'Extrinsic',  # ['Intrinsic', 'Extrinsic']
    # If intrinsic evaluation:
    'EvaluationMetrics': ['adjusted_rand', 'completeness', 'homogeneity',
    'rand', 'v_measure', 'normalized_mutual_info', 'adjusted_mutual_info',
    'mutual_info', 'fowlkes_mallows'],
    'GoldenStandardPath': '/GS',# Path to the golden standard if exists
    'Threshold': 0.2, # Filtering low news article recommendation scores
    'TopK': 20 # Number of top news article recommendation candidates
}
```

Figure 8: The hyperparameters of SEERa at each layer.

```
>> cd seera/src/
>> python main.py
Data Reading:
Start and end dates: from (1-11-2010) to (15-11-2010)
#Tweets: 9414
#Users: 895
Time Interval: 1 day
---------------------------------------------
Topic Modeling:
#Topics: 25
The lda_model is saved.
---------------------------------------------
User Modeling:
Shape of users_topic matrix: (15, 895, 25)
15 graphs are created for 15 days with 895 users.
---------------------------------------------
Graph Embedding:
Embedding Dimension: 40
Method: Node2Vec
Embedded Matrix: (895, 40) is saved.
---------------------------------------------
Graph Clustering:
method: Louvain
#Nodes: 895 / #Edges: 20960
Clustering output: 24 communities
C0: 80 users / Favorite topic: T14 (88.7%)
C1: 67 users / Favorite topic: T09 (94.0%)
C2: 66 users / Favorite topic: T10 (89.3%)
C3: 45 users / Favorite topic: T11 (86.6%)
C4: 41 users / Favorite topic: T21 (82.9%)
.
.
.
C24: 5 users / Favorite topic: T19 (100%)
```

Figure 9: Quick start and sample run of SEERa.

Table 2: SEERa's benchmark on a Twitter dataset.

Method	News Recommendation		
	MRR	nDCG@5	nDCG@10
Community Prediction			
Fani et al. [13]	**0.225**	**0.108**	**0.105**
Appel et al. [3]	0.176	0.056	0.055
Temporal Community Detection			
Hu et al. [19]	0.173	0.056	0.049
Fani et al. [12]	0.065	0.040	0.040

Table 2 shows SEERa's main benchmark results on a Twitter dataset of 2M tweets authored by 135K users within 59+1 days and the performance of predicted future communities on the 60th day for 50 topics of interest. Complete results have been reported publicly in the codebase. We are testing scalability of SEERa on a large-scale dataset including 300M tweets authored by 3.5M users within 6 months and the result will be released once our experiments have been accomplished.

3 QUICK START

SEERa can be obtained by:

```
git clone https://github.com/fani-lab/seera.git
```

SEERa has hyperparameters at each layer that should be set at ./src/params.py. Figure 8 shows the summary of parameters. For instance, number of topics or embedding dimension can be adjusted in tml and gel sections, respectively. The entry point to the framework is ./src/main.py that executes the pipeline until the final delivery of future user communities followed by their evaluation on news article recommendation. Figure 9 display an example output of our framework. From the figure, we ran SEERa on a sample dataset from Nov. 1 to Nov. 15, 2010 (today) to predict future communities on Nov. 15+1 (tomorrow). SEERa found 25 communities from which cluster C0 have 80 users, most of them (88.7%) interested in topic T14, *'2010s_Haiti_cholera_outbreak'*. Should a news recommender system recommend news articles to users tomorrow, it would select news articles about topic T14 for community C0, expecting a high click-through rate. Due to the SEERa's documentation and ease of use, researchers and practitioners familiar with python programming can run, modify, and use SEERa in different applications.

4 CONCLUDING REMARKS

In this paper, we propose SEERa, an open-source end-to-end framework, to identify future user communities in a text streaming social network. SEERa is to address the emerging need in Social IR and Social Recommendation systems for future communities. SEERa (1) is designed with extensibility in mind. While hosts a wide variety of methods in each layer of its pipeline, it easily accommodates new methods as well as application-level use cases; (2) can be easily configured for any online social platform with streaming content; (3) improves the efficiency and efficacy of the underlying time-sensitive applications using the predicted future user communities; and (4) benefits the information retrieval and recommender systems community with reproducibility and repeatability of the research work on community prediction.

