Supplementary Material for "TecHNet: Scalable Temporal Hypergraph Neural Network"

Yuanyuan Xu
University of New South Wales
yuanyuan.xu@unsw.edu.au

Wenjie Zhang
University of New South Wales
wenjie.zhang@unsw.edu.au

Ying Zhang
Zhejiang Gongshang University
ying.zhang@zjgsu.edu.cn

Xiwei Xu
CSIRO Data61
xiwei.xu@data61.csiro.au

Xuemin Lin
Shanghai Jiao Tong University
xuemin.lin@gmail.com

1 PRELIMINARIES AND BACKGROUNDS

1.1 One of the Motivation

As discussed in Section 3.3, the existing solutions suffer from the *over-smoothing* problem due to the multi-hop hyperedge-centric neighbors and hyperedge-centric message passing. To illustrate, we take the CATWalk as an example, visualizing the sampled hyperedge neighbors for three randomly selected hyperedges, where we set $L = 3$ and $k = 6$. For clarity, we renumber the IDs of these hyperedge-centric neighbors from 0 to $K^L \cdot |H_e| - 1$. We observe that (1) For both H_1 and H_2, it is common to find repeated neighbors among multi-hop hyperedge neighbors, with the maximum frequency reaching 114. (2) The frequency distribution is long-tailed: a few hyperedges are seen more than 100 times, but most appear fewer than 5 times. These frequently occurring hyperedges may overwhelm the hyperedge embeddings' expressiveness, resulting in sub-optimal embedding quality and poor generalization capability. The above observation motivates us to develop more effective techniques for hyperedge-centric neighbor extraction and the message passing mechanism.

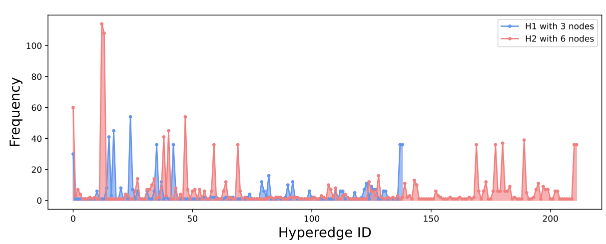

Figure 1: The frequency of hyperedge sampled by CATWalk for two incoming hyperedges (H_1 and H_2) on the Congress Bills dataset, where L and K are set to 3 and 6, respectively. We re-number sampled hyperedges and count them.

2 THEORETICAL RESULTS

2.1 Proof of Theorem 1

Theorem 1. *Given a new hyperedge (e_i, t) with m nodes at time t, denoted by $\mathcal{V}_i = \{v_1, \cdots, v_m\}$, and suppose the hash tables corresponding to these nodes are already full at time t. For any node v_j within the hyperedge (e_i, t), its newest hyperedge-centric neighbors can be inserted in its hash table $\mathcal{N}_{v_j}^{t^-}(K)$ with probability $Pr \propto \exp\left(\frac{\theta\cdot\lambda}{K}(t - t^-)\right)$, where λ is a constant, θ is the hyperparameter that controls this retention probability, and t^- is the previous timestamp.*

Proof. Here, we assume that hyperedges come in for any node (e.g., v) by following a Poisson process with a constant intensity (e.g., λ). Let v_m be one of the historical temporal neighbors of v_j, where $v_j, v_m \in (e_i, t)$. According to Section 4.2.2, v_m is hashed to the position $hash(v_m, t)$, and the replacement probability that any other temporal neighbors are inserted to $\mathcal{N}_{v_j}^{t^-}(K)$ is θ. Since we suppose v_m is already inserted to $\mathcal{N}_{v_j}^{t^-}(K)$, we only need to evaluate the probability that it does not get replaced by another temporal neighbor. We define the event of hyperedges with v_j arriving since t^- as Θ_{v_j} and we know from the Poisson progress that

$$\mathbb{E}[\Theta_{v_j}] = \lambda \cdot (t - t^-). \tag{1}$$

On average, each hyperedge has probability $\frac{\theta}{K}$ of replacing (v_m, t). Then, we define the event of replacement for $hash(v_m, t)$ as $\Theta_{hash(v_m,t)}$ and we have

$$\mathbb{E}[\Theta_{hash(v_m,t)}] = \frac{\theta \cdot \lambda \cdot (t - t^-)}{K}. \tag{2}$$

