Mining Twitter for Social Event and Misinformation Detection

Yingtong Dou
Department of Computer Science
University of Illinois at Chicago
Homepage: http://ytongdou.com
Email: ydou5@uic.edu
Twitter: @dozee_sim

UNIVERSITY OF ILLINOIS CHICAGO

Outlines

- **Background:** Twitter and Graph Neural Network
- **Paper I:** Detecting Misinformation on Twitter
- **Paper II:** Discovering Social Event on Twitter
- **Q&A**

Twitter Fact Sheet

- Twitter has **396M** users in 2021.
- **51M** Americans use Twitter daily.
- In the US, **92%** of tweets come from the top **10%** of users.
- **48%** Twitter users use Twitter to get news.
- Every second, there are **6k** new tweets are tweeted on Twitter.

Stats from https://backlinko.com/twitter-users

Twitter vs FB, Reddit, TikTok

- Twitter has an explicit social network structure.
- Twitter is an open platform to discuss public events.
- Structured Twitter data can be accessible via Twitter Developer API.

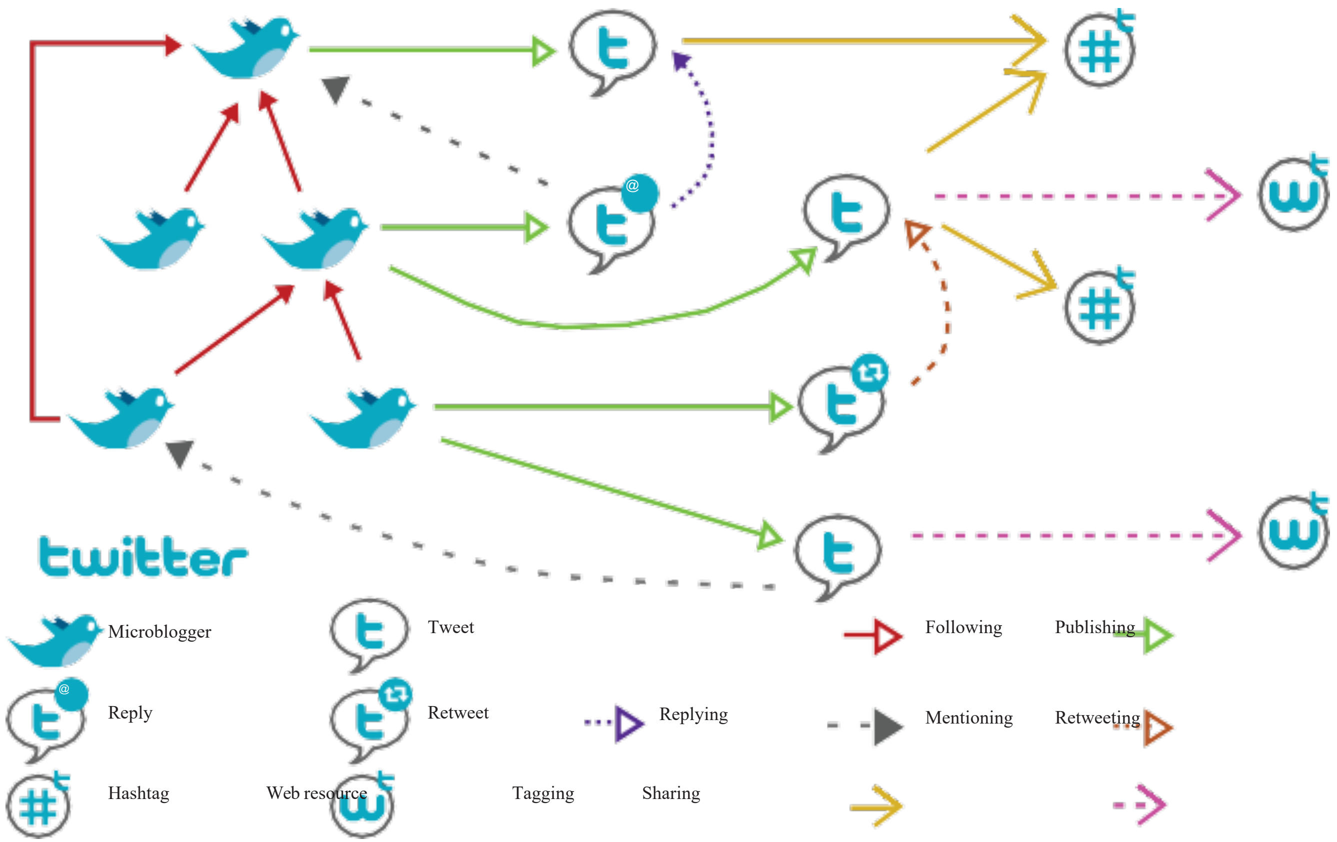


Image from Jabeur, et al. "Uprising microblogs: A Bayesian network retrieval model for tweet search." *ACM symposium on applied computing*. 2012.

Leveraging Twitter Data with Twitter API

- **Conduct academic research**
  - From computer science to social science.
- **Solve problems with applied research for NGOs**
  - Conduct scientific studies that solve problems to impact the mission of NGOs.
- **Enrich investigative journalism and independent research**
  - Use Twitter data to explore global to local topics and events that can inform projects and publications.
- **Conduct market research for business**
  - Understand your audience and what they value by uncovering trends and surfacing important conversations on Twitter.

https://developer.twitter.com/en/use-cases/do-research

CS Research using Twitter Data

- Social Event Detection
- Fraudster/Spammer/Bot/Sybil Detection
- Social Recommendation
- Misinformation Detection
- Graph Mining (link prediction, node classification/clustering)
- Sentiment/Emotion/Opinion/Stance/Topic Mining
- Multi-modal Data Mining
- Conversational Agent
- And More …

Botnet on Twitter

Human-like bot accounts on Twitter.

The profile pictures of accounts on the left, made opaque and superimposed.

Image from Graphika. Fake Cluster Boosts Huawei Accounts with GAN Faces Attack Belgium over 5G Restrictions

Botnet on Twitter

- A group of Twitter accounts retweet each other's tweets with similar content
- The densely connected accounts can be easily discovered from the graph perspective.

