Semantics-Enhanced Temporal Graph Networks for Content Popularity Prediction

Jianhang Zhu, *Graduate Student Member, IEEE*, Rongpeng Li, *Member, IEEE*, Xianfu Chen, *Member, IEEE*, Shiwen Mao, *Fellow, IEEE*, Jianjun Wu, and Zhifeng Zhao, *Member, IEEE*

***Abstract*—The surging demand for high-definition video streaming services and large neural network models implies a tremendous explosion of Internet traffic. To mitigate the traffic pressure, architectures with in-network storage have been proposed to cache popular contents at devices in closer proximity to users. Correspondingly, in order to maximize caching utilization, it becomes essential to devise an effective popularity prediction method. In that regard, predicting popularity with dynamic graph neural network (DGNN) models achieves remarkable performance. However, DGNN models still suffer from tackling sparse datasets where most users are inactive. Therefore, we propose a reformative temporal graph network, named semantics-enhanced temporal graph network (STGN), which attaches extra semantic information into the user-content bipartite graph and could better leverage implicit relationships behind the superficial topology structure. On top of that, we customize its temporal and structural learning modules to further boost the prediction performance. Specifically, in order to efficiently aggregate the diversified semantics that a content might possess, we design a user-specific attention (`UsAttn`) mechanism for the temporal learning. Unlike the attention mechanism that only analyzes the influence of genres on content, `UsAttn` also considers the attraction of semantic information to a specific user. Meanwhile, as for the structural learning, we introduce the concept of positional encoding into our attention-based graph learning and novelly adopt a semantic positional encoding (`SPE`) function, which effectively boost the performance of lightweight algorithms. Finally, extensive simulations verify the superiority of our models and demonstrate their effectiveness in content caching.**

***Index Terms*—Content caching, dynamic graph neural network, popularity prediction, semantics.**

Manuscript received 5 April 2023; revised 23 October 2023; accepted 28 December 2023. Date of publication 3 January 2024; date of current version 2 July 2024. This work was supported in part by the Zhejiang Key Research and Development Plan under Grant 2022C01093, in part by the National Natural Science Foundation of China under Grant 62071425, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005. A part of this paper has been accepted by the 2023 IEEE International Conference on Communications [DOI: 10.1109/ICC45041.2023.10279564]. Recommended for acceptance by W. Gao. *(Corresponding author: Rongpeng Li.)*

Jianhang Zhu and Rongpeng Li are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: zhujh20@zju.edu.cn; lirongpeng@zju.edu.cn).

Xianfu Chen is with the VTT Technical Research Centre of Finland, 90570 Oulu, Finland (e-mail: xianfu.chen@ieee.org).

Shiwen Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201 USA (e-mail: smao@ieee.org).

Jianjun Wu is with the Huawei Technologies Company, Ltd., Shanghai 201206, China (e-mail: wujianjun@huawei.com).

Zhifeng Zhao is with the Zhejiang Lab, Hangzhou 311121, China, and also with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: zhaozf@zhejianglab.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2023.3349315, provided by the authors.

Digital Object Identifier 10.1109/TMC.2023.3349315

I. Introduction

The surging demand for high-definition video streaming services and large neural network models results in tremendous pressure on the Internet [2], [3], [4]. It is pointed out that caching popular contents in advance has the potential to reduce the backhaul traffic up to 35% [5]. Correspondingly, in-network caching for multimedia contents emerges as a promising technique and garners extensive attention [6], [7]. Moreover, Ref. [8] outlines a framework that manages and orchestrates diverse deep neural network (DNN) models at the network edge to satisfy heterogeneous service requirements, and demonstrates its contribution on alleviating peak backhaul traffic as well. Therefore, large DNN models are as equally marketable as multimedia videos in content caching [9]. But for the sake of simplicity and dataset availability, the prime focus of our investigation remains centered on the video content, with a flexible extension to cache DNNs. On the other hand, compared with the continual explosion of content volume, it is infeasible to increase the device caching capability immoderately due to the practical economic and technical limitations [10]. This predicament makes the design of competent caching strategies much more crucial, wherein accurate popularity prediction plays a decisive role.

Recently, DNNs have demonstrated their remarkable potential in unveiling the embedded temporal correlation for popularity prediction [11], [12], [13]. Meanwhile, along with users requesting contents, the interactions between users and contents gradually constitute a dynamic bipartite interaction graph. Some recent graph neural network (GNN) model-based methods, which resort to exploiting the inherent structural pattern in such a bipartite graph, manifest themselves in providing superior prediction accuracy within a recommendation system (RS) [14], [15]. In particular, such GNN models enable us to speculate for inactive users with few requests in an interaction-intense graph by associating them with other active users that exhibit similar behaviors. However, these GNN models [14], [15] are contingent on an assumption of a static bipartite graph. In order to blend the merits of both structural learning and temporal learning, recommendation with dynamic graph neural network (DGNN) models emerges [16], which is always synergistic with caching [17]. Thus, caching with DGNN models also achieves satisfactory improvement [18].

Nevertheless, our prior work in [18] discovers that the model's performance is not gratifying for cases where most users in

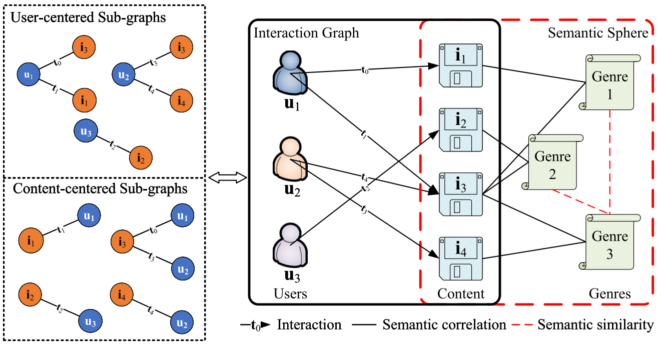


Fig. 1. Example of the dynamic interaction graph and the implicit semantic relationships between contents.

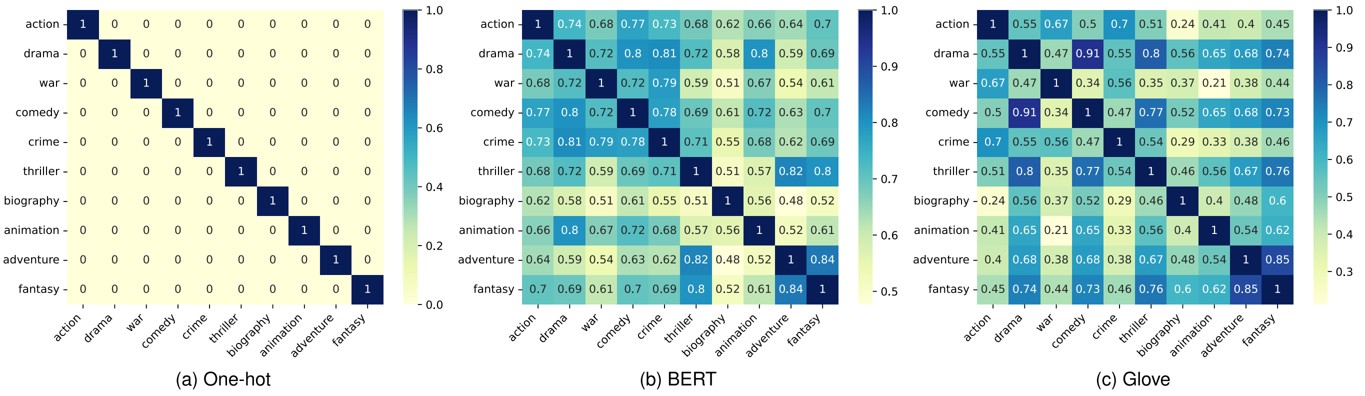
the bipartite graph are inactive. To overcome the data sparsity, enlarging the receptive field of the DGNN model (e.g., the stacking of GNN layers or an increase in the number of first-order neighbors) can be productive [19], but it usually incurs significant computational complexity and time cost. On the contrary, along with fine-grained methods, enlarging the breadth of available information, such as considering the side information of users and contents in DGNN, sounds more appealing [20], [21]. Specifically, the side information enriches the sparse graph and may also reflect user's intention, both of which would benefit the reasoning and interpretability of user's future behavior. But given the importance of user privacy, it is more appropriate to conduct an excavation for the general content information (e.g., analyzing the semantic correlations among the genre information of contents).