REFERENCES

[1] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011. Analyzing User Modeling on Twitter for Personalized News Recommendations. In *User Modeling, Adaption and Personalization - 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings (Lecture Notes in Computer Science, Vol. 6787)*, Joseph A. Konstan, Ricardo Conejo, José-Luis Marzo, and Nuria Oliver (Eds.). Springer, 1–12. https://doi.org/10.1007/978-3-642-22362-4_1

[2] Despoina Antonakaki, Paraskevi Fragopoulou, and Sotiris Ioannidis. 2021. A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks. *Expert Systems with Applications* 164 (2021), 114006.

[3] Ana Paula Appel, Renato LF Cunha, Charu C Aggarwal, and Marcela Megumi Terakado. 2018. Temporally evolving community detection and prediction in content-centric networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 3–18.

[4] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2016. Efficient Methods for Influence-Based Network-Oblivious Community Detection. *ACM Trans. Intell. Syst. Technol.* 8, 2, Article 32 (Dec. 2016), 31 pages. https://doi.org/10.1145/2979682

[5] Punam Bedi and Chhavi Sharma. 2016. Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 6, 3 (2016), 115–135.

[6] Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. 2021. Cross-lingual Contextualized Topic Models with Zero-shot Learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, Online, 1676–1683. https://doi.org/10.18653/v1/2021.eacl-main.143

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3 (2003), 993–1022.

[8] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[9] Yukuo Cen, Zhenyu Hou, Yan Wang, Qibin Chen, Yizhen Luo, Xingcheng Yao, Aohan Zeng, Shiguang Guo, Peng Zhang, Guohao Dai, Yu Wang, Chang Zhou, Hongxia Yang, and Jie Tang. 2021. CogDL: Toolkit for Deep Learning on Graphs. *arXiv preprint arXiv:2103.00959* (2021).

[10] Uttam Chauhan and Apurva Shah. 2021. Topic modeling using latent Dirichlet allocation: A survey. *ACM Computing Surveys (CSUR)* 54, 7 (2021), 1–35.

[11] Narimene Dakiche, Fatima Benbouzid-Si Tayeb, Yahya Slimani, and Karima Benatchba. 2019. Tracking community evolution in social networks: A survey. *Information Processing & Management* 56, 3 (2019), 1084–1102.

[12] Hossein Fani, Ebrahim Bagheri, and Weichang Du. 2017. Temporally Like-Minded User Community Identification through Neural Embeddings *(CIKM '17)*. Association for Computing Machinery, New York, NY, USA, 577–586. https://doi.org/10.1145/3132847.3132955

[13] Hossein Fani, Ebrahim Bagheri, and Weichang Du. 2020. Temporal Latent Space Modeling for Community Prediction. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12035)*. Springer, 745–759. https://doi.org/10.1007/978-3-030-45439-5_49

[14] Hossein Fani, Eric Jiang, Ebrahim Bagheri, Feras N. Al-Obeidat, Weichang Du, and Mehdi Kargar. 2020. User community detection via embedding of social network structure and temporal content. *Inf. Process. Manag.* 57, 2 (2020), 102056. https://doi.org/10.1016/j.ipm.2019.102056

[15] Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-Fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (Toronto, ON, Canada) *(CIKM '10)*. Association for Computing Machinery, New York, NY, USA, 1625–1628. https://doi.org/10.1145/1871437.1871689

[16] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. 2020. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowl. Based Syst.* 187 (2020). https://doi.org/10.1016/j.knosys.2019.06.024

[17] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (2018), 78–94.

[18] Maarten Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794* (2022).

[19] Zhiting Hu, Junjie Yao, Bin Cui, and Eric Xing. 2015. Community level diffusion extraction. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 1555–1569.