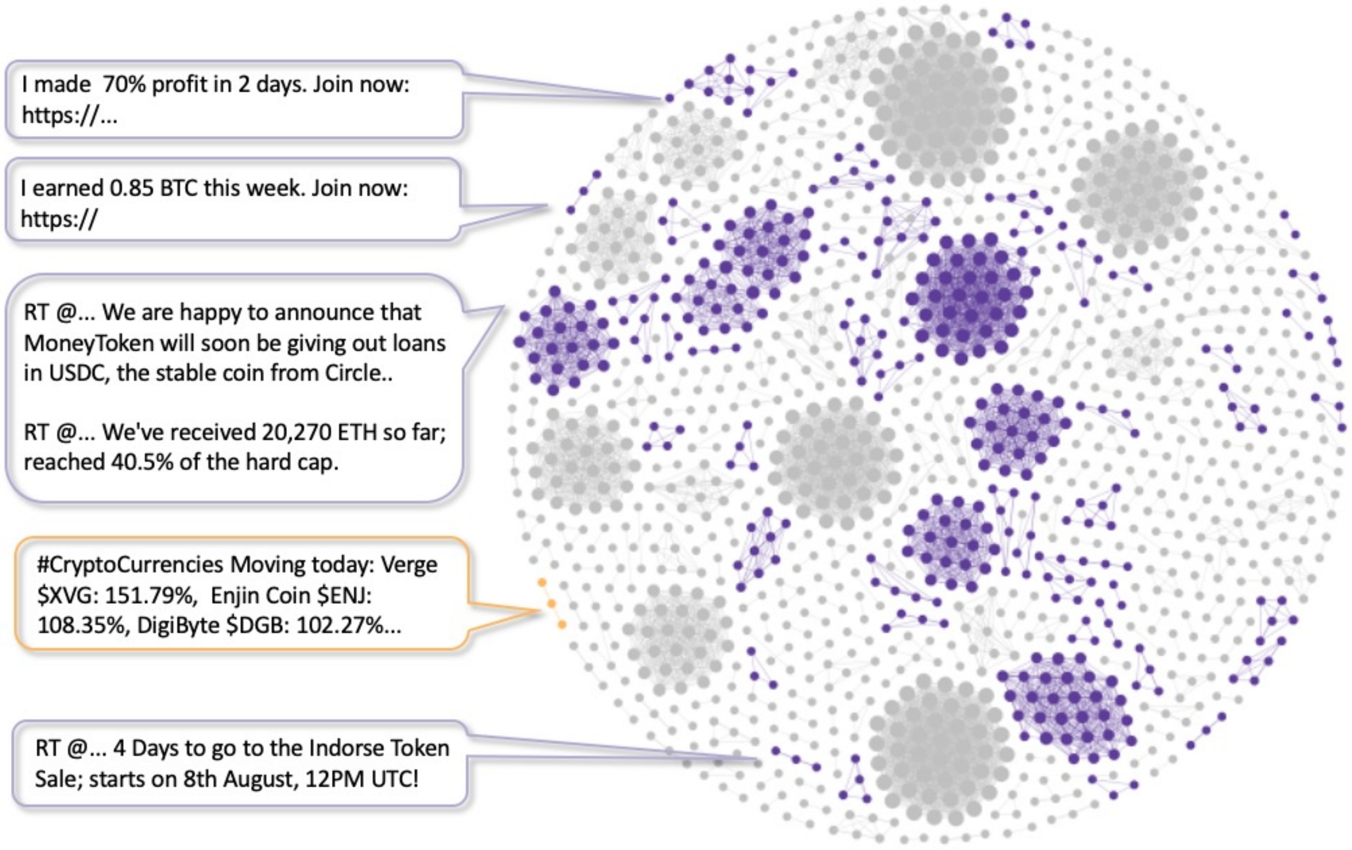
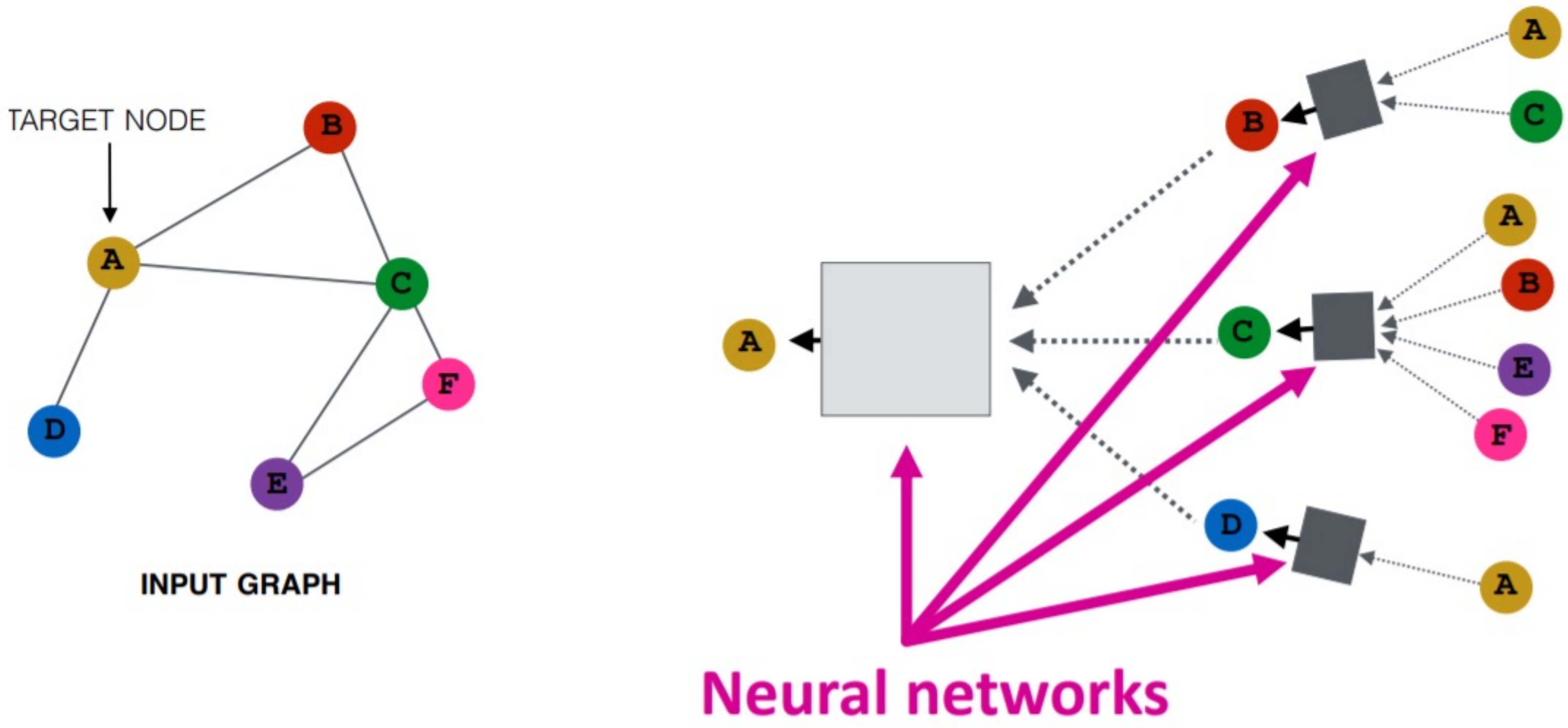