Fig. 1 presents an example of the dynamic interaction graph and the implicit semantic relationships between contents. The requests of two users, u_1 and u_2, only intersect at content i_3. The sparsity of data makes it intractable for classical GNN-based methods to accurately predict the preference of u_1 for content i_4. On the other hand, the content genre information and their underlying similarities in the semantic sphere, which are indicated by the red dotted lines in Fig. 1, reveal a strong correlation between i_4 and i_3 as well as a weak correlation between i_4 and i_1, respectively. Thanks to the attachment of semantics, these two kinds of underlying connectivity between i_4 and i_1 are unveiled to assist the prediction, and an inference that u_1 is likely to request i_4 can be boldly triggered. Furthermore, as demonstrated in Fig. 2, there exist several natural language processing (NLP) methods, such as one-hot, BERT [22], and Glove [23], available for computing the semantic similarities. It is also natural to conjecture that a more precise computation of semantic similarities might benefit a superior speculation.

In this paper, we propose a semantics-enhanced temporal graph network (STGN) to strengthen the DGNN model performance in dealing with sparse datasets and improve popularity prediction for content caching. In particular, semantics ameliorates the temporal learning module to better track the dynamical variations in a user-content bipartite graph, and circumvents the difficulties of discovering patterns in a sparse dataset through the supplement to content-centered sub-graphs for the structural learning. Besides, we adopt several mature NLP methods to encode genre messages as semantic information, and then treat the embedded semantic information as part of the input to the predictive model. Additionally, considering that a content might possess multiple genres (e.g., a fictional action movie containing both fiction and action genres) and the predilection might vary across users as well, we further design a user-specific attention (`UsAttn`) mechanism for a more fine-grained aggregation of various semantics, so as to improve the utilization of the diversified semantic features. Unlike the attention mechanism that only considers the influence of genres in content, `UsAttn` leverages user-content pairs in the bipartite graph to calculate the attention coefficients and capably analyzes the attraction of semantic information to a specific user during the prediction. Meanwhile, given the complication of distinguishing one specific user from a content-centered sub-graph where the same content contains massive user relevancy, as shown in Fig. 1, the aforementioned enhancement with `UsAttn` cannot be applied into the semantic analysis in our attention-based structural learning module. Instead, it requires some alternative options from an innovative perspective. Specifically, inspired by the preliminary effectiveness of a dot-product-based positional encoding (PE) function in Transformer [24], we develop a semantic positional encoding (`SPE`) function, deduced from a Fourier kernel-based method, to improve the effectiveness of incorporating multi-dimensional semantics in structural learning. Particularly, we think that it might provide a viable answer to efficiently incorporate and utilize the aggregated semantics on top of DGNNs, especially for the lightweight models. Furthermore, we apply our proposed model to a caching strategy in a multi-tier caching system and conduct extensive simulations to evaluate its superiority. In brief, the main contributions of this paper are summarized as follows.

- To deal with the data sparsity, we propose an STGN, which leverages the implicit connections between requested contents and their semantic features from both temporal and structural learning perspectives.
- Motivated by the fact that a content usually carries rich semantic information, we devise a `UsAttn` mechanism to exploit potential semantic correlations within the user-content bipartite graph and enhance the temporal learning.
- In order to improve the effectiveness of semantics in structural learning, we incorporate a theoretical-grounded multi-dimensional `SPE` from the Fourier kernel into the attention-based graph learning module, which benefits the lightweight models particularly.
- Extensive experiments based on a real-world dataset verify the improvement in prediction performance achieved by our STGN model and validate the effectiveness of the STGN model-based proactive caching strategy in terms of several widely-adopted metrics (i.e., cache hit rate, cumulative transmission delay and hop count [25], [26], [27]).

The remainder of this paper is organized as follows. The related works are discussed in Section II. Then, we introduce system models and formulate the problem in Section III. We elaborate on the details of the proposed prediction model and its modified versions for effective semantic learning in Sections IV



(a) One-hot (b) BERT (c) Glove

Fig. 2. Visualization of cosine similarities between some genres' embeddings computed by different NLP methods.

TABLE I
MAJOR NOTATIONS USED IN THE PAPER

Notation	Definition
u_j, i_k	User j and content k
$\mathcal{U}, \mathcal{I}$	The sets of users and contents
$\boldsymbol{v}_{u_j}, \boldsymbol{v}_{i_k}, \boldsymbol{e}_{jk}$	Raw features of u_j, i_k and their edges
$\mathcal{V}_{\mathcal{U}}, \mathcal{V}_{\mathcal{I}}, \mathcal{E}$	The raw feature sets of users, contents and their edge
$p_{jk}(\hat{T})$	Real preferences of u_j for i_k at $\hat{T}$
$\tilde{p}_{jk}(\hat{T})$	Predicted preferences of u_j for i_k at $\hat{T}$
p_{thre}	Threshold value for judging
$\text{Pop}^{i_k}(\hat{T})$	Popularity of i_k at $\hat{T}$
$\mathbb{P}^{i_k}(\Delta_P)$	Popularity of i_k during the cache updating period Δ_P
δ_p	Popularity predicting period
K_1, K_2, K_3	Maximum caching capability for devices in *Tier 1, 2, 3*
$\mathcal{C}(\Delta_P)$	Real request set during the update period
$\widetilde{\mathcal{C}}(\Delta_P)$	Predicted popularity set for all contents during Δ_P
$\widetilde{\mathcal{C}}_K(\Delta_P)$	Caching set with size K during Δ_P
$h(\Delta_P)$	Cache hit rate
$\mathbf{Msg}_{jk}$	Message that merges all raw features of an interaction
$\mathbf{Msg}'_{jk}$	The semantics-enhanced message
$\mathbf{h}_j(\hat{T})$	Short-term preference of u_j
$\mathbf{Mem}_j$	Long-term preference of u_j
$\mathbf{Mem}'_j$	The updated long-term preference of u_j concatenated with extra features
$\mathbf{E}_{u_j}(\hat{T}), \mathbf{E}_{i_k}(\hat{T})$	Final embedding representations for u_j and i_k
$\mathbf{E}_{jk}$	User-specific embedding
$\boldsymbol{s}_{kN_k}$	The N_k-th semantic information of i_k
$\boldsymbol{S}_{jk}$	User-specific semantic feature for u_j and i_k
$\boldsymbol{S}_k$	Aggregated semantic feature for i_k

and V, respectively. In Section VI, we present the experimental results and discussions. Finally, we draw the conclusion and future directions in Section VII.

For convenience, we also list the mainly used notations of this paper in Table I.

II. RELATED WORK

A. Content Caching and Popularity Prediction

Traditionally, albeit the remarkable portability, the widely-mentioned reactive caching strategies, such as least recently used (LRU) and least frequently used (LFU), only focus on the patterns of local requests, thus failing to handle the unexpected requests [28]. Accordingly, it becomes inevitable to design proactive caching strategies, wherein accurate popularity prediction plays a decisive role [10]. With the development of artificial intelligence (AI), applying DNNs to predict popularity has thrived. For instance, in Ref. [11], a model based on stacked autoencoders (SAE) is proposed to compute the popularity from the content request sequence. In addition, Refs. [12], [13] use a recurrent neural network (RNN) and its variant, long short-term memory (LSTM), to discover the patterns within the temporal content requests, so as to facilitate popularity-assisted content caching. Nevertheless, due to the insufficient historical data, LSTM or other sequence-based prediction models may fail to accurately predict for those inactive users.

Furthermore, the user-content interactions constitute a bipartite graph and lay the very foundation for adopting GNN to enhance the learning performance [15]. Although GNN has won remarkable achievement in RS [14], most existing works assume that the underlying graph is static, which does not conform to the real-life [16]. Consequently, popularity prediction with DGNN has been attracting significant attention. Different from the conventional GNN, a DGNN model is able to jointly learn the structural and temporal patterns of dynamic graphs. For example, Ref. [29] proposes a DyRep model to calculate the dynamic graph with a recurrent architecture. Besides, in Ref. [30], the authors employ a temporal graph convolutional network (T-GCN) model that combines GCN and the gated recurrent unit for traffic forecasting. To further improve the temporal learning and ameliorate the scalability issue of T-GCN, a spatial-temporal prediction algorithm stacking the dilated temporal convolution network (TCN) and the dynamic GCN is proposed in Ref. [31] to make prediction for mobile multi-sensor network. Learning from the "positional encoding" of the self-attention mechanism in Transformer [24], Ref. [19] proposes a "time encoding function" to encode the timestamp information for the graph attention network (GAT) [32], which is called the temporal graph attention mechanism (TGAT). Notably, due to the utilization of GAT, it also mitigates the universal deficiencies of GCN in previous works, i.e., the scalability and neglect of importance difference among vertexes. TGN in [33] introduces an additional temporal learning module on top of the TGAT for a deeper refinement of the temporal characteristics. Ref. [18] optimizes the temporal learning module of TGN with an age of information (AoI) based attention mechanism to filter and aggregate fresh historical messages, and realizes satisfactory results in content caching.