[20] Zhenyan Ji, Huaiyu Pi, Wei Wei, Bo Xiong, Marcin Wozniak, and Robertas Damasevicius. 2019. Recommendation Based on Review Texts and Social Communities: A Hybrid Model. *IEEE Access* 7 (2019), 40416–40427. https://doi.org/10.1109/ACCESS.2019.2897586

[21] Danielle Hyunsook Lee and Peter Brusilovsky. 2017. Improving personalized recommendations using community membership information. *Inf. Process. Manag.* 53, 5 (2017), 1201–1214. https://doi.org/10.1016/j.ipm.2017.05.005

[22] Zhiping Lin and Zhao Kang. 2021. Graph Filter-based Multi-view Attributed Graph Clustering. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Zhi-Hua Zhou (Ed.). ijcai.org, 2723–2729. https://doi.org/10.24963/ijcai.2021/375

[23] Andrew Kachites McCallum. 2002. MALLET: A Machine Learning for Language Toolkit. (2002). http://mallet.cs.umass.edu.

[24] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 27, 1 (2001), 415–444. https://doi.org/10.1146/annurev.soc.27.1.415 arXiv:https://doi.org/10.1146/annurev.soc.27.1.415

[25] Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. 2013. Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Dublin, Ireland) *(SIGIR '13)*. Association for Computing Machinery, New York, NY, USA, 889–892. https://doi.org/10.1145/2484028.2484166

[26] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. graph2vec: Learning Distributed Representations of Graphs. https://doi.org/10.48550/ARXIV.1707.05005

[27] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[28] Benedek Rozemberczki, Paul Scherer, Yixuan He, George Panagopoulos, Alexander Riedel, Maria Astefanoaei, Oliver Kiss, Ferenc Beres, , Guzman Lopez, Nicolas Collignon, and Rik Sarkar. 2021. PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. 4564–4573.

[29] Shanshan Ruan, Rashid Mehmood, Ali Daud, Hussain Dawood, and Jalal S. Alowibdi. 2017. An Adaptive Method for Clustering by Fast Search-and-Find of Density Peaks: Adaptive-DP. In *Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 119–127. https://doi.org/10.1145/3041021.3054148

[30] sashafrey, Murat Apishev, Oleksandr Frei, Nikita Shapovalov, Peter Romov, Konstantin Arkhipenko, Dmytry Fedoyaka, Kirill Bobyrev, Michael Solotky, Vadim Markovtsev, Dmitriy Selivanov, Andrei Kashin, Alexander, vadimantiy, Valeriy, Vladimir Polushin, Nikolai Kruglikov, Lev Abashkin, Jeroen Ooms, Dmitry Yutkin, Johnny, and Anton Belyy. 2017. *bigartm/bigartm: Beta release 0.9.0*. https://doi.org/10.5281/zenodo.1041951

[31] Xing Su, Shan Xue, Fanzhen Liu, Jia Wu, Jian Yang, Chuan Zhou, Wenbin Hu, Cecile Paris, Surya Nepal, Di Jin, Quan Z. Sheng, and Philip S. Yu. 2022. A Comprehensive Survey on Community Detection With Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems* (2022), 1–21. https://doi.org/10.1109/TNNLS.2021.3137396

[32] Etienne Gael Tajeuna, Mohamed Bouguessa, and Shengrui Wang. 2018. Modeling and predicting community structure changes in time-evolving social networks. *IEEE Transactions on Knowledge and Data Engineering* 31, 6 (2018), 1166–1180.

[33] Cong Tran, Won-Yong Shin, and Andreas Spitz. 2021. Community detection in partially observable social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16, 2 (2021), 1–24.

[34] Christophe Van Gysel and Maarten de Rijke. 2018. Pytrec_eval: An Extremely Fast Python Interface to trec_eval. In *SIGIR*. ACM.

[35] Yu Xie, Bin Yu, Shengze Lv, Chen Zhang, Guodong Wang, and Maoguo Gong. 2021. A survey on heterogeneous network representation learning. *Pattern Recognition* 116 (2021), 107936.

[36] Jianling Zhong, Weiwei Guo, Huiji Gao, and Bo Long. 2020. Personalized Query Suggestions. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1645–1648. https://doi.org/10.1145/3397271.3401331

[37] Yujia Zhou, Zhicheng Dou, Bingzheng Wei, Ruobing Xie, and Ji-Rong Wen. 2021. Group based Personalized Search by Integrating Search Behaviour and Friend Network. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 92–101. https://doi.org/10.1145/3404835.3462918