Image from Pacheco, Diogo, et al. "Uncovering Coordinated Networks on Social Media: Methods and Case Studies." *ICWSM*. 2021.

Graph Mining and Graph Neural Network



Key idea: the connected nodes are similar (homophily assumption).
Objective: learn an optimal neural network-based neighborhood encoder.

Image from Ying, Rex, et al. "Graph convolutional neural networks for web-scale recommender systems." KDD 2018.

Graph Neural Network for Graph Classification

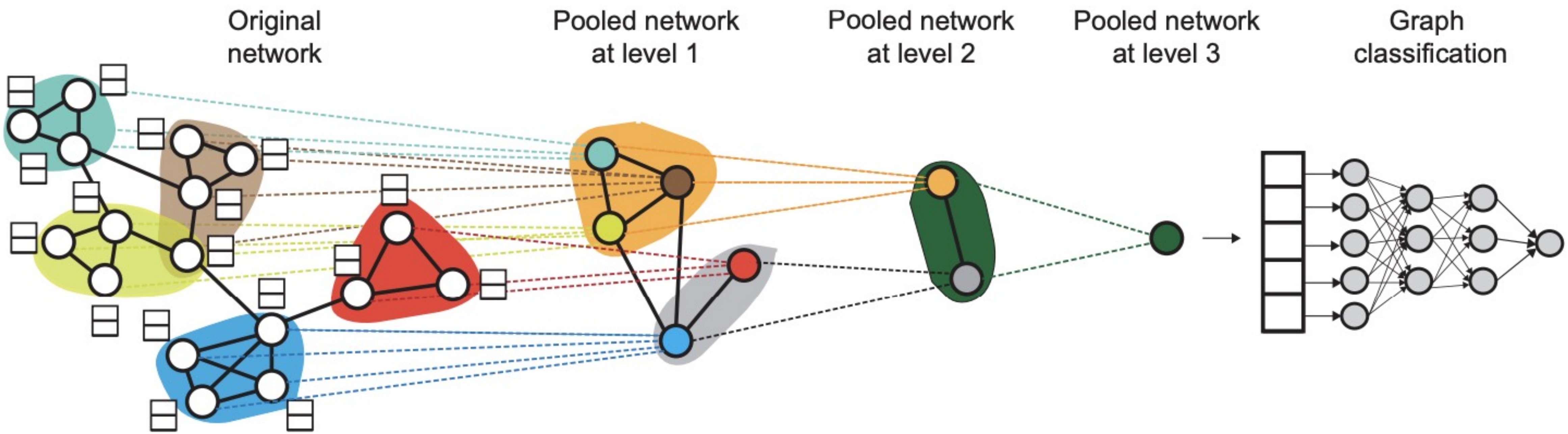


Image from Ying, Zhitao, et al. "Hierarchical graph representation learning with differentiable pooling." NeurIPS (2018).

In This Talk

- **Misinformation Detection on Twitter**
  - Introduce how to crawl Twitter data.
  - Modeling news propagation on Twitter as a tree-structured graph.
  - Supervised graph classification problem.
- **Social event detection on twitter**
  - Encoding different relations and entities on Twitter using GNN.
  - Unsupervised node clustering problem.

SIGIR'21: Misinformation Detection on Twitter

User Preference-aware Fake News Detection

Yingtong Dou, Congying Xia, Philip Yu (University of Illinois at Chicago)

Kai Shu (Illinois Institute of Technology)

Lichao Sun (Lehigh University)

Paper: https://arxiv.org/pdf/2104.12259.pdf
Code: https://github.com/safe-graph/GNN-FakeNews
Benchmark: https://paperswithcode.com/dataset/upfd
PyG Example: https://tinyurl.com/a6s92t37
DGL Example: https://tinyurl.com/yjwvd93b

UNIVERSITY OF ILLINOIS CHICAGO
ILLINOIS INSTITUTE OF TECHNOLOGY
LEHIGH UNIVERSITY

Problem Description

- **Fake news detection**
  - Fake news and real news are circulating on the social network.
  - A fake news has suspicious signals from its content, source, and social context.
  - Given the annotated news and their metadata, we want to train a neural network to classify unlabeled news.
- **Our motivation**
  - A social network user has his/her preference in consuming news. For a given piece of news, we assume its engaged users prefer similar types/content of news.
  - We propose to model user news consumption preference using user historical Tweets.

Data Collection

- **Existing Data:**
  - The FakeNewsNet dataset from Prof. Kai Shu.
  - News content, timestamp, news retweets id, retweets timestamp.

- **We need to collect:**
  - Retweet user metadata.
  - Retweet user historical tweets.