TABLE II
SUMMARY OF DIFFERENCES WITH RELATED LITERATURE ON POPULARITY PREDICTION

	Temporal	Structural	Sparsity	Brief Description
[12], [13]	●	○	○	Hard to predict for inactive users
[18], [29]–[31], [33]	●	●	○	Failing to handle the data sparsity
[19]	●	●	◑	Ignoring computational efficiency
[20], [21], [34]–[37]	○	●	◑	Ignoring the data accessibility
Ours	●	●	●	Learning with accessible data comprehensively and efficiently

Notations: ○ *indicates not included;* ● *indicates fully included;* ◑ *means partially included.*

Nevertheless, when most users in the graph are inactive, it is in general difficult to obtain satisfactory caching performance by deploying existing DGNN models in a straightforward way.

B. Tackling Data Sparsity

To overcome the data sparsity, Ref. [19] suggests that it is beneficial to stack more TGAT layers for enlarging the receptive field, but it comes at the expense of massive computation cost. On the other hand, some works attempt to solve this problem by supplementing the side information of the bipartite graph (e.g., user social influence [20]). Refs. [21], [34], [35], [36], [37] propose to introduce a knowledge graph (KG) for incorporating the side information of the requested contents into a static graph model, which leverages the implicit associations among the contents and yields superior prediction performance. However, these works ignore the dynamics of the interaction graph, while the KG construction also implies the demand for a significant amount of side information (e.g., the director and release date of the content), which may not be available in many cases. Therefore, it is more worthwhile to leverage limited content information (e.g., genre information) with a deeper excavation. In that regard, the astonishing development of NLP, such as one-hot, BERT [22], and Glove [23], makes it promising to capture the implicit semantic relations between the words.

Finally, in order to further highlight the contributions of our work, we summarize the distinctions between our method and other relevant literature on popularity prediction in Table II.

C. Positional Encoding in Transformer

It is meaningful to develop effective means of computing embeddings, so as to better unveil correlations. As for the temporal learning, it is simple and sufficient to adopt a `UsAttn`-based mechanism to discover the relationships between multiple genres related to contents and users. However, it becomes troublesome for the application in structural learning module, considering the massive relevance among users to the same content. The illuminated work in Ref. [19], which generalizes the definition of position and encodes the timestamps with a customized PE function, motivates us to treat the semantic information as a special kind of position. Therefore, we adopt an `SPE` to strengthen the association analysis between two semantics-attached content embeddings. As a specially designed PE function, it is also inspired by works with learnable approaches to encode positions [22], [38]. Besides, considering the heavy computational cost and non-uniform decay in different dimensions to encode each dimension independently before the concatenation [39], [40], the proposed `SPE` treats the multi-dimensional position as a whole and then encodes it directly by learnable Fourier features [39]. To our best knowledge, our `SPE` belongs to the first work to view the multi-dimensional semantic information as the position and encode from the perspective of Fourier features.

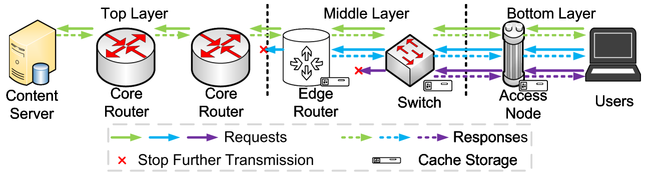


Fig. 3. Requests and responses in a multi-tier caching system.

III. SYSTEM MODELS AND PROBLEM FORMULATION

A. System Models

1) Network Model: As shown in Fig. 3, we concentrate on a multi-tier caching system, where caches are scattered over the devices close to users, such as the *edge routers*, *switches*, and some *access nodes*. We conceptually simplify the network as a three-layer topology as below.

- *Top Layer* – It is composed of *core routers*, which are responsible for connecting content providers with other network elements.
- *Middle Layer* – It encompasses *edge routers* and *switches*. In particular, the *switches* usually connect various devices in a network and communicate with the core network through the *edge routers*. And the location of *switches* is lower than the *edge routers*, while they are in the same layer.
- *Bottom Layer* – It consists of *access nodes*, which are deployed to connect users with the *switches*.

In this paper, we primarily take account of the in-network caching capability of the devices in *Bottom* and *Middle* layers, i.e., the *access nodes*, *switch*, and *edge router*, and denote them as *Tier 1*, *Tier 2*, and *Tier 3*, respectively. Moreover, as depicted in Fig. 3, once a copy of the target content is cached at a lower-tier device, the request will be directly responded and no longer be sent to any higher-tier devices.

2) Request Model: In this paper, we model the request records in the format of user-content pairs as a graph. We denote the set of users as $\mathcal{U} = \{u_0, u_1, \ldots, u_j\}$ and the set of contents as $\mathcal{I} = \{i_0, i_1, \ldots, i_k\}$, where u_j and i_k denote user j and content k, respectively. Furthermore, according to the indices, we allocate the randomly-initialized embeddings as their raw features and create the sets of users and contents, which are deemed as the input of the predictive model. The raw feature sets of users and contents are denoted as $\mathcal{V}_{\mathcal{U}} = \{\boldsymbol{v}_{u_0}, \boldsymbol{v}_{u_1}, \ldots, \boldsymbol{v}_{u_j}\}$ and $\mathcal{V}_{\mathcal{I}} = \{\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}, \ldots, \boldsymbol{v}_{i_k}\}$, where $\boldsymbol{v}_{u_j}$ and $\boldsymbol{v}_{i_k}$ are the vertexes in the dynamic bipartite graph corresponding to u_j and i_k, respectively. The interactions, i.e., users requesting contents, can be naturally regarded as the edges, which can be denoted as $\mathcal{E} = \{\boldsymbol{e}_{00}^{T_0}, \boldsymbol{e}_{01}^{T_1}, \ldots, \boldsymbol{e}_{jk}^{T_n}\}$. Herein, $\boldsymbol{e}_{jk}^{T_n}$ represents the embedding vector of interactions between u_j and i_k at T_n, and indicates the user-behavior type (e.g., watching videos or listening to music).

Next, we formulate the evolving interactions as a dynamic graph using a set of quadruples, $\mathcal{G} = \{(\boldsymbol{v}_{u_0}, \boldsymbol{v}_{i_0}, \boldsymbol{e}_{00}, T_0), \ldots, (\boldsymbol{v}_{u_j}, \boldsymbol{v}_{i_k}, \boldsymbol{e}_{jk}, T_n)\}$, where T_n denotes the timestamp of the n-th interaction.[1] In addition, we integrate each quadruple into a piece of historical message as the input of our DGNN model. For instance, the interaction occurred at T_n between u_j and i_k is formulated as $\mathbf{Msg}_{jk} = [\boldsymbol{v}_{u_j} || \boldsymbol{v}_{i_k} || \boldsymbol{e}_{jk} || T_n]$, where $||$ is the concatenation operator. Moreover, as presented before, content may contain various semantic genres, and we need to encode all N_k genres of content i_k with the NLP methods for fully utilizing inherent semantic characteristics in the subsequent prediction. Correspondingly, the encoded semantic features are represented as $\mathcal{S}_k = \{\boldsymbol{s}_{k1}, \ldots, \boldsymbol{s}_{kN_k}\}$.

B. Problem Formulation

In this paper, we evaluate the performance of our STGN model in caching task with the widely-accepted cache hit rate. Considering the existence of a maximum number of caching items K, only the top-K contents $\tilde{\mathcal{C}}_K(\Delta_P)$ in a popularity ranking list $\widetilde{\mathcal{C}}(\Delta_P)$ are cached during the cache updating period Δ_P. Given the real request set is $\mathcal{C}(\Delta_P)$, we calculate the hit rate during the cache updating period Δ_P with

$$h(\Delta_P) = \frac{\mathbf{I}(\mathcal{C}(\Delta_P), \tilde{\mathcal{C}}_K(\Delta_P))}{\mathbf{I}(\mathcal{C}(\Delta_P), \mathcal{C}(\Delta_P))}, \qquad (1)$$

where $\mathbf{I}(\mathcal{X}, \mathcal{Y})$ represents the hit number for the elements in $\mathcal{Y}$ to $\mathcal{X}$.

Based on (1), we can further calculate the cache hit rate $h_x(\Delta_P)$ for each tier, where $x \in \{1, 2, 3\}$ denotes the tier index in Fig. 3. Correspondingly, K_1, K_2, K_3 are the diverse caching capacity in each tier. Notably, for a more complicated network topology, where *Tier 1* contains several devices, we assume that the popularity ranking list of each equipment is calculated according to the requests of the attached users. Simultaneously, *Tier 2* and *3* cache the top-K_2 and top-K_3 contents in the overall ranking list $\widetilde{\mathcal{C}}(\Delta_P)$ excluding the cached contents in lower-tier devices.