By the property of a Poisson process, the probability that none of the hyperedge-centric neighbors get inserted into the same position from t^- to t is $\exp\left(\frac{\theta\cdot\lambda}{K}(t - t^-)\right)$. Last, we have

$$Pr((v_m, t) \in \mathcal{N}_{v_j}^{t^-}(K)) \propto \exp\left(\frac{\theta \cdot \lambda}{K}(t - t^-)\right) \tag{3}$$

Overall, the proof is completed. □

2.2 Proof of Lemma 1

Lemma 1. *Hyper2Token is permutation invariant and is a universal approximation of invariant multi-set functions.*

Proof. Let $\pi(S)$ be a given permutation of set S, we aim to show that $\Psi(S) = \Psi(\pi(S))$. We first recall the Hyer2Token: Let $S = \{z_1(t), \cdots, z_{|H_e|}(t)\}$, where $z_i(t) \in \mathbb{R}^{d_n}$, be the input set and $Z_e(t)$ be its matrix representation:

$$\Psi(Z_e(t)) = \frac{1}{|H_e|} \cdot \sum_{i=1}^{|H_e|} \mathrm{MLP}(z_i(t)), \tag{4}$$

$$= \mathrm{MEAN}(\mathrm{MLP}(z_1(t)), \cdots, \mathrm{MLP}(z_{|H_e|}(t))). \tag{5}$$

Let $\Psi(\pi(S)) = [z_{\pi(1)}(t), \cdots, z_{\pi(|H_e|)}(t)]$ be a permutation of the input matrix $Z_e(t)$. Accordingly, we can see the output of the

Table 1: Dataset Statistic.

Alias	Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	#Timestamps	max $\lvert H_{\mathcal{E}}\rvert$	average length $\overline{\lvert H\rvert_e}$	Domain
EE	Email-Enron	143	10,883	10,788	18	2.5	Social network
NC	NDC-Classes	1,161	49,724	5,891	24	3.2	Drug network
NS	NDC-Substances	5,311	112,405	7,734	25	2.6	Drug network
UT	Users-Threads	125,602	192,947	189,917	14	2.2	Social network
CB	Congress-Bills	1,718	260,851	5,936	25	4.3	Bill network
TMS	Threads-Math-Sx	176,445	719,792	718,340	21	2.5	Social network
CD	Coauth-DBLP	1,924,991	3,700,067	83	25	3.0	Scholar network
TSO	Threads-Stack-Overflow	2,675,955	11,305,343	11,260,218	25	3.5	Social network

Hyper2Token is permuted by $\pi(\cdot)$:

$$\Psi(\pi(Z_e(t))) = \text{MEAN}(\text{MLP}(z_{\pi(1)}(t)), \cdots, \text{MLP}(z_{\pi(|H_e|)}(t))), \tag{6}$$

$$= \text{MEAN}(\pi(\text{MLP}(z_1(t)), \cdots, \text{MLP}(z_{|H_e|}(t)))), \tag{7}$$

$$= \Psi(Z_e(t)). \tag{8}$$

In the last step, we leverage the property that MEAN$(\cdot)$ is permutation invariant. Based on Eqs. (6)-(8), we can see that Hyper2Token possesses permutation invariance. Given that Hyper2Token employs a two-layer MLP, it functions as a universal approximator capable of modeling any function. Therefore, Hyper2Token serves as a universal approximator. □

3 MODEL TRAINING

During the training phase, for each hyperedge in the training set, we adopt the commonly used negative sample generation method [9] to generate a negative sample. Next, for each hyperedge in the training set such as $(e, t) = \{v_1, v_2, \ldots, v_m\}$, including both positive and negative samples. During the batch processing, we do the node-wise and hyperedge-wise message passing to generate the hyperedge embeddings. For the hyperedge prediction, we use a 2-layer perception over hyperedge embeddings to generate a predicted score, which can be plugged into the cross-entropy loss for training or compared with a threshold to make the final prediction.