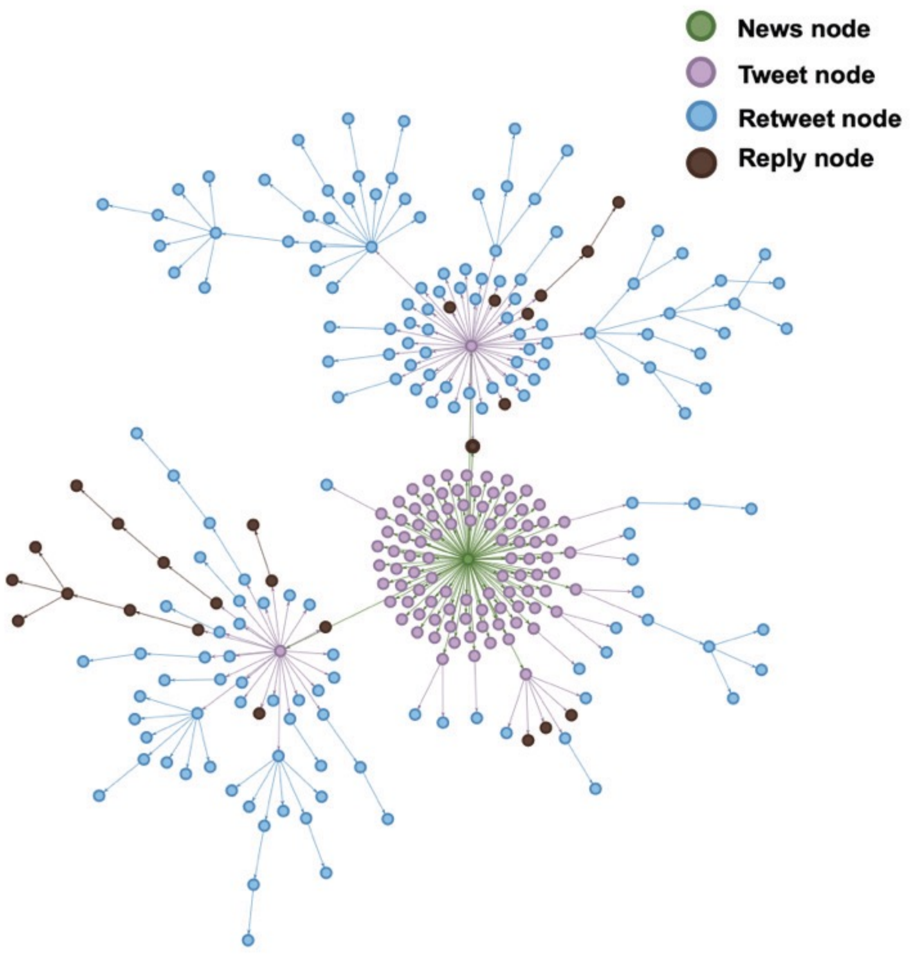
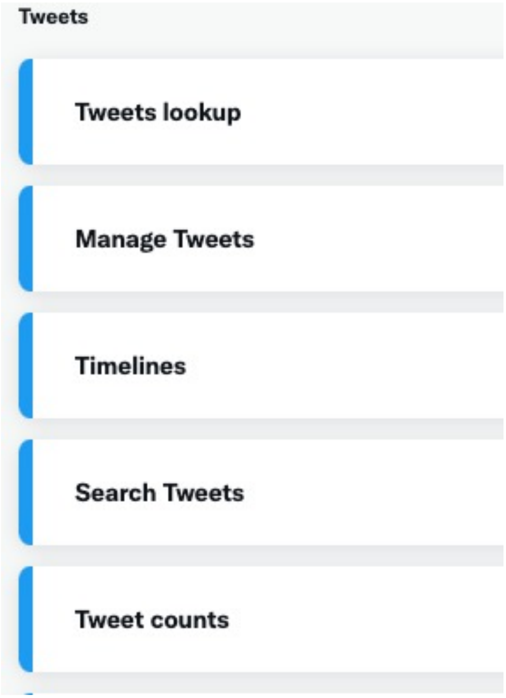


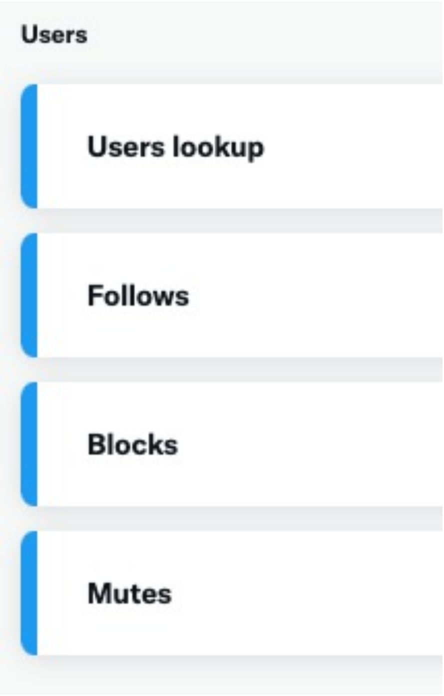
Image from Shu, Kai, et al. "Hierarchical propagation networks for fake news detection: Investigation and exploitation." AAAI. 2020.

Twitter Developer API

- University student can apply a free Twitter developer account using the school email.

- Twitter API can control a Twitter account using APIs and can access most of Twitter data in a limited request rate.





https://developer.twitter.com/en/docs

Python Crawler Code

- We use a Python Twitter crawler called tweepy to crawl the Twitter data.
- Tweepy will wait when API meets the rate limit.
- `api.user_timeline` will return a Twitter status object in json format.

```python
import tweepy
import json

# Twitter Developer API tokens provided by Twitter
auth = tweepy.OAuthHandler('xxx', 'xxx')
auth.set_access_token('xxx', 'xxx')

api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True)

m, n = 0, 0
for i, user in enumerate(id_mappings):  # user id to twitter id mappings
    try:
        # get recent 200 tweets of the user
        statuses = api.user_timeline(user_id=user, count=200)
        json_object = [json.dumps(s._json) + '\n' for s in statuses]
        # write the recent 200 tweets into a json file
        with open('content/' + str(user) + ".json", "w") as outfile:
            outfile.writelines(json_object)
        outfile.close()
    except tweepy.TweepError as err: # handled deleted/suspended accounts
        if str(err) == 'Not authorized.':
            m+=1
            print(f'Not authorized: {m}')
        else:
            n+=1
            print(f'Page does not exist: {n}')
    print(f'user number: {i}')

print(f'Not authorized: {m}, Page does not exist: {n}.')
```