Notably, though cache hit rate is sufficiently compelling for assessing the effectiveness of a caching strategy, and can partially reflect the influence of popularity prediction on content caching, it will be more intuitive to further assess the effectiveness of caching from a quality-of-service (QoS) perspective. Therefore, we also evaluate the performance of cumulative transmission delay and hop count, as suggested in [25], [26], [27]. Specifically, the transmission delay, which measures the elapsed time that users have to wait until receiving the first piece of data, is related to the location of content and the transmission rate. The hop count denotes the number of retrieving hops for catering user's demand, and a smaller hop count usually connotes a lower probability of network congestion. It can be observed that all the aforementioned metrics unanimously imply to cache the more popular content closer to users [41]. For simplicity, we primarily treat the cache hit rate as the foundation of our investigation.

In line with the previous analysis, it becomes essential to know the popularity of each content for the future cache update period Δ_P and obtain the popularity ranking list $\widetilde{\mathcal{C}}(\Delta_P)$ in advance. We assume the list is sorted based on the popularity combining outcomes from several time slots sampled by the predicting period $\delta_p \ll \Delta_P$. Therefore, the overall popularity of i_k during the update period Δ_P can be formulated as

$$\mathbb{P}^{i_k}(\Delta_P) = \sum_{n_\delta \in N_\delta} \text{Pop}^{i_k}(n_\delta \times \delta_p), \; \forall i_k \in \mathcal{I}, \qquad (2)$$

where $\mathcal{N}_\delta = \{0, 1, \ldots, \lfloor \frac{\Delta_P}{\delta_p} \rfloor\}$, $\lfloor\rfloor$ is a floor operator, and $\text{Pop}^{i_k}(\hat{T})$ represents the popularity of i_k at the future time $\hat{T} = n_\delta \times \delta_p$. Consequently, the popularity ranking list can be obtained to guide the content caching task. Notably, in order to distinguish the contents with the same popularity, we decide their prioritization consistent with LRU.

Obviously, the popularity $\text{Pop}^{i_k}(\hat{T})$ is indispensable, and it can be obtained by gathering the preferences of all users [42], which we calculate as

$$\text{Pop}^{i_k}(\hat{T}) = \sum_{u_j} \mathbf{1}\left(p_{jk}(\hat{T}) > p_{\text{thre}}\right), \; \forall i_k \in \mathcal{I}, \qquad (3)$$

where $p_{jk}(\hat{T})$ indicates the real preference of u_j for i_k at $\hat{T}$, p_{thre} is the threshold value for judging emergence of such a request, and $\mathbf{1}(\zeta)$ is an indicator function that only equals 1 if the condition ζ is satisfied.

As real preference p_{jk}[2] is unknown a priori, we aim to calculate a predicted result $\tilde{p}_{jk}$ with the embeddings of u_j and i_k at $\hat{T}$, namely $\mathbf{E}_{u_j}(\hat{T})$ and $\mathbf{E}_{i_k}(\hat{T})$. That is,

$$\tilde{p}_{jk}(\hat{T}) = F\left(\mathbf{E}_{u_j}(\hat{T}), \mathbf{E}_{i_k}(\hat{T})\right), \qquad (4)$$

where a multi-layer perceptron (MLP) can be adopted to realize the function $F(\cdot)$. In this regard, our target in (1) converts to generating feasible representations with the predictive model from the dynamic interaction graph, so as to minimize the binary cross entropy loss (BCELoss) between the real preference, p_{jk}, and the predicted one, $\tilde{p}_{jk}$, $\forall u_j \in \mathcal{U}$, $i_k \in \mathcal{I}$ [16],

$$\mathcal{L} = -\sum_{u_j, i_k} \left(p_{jk} \log(\tilde{p}_{jk}) + (1 - p_{jk}) \log(1 - \tilde{p}_{jk})\right). \qquad (5)$$

[1] Notably, for simplicity of representation, we omit the T_n in $\boldsymbol{e}_{jk}^{T_n}$ and the superscript j and k of the vertexes T_n^{jk}, which represents the n-th interaction that happens between u_j and i_k.

[2] Notably, for simplicity of representation, we omit the $\hat{T}$ of the $p_{jk}(\hat{T})$, $\tilde{p}_{jk}(\hat{T})$, the embeddings $\mathbf{E}_{u_j}(\hat{T})$ and $\mathbf{E}_{i_k}(\hat{T})$ in the following equations.

IV. Semantics-Enhanced Temporal Graph Network

In this section, we focus on the design of the STGN, so as to obtain the desired embedding representations, $\mathbf{E}_{u_j}(\hat{T})$ and $\mathbf{E}_{i_k}(\hat{T})$, $\forall u_j \in \mathcal{U}$, $i_k \in \mathcal{I}$, from a sparse dataset.

A. Conventional TGN

According to the different roles in prediction, the conventional TGN model consists of two prime segments, including the temporal learning module and the structural learning module.

1) Temporal Learning Module: The temporal learning module, which consists of a message aggregator and a memory updater, is adopted to compress a user's historical messages into a refined representation. Specifically, the message aggregator leverages several fresh historical messages of u_j before the prediction time $\hat{T}$ to obtain a compressed feature $\boldsymbol{h}_j(\hat{T})$. Thus, $\boldsymbol{h}_j(\hat{T})$ can also be deemed as a feature that is able to represent the short-term preference of u_j, which can be formulated as

$$\boldsymbol{h}_j(\hat{T}) = \mathrm{Agg}\left(\mathbf{Msg}_{j0}, \ldots, \mathbf{Msg}_{jk}\right), \tag{6}$$

where $\mathrm{Agg}(\cdot)$ is a filtering and aggregation function that can be implemented diversely. In the remainder of this paper, we primarily consider three approaches, including filtering the latest message, using the mean value of all messages [33], and an attention-based weighted summation of limited fresh messages with an AoI filter [18]. We denote them as TGN-L, TGN-M, and TGN-A, respectively.

Subsequently, in order to acquire a much more representative temporal feature, a memory updater is adopted to update the long-term preference $\mathbf{Mem}_j$ based on the compressed short-term preference $\boldsymbol{h}_j(\hat{T})$. In order to realize the updater, a learnable function, such as LSTM or the gated recurrent unit (GRU), is necessary. Here, considering the advantage in convergence speed [43], we complete the update procedure with a GRU, which is mathematically formulated as

$$\begin{aligned}
&\mathbf{Mem}_j \leftarrow \boldsymbol{Z} \cdot \boldsymbol{H} + (1 - \boldsymbol{Z}) \cdot \mathbf{Mem}_j, \\
&\boldsymbol{Z} = \sigma\left(\boldsymbol{h}_j(\hat{T})\boldsymbol{W}_{hZ} + \mathbf{Mem}_j \boldsymbol{W}_{MZ} + \boldsymbol{b}_Z\right), \\
&\boldsymbol{H} = \tanh\left(\boldsymbol{h}_j(\hat{T})\boldsymbol{W}_{hH} + \left(\boldsymbol{F} \cdot \mathbf{Mem}_j\right)\boldsymbol{W}_{MH} + \boldsymbol{b}_H\right), \\
&\boldsymbol{F} = \sigma\left(\boldsymbol{h}_j(\hat{T})\boldsymbol{W}_{hF} + \mathbf{Mem}_j \boldsymbol{W}_{MF} + \boldsymbol{b}_F\right),
\end{aligned} \tag{7}$$

where $\boldsymbol{W}_{hZ}$, $\boldsymbol{W}_{hF}$, $\boldsymbol{W}_{hH}$, $\boldsymbol{W}_{MZ}$, $\boldsymbol{W}_{MF}$ and $\boldsymbol{W}_{MH}$ denote the trainable weights, while $\boldsymbol{b}_Z$, $\boldsymbol{b}_F$ and $\boldsymbol{b}_H$ are the learnable bias values of the GRU. $\sigma(\cdot)$ and $\mathrm{tanh}(\cdot)$ are the activation functions.

2) Structural Learning Module: The structural learning module aims to generate embeddings for future prediction. In particular, it is also responsible for keeping the representations of the inactive users up-to-date by exchanging features among neighbors in the graph. Obviously, the timestamp of each interaction also plays a vital role in the mapping procedure. Therefore, we adopt a TGAT model [19] to accomplish this unconventional structural learning. Notably, the TGAT mechanism is a module that deploys a learnable time encoding function on the basis of a classical GAT module [32]. In particular, the specially designed time encoding function is formulated as

$$\Phi_{d_T}(\Delta_t) = \sqrt{\frac{1}{d_T}}[\cos(\omega_1 \Delta_t), \ldots, \cos(\omega_{d_T} \Delta_t)]^{\intercal}, \tag{8}$$

where $\omega_1, \omega_2, \ldots \omega_{d_T}$ are the trainable parameters, the superscript $\intercal$ indicates the transpose operator, Δ_t denotes the time slot between the interaction-occurring time T_n and the time to predict $\hat{T}$, (i.e., $\Delta_t = \hat{T} - T_n$). d_T is the dimension number of the desired time encoding.