4 EXPERIMENTAL SETTING

4.1 Datasets

We employ 8 real-world datasets for model evaluation. We provide a more detailed introduction in Table 1, including the averaged hyperedge length $\overline{|H_e|}$ and application domains.

4.2 Baselines

- NHP [13] is a classical neural hyperlink prediction method, which is an enhanced version of the self-attention-based graph convolutional network for hypergraphs (HyperSAGC N) [14]. We use the time encoding function to encode timestamps and regard them as the hyperedge features. The source code is provided at [1].
- CHESHIRE [8] treats a hyperedge as a fully connected graph (clique) and uses a Chebyshev spectral graph convolution neural network to refine the embeddings of the nodes within the hyperedge, which is a hyperedge prediction method. We also use the time encoding function to encode timestamps and regard them as the hyperedge features. The source code is provided at [4].
- CE-CAW: We apply CAW [12] on the CE of the temporal hypergraph. CAW is a temporal edge prediction method that uses causal anonymous random walks to capture the dynamic laws of the network in an inductive manner. The source code is provided at [2].
- CE-Zebra: We apply Zebra [11] on the CE of the temporal hypergraph. Zebra accelerates the computation of T-GNN by directly aggregating the features of its neighbors returned by the top-k temporal Personalized PageRank (T-PPR). The source code is provided at [6].
- CE-Orca: We apply Orca [10] on the CE of the temporal hypergraph. Orca proposes a caching-based framework to address the neighborhood explosion problem by non-trivially caching and reusing intermediate embeddings. The source code is provided at [5].
- CATWalk [7] directly models the temporal and high-order structures to generate hyperedge embeddings. Concretely, it extracts hyperedge-centric neighbors by designing a set-based sampling method and encodes sampled hyperedges and nodes using the permutation invariant message passing. The source code is provided at [3].

REFERENCES

[1] 2020. NHP. https://drive.google.com/file/d/1pgSPvv6Y23X5cPiShCpF4bU8eqz_34YE/view.
[2] 2021. CAW. https://github.com/snap-stanford/CAW.
[3] 2023. CATWalk. https://github.com/ubc-systopia/CATWalk.
[4] 2023. CHESHIRE. https://github.com/canc1993/cheshire-gapfilling.
[5] 2023. Orca. https://github.com/LuckyLYM/Orca.
[6] 2023. Zebra. https://github.com/LuckyLYM/Zebra.
[7] Ali Behrouz, Farnoosh Hashemi, Sadaf Sadeghian, and Margo Seltzer. 2024. CAT-Walk: Inductive Hypergraph Learning via Set Walks. *Advances in Neural Information Processing Systems* 36 (2024).
[8] Can Chen, Chen Liao, and Yang-Yu Liu. 2023. Teasing out missing reactions in genome-scale metabolic networks through hypergraph learning. *Nature Communications* 14, 1 (2023), 2375.
[9] Can Chen and Yang-Yu Liu. 2023. A survey on hyperlink prediction. *IEEE Transactions on Neural Networks and Learning Systems* (2023).
[10] Yiming Li, Yanyan Shen, Lei Chen, and Mingxuan Yuan. 2023. Orca: Scalable Temporal Graph Neural Network Training with Theoretical Guarantees. *Proceedings of the International Conference on Management of Data* 1, 1 (2023), 52:1–52:27.
[11] Yiming Li, Yanyan Shen, Lei Chen, and Mingxuan Yuan. 2023. Zebra: When Temporal Graph Neural Networks Meet Temporal Personalized PageRank. *Proceedings of The VLDB Endowment* 16, 6 (2023), 1332–1345.

[12] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2021. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks. In *Proceedings of International Conference on Learning Representations (ICLR)*. OpenReview.net.
[13] Naganand Yadati, Vikram Nitin, Madhav Nimishakavi, Prateek Yadav, Anand Louis, and Partha Talukdar. 2020. NHP: Neural Hypergraph Link Prediction. In *Proceedings of the ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, 1705–1714.
[14] R Zhang, Y Zou, and J Ma. 2020. Hyper-SAGNN: a self-attention based graph neural network for hypergraphs. In *Proceedings of the International Conference on Learning Representations*.