Encoding User Preference

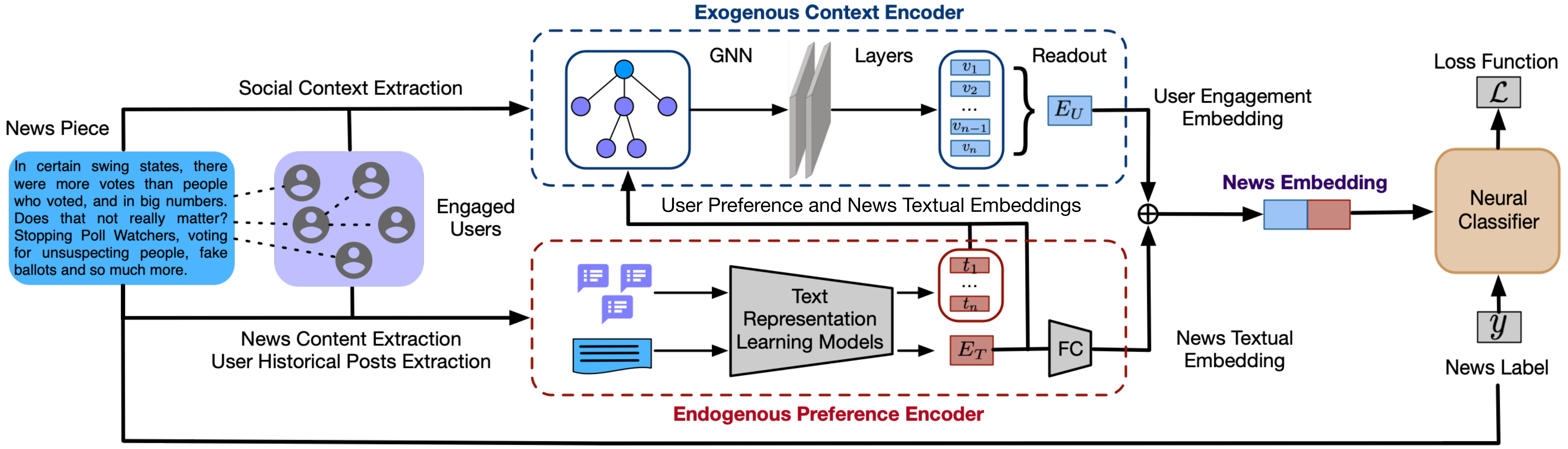
- **Preprocessing text data**
  - Remove special characters and emojis
  - Remove the "#"and "@"
  - Combining all tweets as a document

- **Three text encoders**
  - TFIDF embedding learned from current corpus.
  - Pretrained Word2vec vectors (using spaCy).
  - Pretrained BERT.

Baseline: Profile-based Features

- 1) Verified?, 2) Enable geo-spatial positioning,
- 3) Followers count, 4) Friends count,
- 5) Status count, 6) Favorite count, 7) Number of lists,
- 8) Created time (No. of months since Twitter established),
- 9) Number of words in the description,
- 10) Number of words in the screen name

Proposed UPFD Model



Experiment Setup

Data	#Graphs	#Fake News	#Total Nodes	#Total Edges	#Avg. Nodes per Graph
Politifact	314	157	41,054	40,740	131
Gossipcop	5464	2732	314,262	308,798	58

- **Baselines**
  - SAFE
  - CSI
  - GNN-CL
  - GCNFN
- **Metrics**
  - Accuracy
  - F1-score
- **Implementation**
  - Python
  - PyTorch
  - PyTorch_Geometric

Experiment Result

Table 2: The fake news detection performance of baselines and our model. Stars denote statistically significant under the t-test ($* \ p \leq 0.05, ** \ p \leq 0.01, *** \ p \leq 0.001$).

	Model	POL		GOS	
		ACC	F1	ACC	F1
News Only	SAFE [36]	73.30	72.87	77.37	77.19
	CSI [23]	76.02	75.99	75.20	75.01
	BERT+MLP	71.04	71.03	85.76	85.75
	word2vec+MLP	76.47	76.36	84.61	84.59
News + User	GNN-CL [8]	62.90	62.25	95.11	95.09
	GCNFN [17]	83.16	83.56	96.38	96.36
	UPFD (ours)	**84.62***	**84.65***	**97.23****	**97.22*****

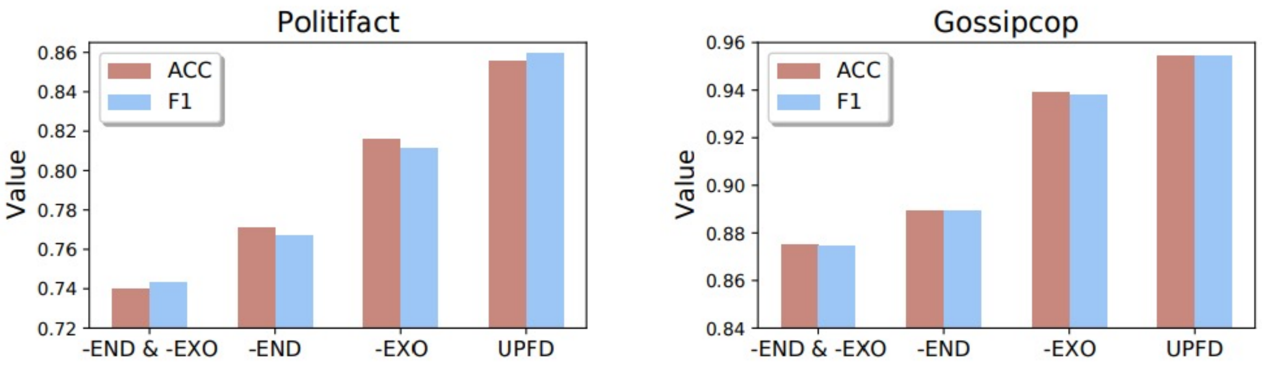


Figure 2: The fake news detection performance of different variants of UPFD framework. -END/-EXO represents the UPFD variant without endogenous/exogenous information.

Proposed UPFD model has the best performance.

Encoding user preference could improve fake news detection performance.

Future Directions

- Encode fine-grained user preference signals.
- Explain fake news detection results based on user preference.
- Consider the temporal information.
- Fake news early detection.
- News popularity/engagement prediction.