Then, the encoded time features are concatenated to the output of the temporal learning module with $\mathbf{Mem}_j$ as the input for the structural learning,

$$\mathbf{Mem}'_j = [\mathbf{Mem}_j || \Phi_{d_T}(0)], \tag{9}$$

which supplements the updated long-term preference $\mathbf{Mem}'_j$ of u_j with the time feature. It is noteworthy that u_j is the center vertex of a user-centered sub-graph that we want to learn, so we define $\Delta_t = 0$ for its prediction. As for u_j's neighbor $k \in \mathcal{N}_j$, its modified preference term $\mathbf{Mem}'_k$ is formulated as

$$\mathbf{Mem}'_k = [\mathbf{Mem}_k || \Phi_{d_T}(\Delta_{t_k})], \forall k \in \mathcal{N}_j. \tag{10}$$

Notably, $\mathbf{Mem}'_j$ and $\mathbf{Mem}'_k$ are the inputs to the structural learning module. As depicted in Fig. 4, the GAT architecture [32] is the paramount part of a TGAT layer to learn the structure of u_j's dynamic sub-graph, and can be encapsulated as

$$\mathbf{E}_{u_j}(\hat{T}) = \mathrm{GAT}(\mathbf{Mem}'_j, \mathbf{Mem}'_{\mathcal{N}_j}). \tag{11}$$

Similarly, we can generate the embedding $\mathbf{E}_{i_k}(\hat{T})$ from the content-centered sub-graph of i_k with

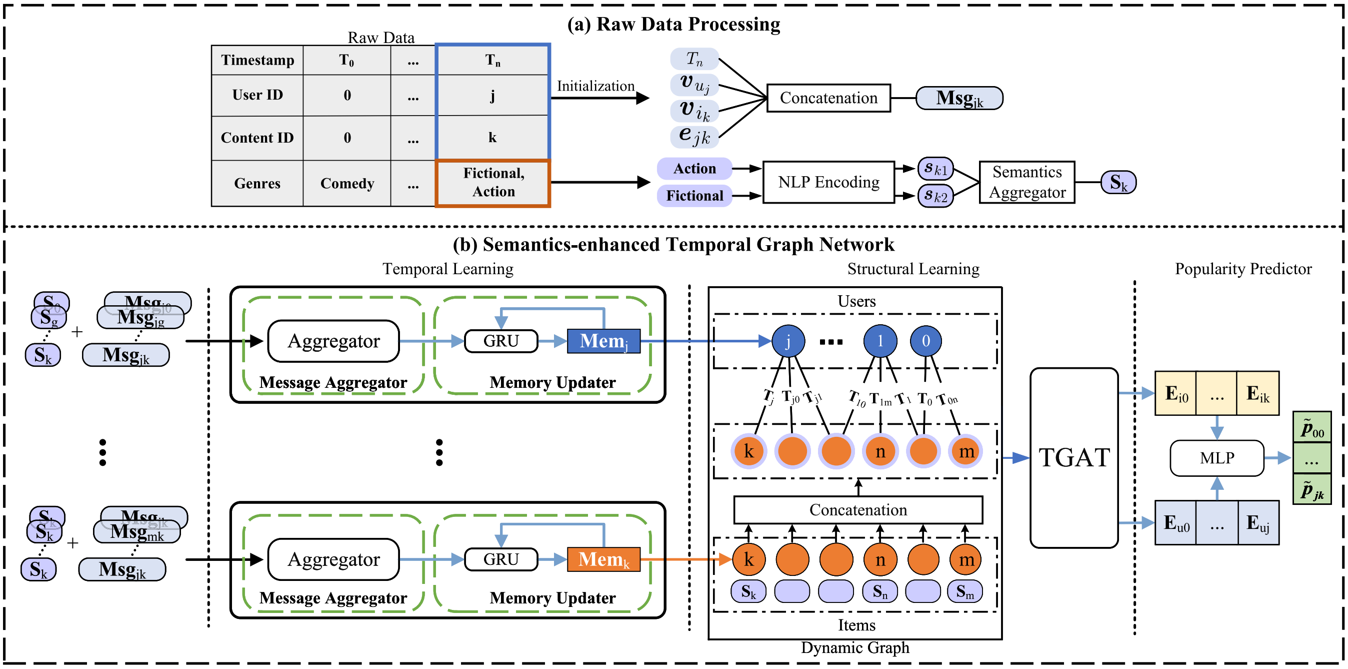
$$\begin{aligned}
&\mathbf{Mem}'_k = [\mathbf{Mem}_k || \Phi_{d_T}(0)], \\
&\mathbf{Mem}'_j = [\mathbf{Mem}_j || \Phi_{d_T}(\Delta_{t_j})], \forall j \in \mathcal{N}_k, \\
&\mathbf{E}_{i_k}(\hat{T}) = \mathrm{GAT}(\mathbf{Mem}'_k, \mathbf{Mem}'_{\mathcal{N}_k}).
\end{aligned} \tag{12}$$

Note that $\mathbf{E}_{u_j}(\hat{T})$ and $\mathbf{E}_{i_k}(\hat{T})$ are utilized as the inputs to the prediction module in (4). Furthermore, the stacking of multiple TGAT layers can leverage more hidden information within the graph by aggregating multi-hop neighbors. But the enlargement of receptive field also implies greater computational complexity [19]. Thus, we only investigate the performance with a one-layer TGAT to speed up the training in our simulations.

B. Semantic Enhancement for TGN

Essentially, the temporal learning module in TGN can be deemed as a procedure for refining the commonality from the temporal perspective. However, the randomly initialized raw features make it complicated to accurately extract and analyze the patterns, especially for a sparse dataset. Consequently, we resort to supplementing the raw input with semantic information, so as to improve the abilities of reasoning and interpretability of our model by extracting the implicit semantic correlations among contents.

We use some pre-trained NLP models, such as one-hot, BERT [22], and Glove [23], to encode the content genre information as semantic messages, $\mathcal{S}_k = \{\boldsymbol{s}_{k1}, \ldots, \boldsymbol{s}_{kN_k}\}$. For the sake of simplicity, we adopt the summation as a semantic aggregator



Timestamp	T$_0$	...	T$_n$
User ID	0	...	j
Content ID	0	...	k
Genres	Comedy	...	Fictional, Action

Fig. 4. Illustration of `M2-STGN`, a TGN model enhanced with semantics in both temporal and structural learning.

to generate the aggregated feature $\boldsymbol{S}_k$ from $\mathcal{S}_k$, which is then incorporated into the raw message as shown in Fig. 4(b). In specific,

$$\boldsymbol{S}_k = \sum_{n\in N_k} \sigma(\boldsymbol{W}_s \boldsymbol{s}_{kn} + \boldsymbol{b}_s), \tag{13}$$

$$\mathbf{Msg}'_{jk} = \sigma(\boldsymbol{W}_1^t \mathbf{Msg}_{jk} + \boldsymbol{W}_2^t \boldsymbol{S}_k), \tag{14}$$

where $\boldsymbol{W}_s$, $\boldsymbol{b}_s$, $\boldsymbol{W}_1^t$ and $\boldsymbol{W}_2^t$ are the trainable parameters to enhance the semantic features, while $\mathbf{Msg}'_{jk}$ is the desired semantics-enhanced historical message in (6). As $\mathbf{Msg}'_{jk}$ can be directly applied to enhance the temporal learning by replacing $\mathbf{Msg}_{jk}$ in (6), we regard such an approach as the semantics-enhanced TGN in a temporal manner, and denote it as `M1-STGN`.

As we discussed above, although the fresh features for inactive users can be located with the help of the graph structure, the performance still suffers from data sparsity. To address this issue, we further attach the semantic features to the input of the structural learning module, establishing implicit semantic pathways for the dynamic graph from the semantic sphere. In our experiments, we also discover that concatenation outperforms summation for merging semantics in the structural learning module. Then, (10) is further modified as

$$\mathbf{Mem}'_k = [\mathbf{Mem}_k || \boldsymbol{S}_k || \Phi_{d_T}(\Delta_{t_k})], \forall k \in \mathcal{N}_j, \tag{15}$$

where $\boldsymbol{S}_k$ is calculated following (13). Similarly, we use `M2-STGN` to represent the TGN model that is further facilitated by the structural learning with semantics.

V. Effective Semantics-Enhanced Temporal Graph Network

Although semantic aggregation can be easily achieved with the aforementioned frameworks, their utilization of semantics is still coarse. Specifically, the summation semantic aggregator doesn't distinguish the impact of different semantics from the same content on different users, while the concatenation in (15) may be oversimplified to compute proper attention coefficients. Thus, we propose two novel methods to utilize the semantics efficiently.

A. *User-Specific Attention Mechanism for Semantic Aggregation*

In order to aggregate multiple semantics fine-grainedly, we can modify the semantic aggregator with an attention mechanism that calculates attention coefficients by analyzing the influence of different genres on the same content. However, it ignores the impacts from users, which are also critical for popularity prediction. Thus, we adopt a `UsAttn` mechanism to aggregate the multiple semantics, as shown in Fig. 5. For different users, this mechanism enables the computation of different attention scores for the diverse semantics of the same content and then generates user-specific semantic features. Mathematically, for each content and user, (13) is reformulated as a linear weighted summation of N_k semantics of content i_k. The weights are calculated by the attention mechanism,

$$\boldsymbol{S}_{jk} = \sigma\left(\sum_{n\in N_k} \alpha_{jn} \boldsymbol{s}_{kn} \boldsymbol{W}_{Vn}\right),$$

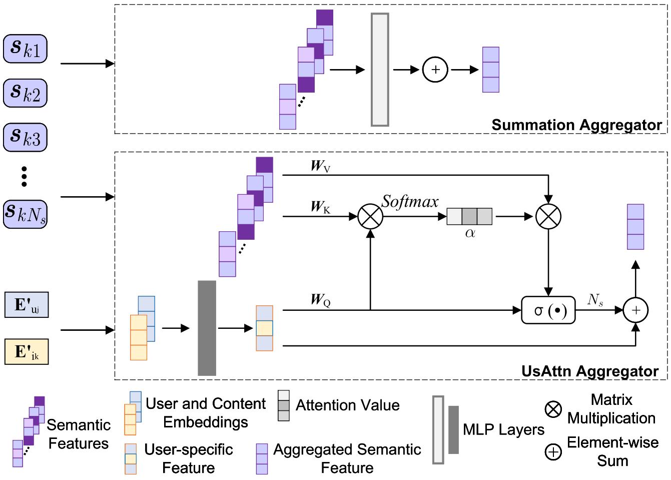


Fig. 5. Illustration of the semantics aggregators(Upper: Summation Aggregator; Lower: `UsAttn` Aggregator).

$$\alpha_{jn} = \frac{\exp(\mathbf{E}_{jk}\boldsymbol{W}_Q \cdot \boldsymbol{s}_{kn}\boldsymbol{W}_{Kn})}{\sum_{m=1}^{N_k}\exp(\mathbf{E}_{jk}\boldsymbol{W}_Q \cdot \boldsymbol{s}_{km}\boldsymbol{W}_{Km})}, \tag{16}$$

where $\boldsymbol{W}_{Kn}$, $\boldsymbol{W}_Q$ and $\boldsymbol{W}_{Vn}$ are the trainable parameters, and α_{jn} is the attention coefficient of the n-th semantic message of the content. Especially, $\mathbf{E}_{jk}$ is a user-specific embedding, after which the weight calculation will be forced to account for the embeddings of both u_j and i_k. Accordingly, we define

$$\mathbf{E}_{jk} = \text{LeakyReLu}(\boldsymbol{W}_u\mathbf{E}'_{u_j} + \boldsymbol{W}_i\mathbf{E}'_{i_k} + \boldsymbol{b}_{ui}), \tag{17}$$

where $\boldsymbol{W}_u$, $\boldsymbol{W}_i$ and $\boldsymbol{b}_{ui}$ are the trainable parameters, while $\boldsymbol{E}'_{u_j}$ and $\mathbf{E}'_{i_k}$[3] are the results generated in the last prediction or the initialization values for the first round prediction of u_j and i_k, respectively.

Moreover, the stacking of multiple DNN layers possibly results in the over-smoothing issue. In this regard, we further leverage the skip-connection in Transformer [24] to avoid this issue and improve the overall performance. Specifically, for each piece of historical message that happened between u_j and i_k, the aggregated semantics is denoted as

$$\boldsymbol{S}_k = N_k \cdot \boldsymbol{S}_{jk} + \boldsymbol{E}_{jk}, \tag{18}$$

which is the desired representation that we use in (14), so as to further optimize `M1-STGN` or the temporal learning module of `M2-STGN`.

B. Semantic Positional Encoding for Structural Learning

For different user-content pairs, the proposed `UsAttn` mechanism allocates different attention weights for content's diverse semantic messages in temporal learning. However, it is hard to generalize this mechanism into the structural learning module. Specifically, the structural learning for i_k is generally conducted by calculating a sub-graph centering around i_k, where all its neighbors are users. In other words, it is elusive for us to determine a specific user before enhancing the structural learning with `UsAttn`. This dilemma motivates us to find another method to improve semantics utilization in structural learning. Inspired by the positional encoding in Transformer and the expansion of position definition in TGAT, it might be feasible to treat contents' semantic features, generated by (13), as the positions in semantic sphere and then encode them with a customized PE function to strengthen the Transformer-alike structural learning module. To extract useful characteristics from the multi-dimensional semantic position $\boldsymbol{S}_k$, calculated by (13), we adopt a learnable Fourier features positional encoding function, which is derived in Appendix and can be mathematically formulated as follows,

TABLE III
VARIANTS OF OUR PROPOSED STGN MODEL, WHERE `Sum` AND `UsAttn` ARE THE SEMANTIC AGGREGATORS WHILE `SPE` IS THE EXTRA POSITIONAL ENCODING FOR THE GRAPH ATTENTION MODULE

	Temporal Learning	Structural Learning
`M1-STGN`	`Sum`	-
`M2-STGN`	`Sum`	`Sum`
`M1-STGN+U`	`UsAttn`	-
`M2-STGN+U`	`UsAttn`	`Sum`
`M2-STGN+SPE`	`Sum`	`Sum+SPE`
`M2-STGN+U+SPE`	`UsAttn`	`Sum+SPE`

$$\boldsymbol{R}_k = \frac{1}{\sqrt{D_h}}[\cos \boldsymbol{W}_p\phi_1(\boldsymbol{S}_k)||\sin \boldsymbol{W}_p\phi_1(\boldsymbol{S}_k)]^\intercal, \tag{19}$$

where $\phi_1(\cdot)$ is an MLP layer to enhance the semantic features, and D_h is the dimension of the hidden layer. Notably, the initialized $\boldsymbol{W}_p$ is drawn from a normal distribution [44]. Furthermore, we also discover that an additional feature enhancement with another MLP is beneficial to the final performance,

$$\boldsymbol{R}_k \leftarrow \boldsymbol{W}_p^2\text{GeLU}(\boldsymbol{W}_p^1\boldsymbol{R}_k), \tag{20}$$

where $\boldsymbol{W}_p^1$ and $\boldsymbol{W}_p^2$ is the trainable weights, and GeLU$(\cdot)$ is an activation function that is widely adopted in NLP tasks [39].

After the calculation with (19) and (20), we concatenate the encoded semantic positional embeddings into the input of TGAT layer, as in (15). Finally, we summarize all variants of our STGN model in Algorithm 1 and highlight their key differences in temporal and structural learning modules in Table III.

C. Complexity Analysis

On the other hand, to address the practical concern, we further conduct an analysis on the computational complexity for the major variants of our proposed models, and the results are summarized in Table IV. In the table, $M = |\mathcal{N}_j|$ or $|\mathcal{N}_k|$ is the number of the neighbors in a sub-graph when executing the attention-based structural learning, while N is the number of filtered messages shown in (6) and $N_s = |\mathcal{S}_k|$ is the number of types of semantic genre information within a content. Additionally, d, d_s and d_S are introduced to represent the dimension numbers of raw input, semantics $\boldsymbol{S}_k$ and the feature accompanied with semantics (e.g., the $\mathbf{Mem}'_k$ in (15)), respectively. Notably, we employ a multi-head attention mechanism for the corresponding calculations in (6) and (11) to improve the performance, where

[3] For simplicity, we omit the time information $\hat{T}'$ of the last prediction in $\mathbf{E}'_{u_j}(\hat{T}')$ and $\mathbf{E}'_{i_k}(\hat{T}')$.