WWW'21: Social Event Detection on Twitter

Knowledge-Preserving Incremental Social Event Detection via Heterogeneous GNNs

Yuwei Cao[1], Hao Peng [2], Jia Wu [3], Yingtong Dou [1], Jianxin Li [2], Philip S. Yu [1]

Paper: https://arxiv.org/pdf/2101.08747.pdf
Code: https://github.com/RingBDStack/KPGNN

[1] UIC [2] [3] MACQUARIE University

Problem Description

- Social event (e.g., the Notre-Dame Cathedral fire) reflect group social behaviors and wide-spread public concerns.
- Social event detection has many applications in fields including crisis management, product recommendation, and decision making.
- Social event detection can be formalized as **extracting clusters of co-related messages from social streams (i.e., sequences of social media messages)** to represent events.

Challenges and Our Solution

- **Challenges**
  - We should leverage the rich semantic and relational information on Twitter.
  - The model should acquire, preserve, and extend knowledge given the streaming social messages.
- **Our solution:**
  - Use GNNs to learn the semantic and structural information.
  - Propose an incremental learning framework which has the pre-training, detection, and maintenance stages.

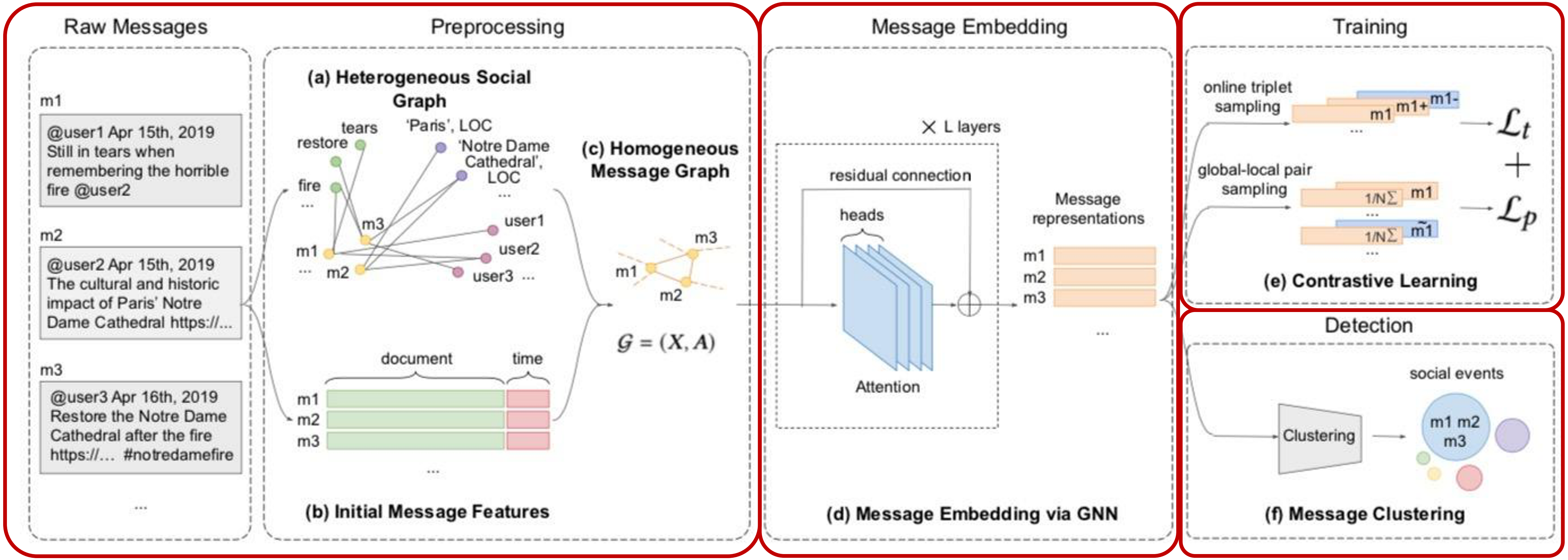
Dataset Information

- **Twitter Event Corpus**
  - 68,841 manually labeled tweets related to 503 event classes, spreading over a period of four weeks.

- 'event_id': manually labeled event class
- 'tweet_id': tweet id 'text': content of the tweet
- 'created_at': timestamp of the tweet
- 'user_id': the id of the sender
- ‘user_loc’, ‘place_type’, ‘place_full_name’: the location of the sender
- 'hashtags': hashtags contained in the tweet
- 'user_mentions': user mentions contained in the tweet
- 'image_urls': links to the images contained in the tweet
- 'entities': a list, named entities in the tweet (extracted using spaCy)
- 'words': a list, tokens of the tweet (hashtags and user mentions are filtered out)
- 'filtered_words': a list, lower-cased words of the tweet
- 'sampled_words': a list, sampled words of the tweet

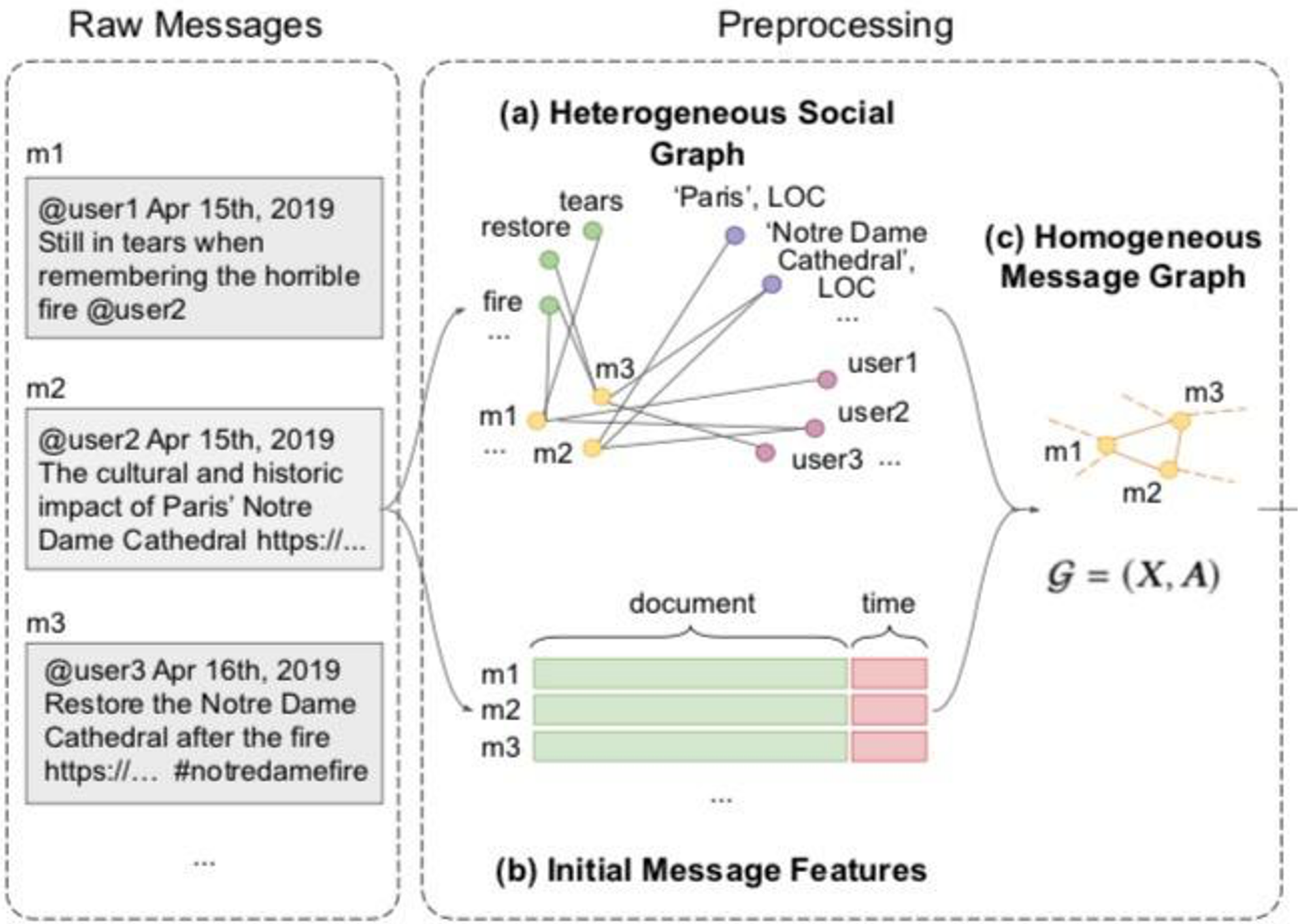
Proposed Method - KPGNN

Unsupervised Loss Function



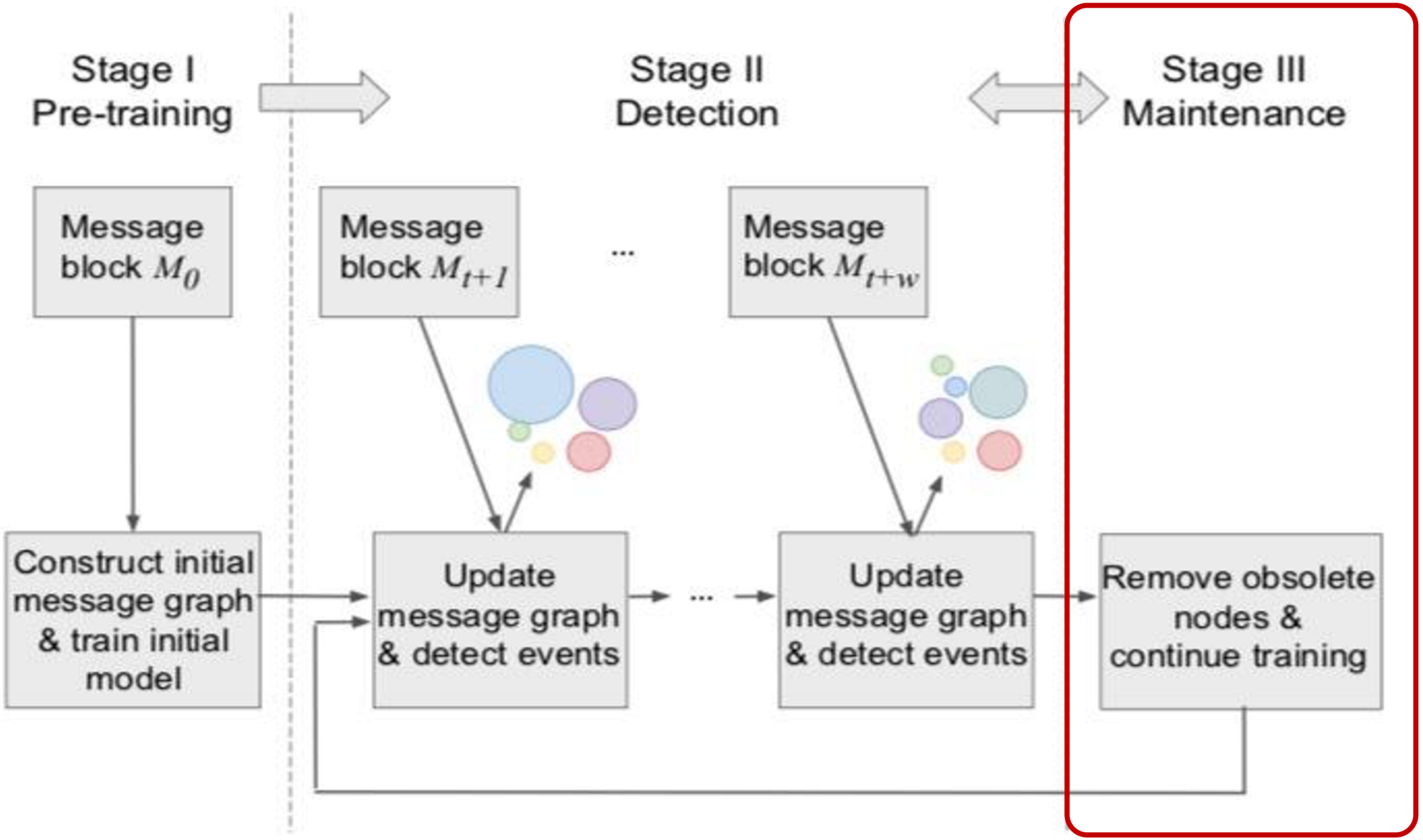
Graph Construction **GNN Training** **Detection Event By Clustering**

KPGNN – Graph Construction



- **(a):** word nodes; name entity nodes: user nodes; tweet message nodes.
- **(a) → (c):** If two message have **at least one common neighbor**, we add one edge between them.
- **(b)** Message feature vector contains natural language semantics and temporal information.