Algorithm 1: The Preference Prediction With STGN.

```
Require: Request dataset and pre-trained NLP model.
Ensure: The representations of $u_j$ and $i_k$, (i.e., $\boldsymbol{E}_j^u$ and
    $\boldsymbol{E}_k^i$) and the preference between $u_j$ and $i_k$.
Initialize the raw data and the parameters for the whole
    network and encode contents' semantics information
    with pre-trained NLP model.
Divide the raw data into several mini batches.
for each
batch$(\boldsymbol{v}_{\mathrm{u_j}}, \boldsymbol{v}_{\mathrm{i_k}}, \boldsymbol{e}_{\mathrm{ui}}, \mathrm{t}, \mathcal{S}_{\mathrm{k}}) \in$ training dataset do
    $\dot{n} \leftarrow$ Sample negatives;
    if aggregate $\mathcal{S}_k$ by summation then
        Aggregate $\mathcal{S}_k$ to compute $\boldsymbol{S}_k$ with (13);
    else
        Obtain user-specific embedding $\mathbf{E}_{jk}$ with (17);
        Aggregate $\mathcal{S}_k$ and $\mathbf{E}_{jk}$ with (16) and (18);
    end if
    Concatenate the message $\mathbf{Msg}_{jk}$ with the
        aggregated semantics in (14);
    Filter and aggregate historical messages in (6) to
        obtain short-term preference $\boldsymbol{h}_j(\hat{T})$;
    Update long-term preference $\mathbf{Mem}_j$ with $\boldsymbol{h}_j(\hat{T})$
        by the method (7);
    Encode the time slot $\Delta_t$ with (8) for all vertexes;
    if semantic positional encoding then
        Encode the summarized semantics with (19)
            and (20);
    end if
    Incorporate the encrypted time and semantics
        features into the updated lone-term preference
        $\mathbf{Mem}_j'$;
    Obtain $\mathbf{E}_j^u(T_p)$ and $\mathbf{E}_k^i(T_p)$ through a TGAT
        module for the structural learning;
    Predict the preference between users and contents
        with (4);
    Optimize with BCELoss$(\cdot)$.
end for
```

k, $k_0 = 2$ denote their utilized head numbers. As demonstrated in [18], [19], the complexity of TGN-L, TGN-M and TGN-A are $O(k_0 Md)$, $O(Nd + k_0 Md)$ and $O(k(N^2 d + Nd^2) + k_0 Md)$, respectively. In our work, we augment the prediction performance of the conventional TGN model in a sparse dataset by incorporating extra semantic features, resulting in the increase of complexity. Specifically, an additional increment of $O(N_s d_s)$ for the semantic summation and $O(N_s^2 d_s + N_s d_s^2)$ for the user-specific attention mechanism are introduced according to the selection of semantic aggregator. Finally, with the incorporation of SPE, we further need another $O(Md_s^2)$ complexity for the semantic enhancement and positional encoding. It is noteworthy that the M2-STGN-A+U+SPE model owns the highest complexity (i.e., $O((N^2 d_s + Nd_s^2) + Md_s^2 + k(N^2 d_S + Nd_S^2) + k_0 Md_S))$, while the computational complexity for the variants of lightweight models, TGN-L and TGN-M, are still in an acceptable level.

VI. Experimental Results and Discussions

In this section, we present the performance of our proposed models in prediction and caching tasks. We also compare our methods with three state-of-the-art models in processing dynamic graphs, including TGAT [19], DyRep [29], and TGN [33]. Besides, in order to analyze the effectiveness of semantic features, we further adopt some widely-accepted NLP methods to encode the genres, i.e., one-hot, BERT [22], and Glove [23]. Moreover, experiments with respect to the cache hit rate as well as the transmission delay and hop count are also conducted to validate the superiority of our model-based caching methods.

A. Experimental Settings

Dataset: In this paper, the experiments are carried out with a public dataset, Netflix,[4] which records a set of user behaviors on Netflix in U.K.. Notably, there are many insignificant historical messages, such as some users only request once or watch the content for an extremely short period. The burst behavior is hard to be predicted accurately, and it may mislead other predictions as well. Therefore, we select those users who have more than 4 requests and view each requested content for more than 3 minutes as the valid input for the prediction. The dataset we actually adopt includes 86,889 interactions, which involve 11,254 different users and 4,057 pieces of content. The number of interactions is less than the dataset[5] used in Ref. [18], while the numbers of users and contents are larger, making the Netflix even much sparser. On the other hand, the content genres are some summarizing keywords of the target, which are eligible to encapsulate a video's universal semantic attributes. For instance, the genres of *The Amazing Spider-Man 2* in this dataset are comprised of *Action, Adventure, Sci-Fi*. In this case, the related genre dimension equals 3. Each content in the used dataset Netflix usually contains 1 to 8 diverse genres, with a combination of 28 genres in total. Afterwards, we perform a $60\% - 20\% - 20\%$ chronological split of the dataset for training, validation, and testing, respectively.

Evaluation Tasks and Training Configuration: To verify the effectiveness of our proposed models, we compare our models with the state-of-the-art models, including TGAT [19], DyRep [29], TGN [33] and its variants (i.e., TGN-L, TGN-M and TGN-A). Notably, the receptive field for the graph in GNN is proportional to the number of GNN layers l and the neighbors M in each sub-graph. As Ref. [19] suggests, we set $l = 2$ and $M = 10$ in TGAT, while $l = 1$ and $M = 10$ in TGN. Besides, we also compare our models to the TGN model with larger receptive field, i.e., $l = 1, 2$ and $N = 11, 12, 13, 14, 15, 20$.

Furthermore, as for pre-trained NLP models, Glove [23] is conducted by global word-to-word occurrence statistics from a large corpus, while BERT [22] is a neural network model based on 12-layer Transformer. Due to the various techniques to encode genre information into embedding representations, the semantic feature dimensions of input in our model are also

[4] https://www.kaggle.com/datasets/vodclickstream/netflix-audience-behaviour-uk-movies

[5] The dataset used in Ref. [18] involves 5,763 users and 56 contents, which consists of 175,856 interactions.

TABLE IV
COMPUTATIONAL COMPLEXITY FOR THE MAJOR VARIANTS OF OUR PROPOSED MODELS

Model	Complexity
M1-STGN-A	$O(N_s d_s + k(N^2 d_S + N d_S^2) + k_0 M d_S)$
M2-STGN-A	$O(2N_s d_s + k(N^2 d_S + N d_S^2) + k_0 M d_S)$
M1-STGN-A+U	$O((N_s^2 d_s + N d_s^2) + k(N^2 d_S + N d_S^2) + k_0 M d_S)$
M2-STGN-A+U	$O(N_s d_s + (N_s^2 d_s + N d_s^2) + k(N^2 d_S + N d_S^2) + k_0 M d_S)$
M2-STGN-L+U	$O(N_s d_s + (N_s^2 d_s + N_s d_s^2) + k_0 M d_S)$
M2-STGN-M+U	$O(N_s d_s + (N_s^2 d_s + N_s d_s^2) + N d_S + k_0 M d_S)$
M2-STGN-A+U+SPE	$O(N_s d_s + (N_s^2 d_s + N_s d_s^2) + M d_s^2 + k(N^2 d_S + N d_S^2) + k_0 M d_S)$
M2-STGN-L+U+SPE	$O(N_s d_s + (N_s^2 d_s + N_s d_s^2) + M d_s^2 + k_0 M d_S)$
M2-STGN-M+U+SPE	$O(N_s d_s + (N_s^2 d_s + N_s d_s^2) + M d_s^2 + N d_S + k_0 M d_S)$

different. Specifically, the raw semantic dimension for encoding with one-hot is 28, while that BERT and Glove are 738 and 50, respectively. In particular, through the subsequent graph-based calculation, the final representations with the dimension of 172 for users can be deemed as encompassing the semantics as well. Notably, to further reduce the computational cost, we also adopt an MLP to compress the representations of BERT. Besides, Ref. [45] also discovers that the outputs from the 6-th to 10-th layers outperform in semantic tasks. Therefore, we average the embeddings from the 6-th to 10-th layers to investigate the performance.

Moreover, we conduct experiments under two tasks, i.e., transductive task and inductive task. Different from the transductive task, the validation set and test set in an inductive task may contain some vertices that have not been observed by models during the training phase. For both tasks, we adopt the *average precision (AP)* and the *area under the ROC curve (AUC)* as evaluation metrics.

Caching Policy Setting: As depicted in Fig. 3, we configure a multi-layer network architecture. The storage capacity for devices in different tiers is diverse and that in the device closer to users is typically smaller [41]. Hence, we assume that the *Tier 1* can store 5 contents, while 7 and 8 contents can be cached at *Tier 2* and *Tier 3*, respectively. In other words, based on the prediction results, we can obtain the content popularity for each access node as well as an overall rating list. The access nodes in *Tier 1* cache the top-5 content according to their respective results, while the *Tier 2* and *Tier 3* deploy the subsequent 7 and 8 contents in the overall rating list that excludes the union set within *Tier 1*.