KPGNN -- Incremental Learning Lifecycle



Nodes Removing Strategy

1. Keep all nodes.
2. Keep the relevant nodes.
3. Keep the latest nodes.

Experiment Setup

- **Baselines**
  - Word2vec
  - LDA
  - WMD
  - BERT
  - Bi-LSTM
  - PP-GCN
  - EventX
- **Metrics**
  - NMI
  - AMI
  - ARI
- **Implementation**
  - Python
  - PyTorch
  - Deep Graph Library

Experiment Result

Table 2: Offline evaluation results on the Twitter dataset. The best results are marked in bold and second-best in italic.

Metrics	Word2vec [26]	LDA [3]	WMD [20]	BERT [6]	BiLSTM [12]	PP-GCN [28]	EventX [21]	$KPGNN_t$	KPGNN
NMI	.44±.00	.29±.00	.65±.00	.64±.00	.63±.00	.68±.02	**.72±.00**	.69±.01	*.70±.01*
AMI	.13±.00	.04±.00	.50±.00	.44±.00	.41±.00	.50±.02	.19±.00	*.51±.00*	**.52±.01**
ARI	.02±.00	.01±.00	.06±.00	.07±.00	.17±.00	.20±.01	.05±.00	*.21±.01*	**.22±.01**

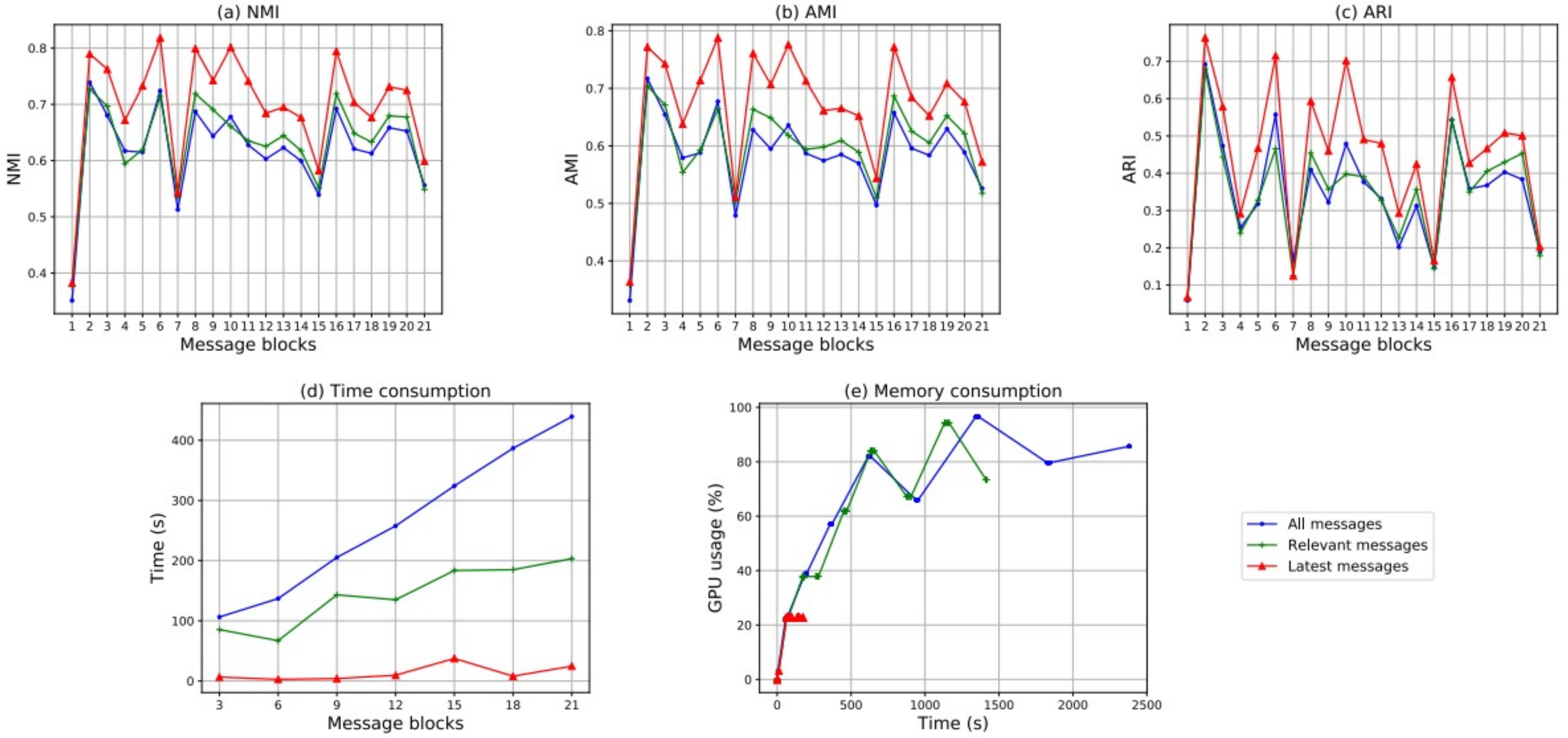
Performance on static dataset

Table 4: The statistics of the social stream.

Blocks	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
# of messages	20, 254	8, 722	1, 491	1, 835	2, 010	1, 834	1, 276	5, 278	1, 560	1, 363	1, 096
Blocks	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}	M_{21}
# of messages	1, 232	3, 237	1, 972	2, 956	2, 549	910	2, 676	1, 887	1, 399	893	2, 410

Incremental evaluation setting

Different Nodes Removing Strategies



Thank You!

Q & A

Yingtong Dou
Department of Computer Science
University of Illinois at Chicago
Homepage: http://ytongdou.com
Email: ydou5@uic.edu
Twitter: @dozee_sim

UNIVERSITY OF ILLINOIS CHICAGO