As for the content caching task, our target is to predict contents' popularity during 24 hours in the test phase. We assume that the candidate content set $\mathcal{I}$ is known apriori. To make the simulations more practical, we supplement the candidate content set with a noise set, which consists of the contents that have been requested within a 50-hour duration before the prediction starting time. Besides, the user set $\mathcal{U}$ of each hour is also assumed to be known in our simulation. As for other hyperparameters, we compute the per-hour popularity with $\Delta_P = 1\text{h}$ and $\delta_P = 60\text{s}$ while the threshold value $p_{\text{thre}} = 0.995$. Notably, our simulations are conducted with an assumption that more popular content is cached at devices in closer proximity to users (e.g., mobile edge nodes). The dynamic bipartite graph is constituted according to the recorded timestamps T_n, user index u_j and content index i_k, which can be easily recorded by edge devices. Therefore, the latency has trivial impact, especially given the latency to collect data is far smaller than the prediction interval δ_p.

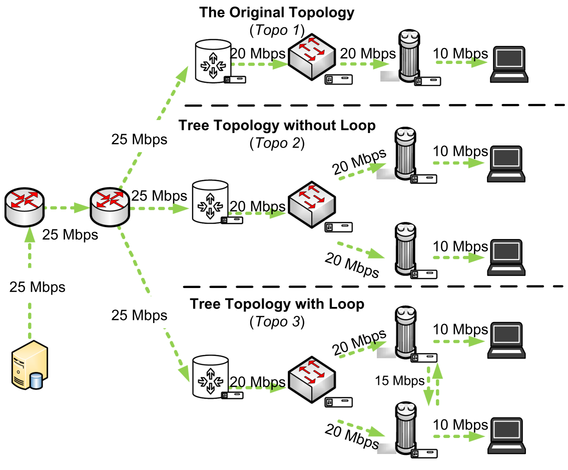


Fig. 6. Network with different topologies and their transmission rate.

To verify the superiority of our models in content caching, a comparison between the traditional caching method, LRU, and the prediction results based strategy is also carried out. We deploy TGN-A and its variants, i.e., M2-STGN-A and M2-STGN-A+U, as the predictive models. Due to the tradeoff between training speed and prediction performance for M2-STGN-L+U and M2-STGN-L+U+SPE, simulations based on them are executed as well. Moreover, testing in the inductive setting, we also conduct extensive ablation studies with M2-STGN-A+U, whose performance in preference prediction is the most superior, so as to examine the influence of different hyperparameters (i.e., different sizes of content supplement set and values of δ_p & p_{thre}) on content caching. Notably, we adjust the size of the content supplement set by changing the duration before the starting time.

Apart from the cache hit rate, to further validate the superiority of our model from a QoS perspective, we also measure the delay of downloading the first 1 MB packet and the transmission hops with the assumed topologies illustrated in Fig. 6. The comparison is mainly conducted between LRU, and the strategies with the best-performing model (i.e., M2-STGN-A+U). Moreover, although *Topology 1* in Fig. 6 (i.e., the topology in Fig. 3) is the topology that we mainly use within the above simulations, we also verify the performance of deploying our model in networks characterized by other diverse topologies. Particularly, compared to *Topology 2*, an extra connection between two access

TABLE V
TRAINING TIME AND THE PERFORMANCE OF PREDICTING CONTENT REQUESTS IN BOTH TRANSDUCTIVE AND INDUCTIVE TASKS

Metric		*AUC for Transductive*	*AP for Transductive*	*AUC for Inductive*	*AP for Inductive*	*Training Time*
Baseline	TGAT	75.891	73.426	66.549	65.955	43.711s
	DyRep	84.027	84.096	76.162	77.562	20.116s
	TGN-L	85.299	83.824	77.285	76.995	14.602s
	TGN-M	86.731	86.022	78.953	79.721	90.729s
	TGN-A	90.507	90.691	83.504	84.999	158.163s
M1-STGN	M1-STGN-L	86.386	85.892	79.980	80.439	16.291s
	M1-STGN-M	88.312	88.095	82.043	82.765	91.855s
	M1-STGN-A	91.210	91.337	85.247	86.175	159.239s
M2-STGN	M2-STGN-L	87.383	86.806	81.131	81.182	17.917s
	M2-STGN-M	88.649	88.558	82.805	83.461	91.309s
	M2-STGN-A	$\boxed{91.773}$	$\boxed{91.953}$	$\boxed{85.877}$	$\boxed{87.019}$	158.754s
M1-STGN+U	M1-STGN-L+U	89.014	88.467	83.096	83.483	16.510s
	M1-STGN-M+U	89.721	89.356	83.585	84.148	92.638s
	M1-STGN-A+U	91.358	91.572	85.327	86.567	158.077s
M2-STGN+U	M2-STGN-L+U	89.749	89.279	84.183	84.387	18.173s
	M2-STGN-M+U	90.107	89.884	84.434	84.868	91.655s
	M2-STGN-A+U	**91.846**	**92.056**	**86.264**	**87.279**	160.544s
M2-STGN+U+SPE	M2-STGN-L+U+SPE	90.778	90.406	84.582	85.211	19.157s
	M2-STGN-M+U+SPE	90.951	90.641	84.542	85.181	92.235s
	M2-STGN-A+U+SPE	91.680	91.824	85.784	86.793	163.363s

TGAT and DyRep are two state-of-the-art DGNN models. TGN-L, TGN-M, and TGN-A are the conventional TGN model's variants with different message aggregators. The best results are highlighted in bold and the second-best results are highlighted in box.

TABLE VI
TEST AVERAGE PRECISION RESULTS OF M2-STGN+SPE WITH DIFFERENT NLP METHODS FOR ENCODING THE GENRE INFORMATION

Semantic model		Transductive	Inductive
One-hot	M2-STGN-L+SPE	87.827	81.581
	M2-STGN-M+SPE	89.730	84.002
	M2-STGN-A+SPE	90.908	85.322
BERT	M2-STGN-L+SPE	87.998	82.300
	M2-STGN-M+SPE	88.972	83.598
	M2-STGN-A+SPE	91.287	86.427
Glove	M2-STGN-L+SPE	88.397	82.447
	M2-STGN-M+SPE	89.392	83.803
	M2-STGN-A+SPE	91.805	86.800

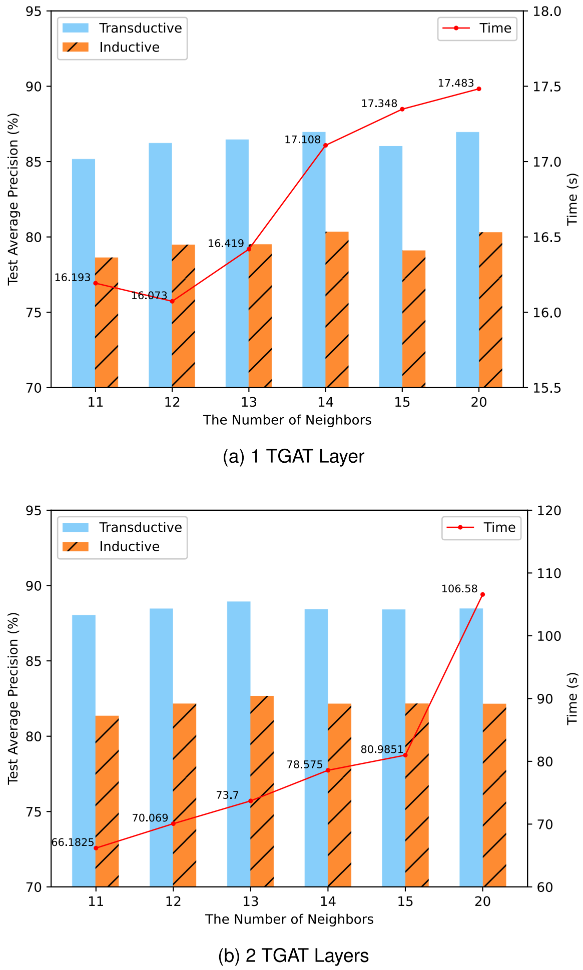
nodes is constructed in *Topology 3*, which constitutes a simple cooperation for the content caching. Besides, in *Topology 2* and *3*, we randomly allocate the users into two access nodes and make prediction with the best predictive model for each community.

B. Results Analysis

Table V demonstrates the prediction performance of our proposed TGN models as well as the baseline models. It can be clearly observed that our models are able to yield better results in both transductive task and inductive task, and even the primitive models in M1-STGN outperform the counterparts of TGN. Moreover, the superiority of M1-STGN+U and M2-STGN+U also proves the effectiveness of the UsAttn semantic aggregator. In addition, due to the introduction of SPE, we can also find that the prediction capabilities of most models have been enhanced, especially for the variants of the lightweight models (i.e., TGN-L and TGN-M).

Fig. 7 presents the prediction performance of TGN-L with different sizes of receptive field. As the receptive field enlarges, the performance of TGN-L in both transductive and inductive tasks improves. However, the average training time for obtaining a single model is gradually increasing as well. Compared with the corresponding results in Table V and Fig. 7, we can also discover that most variant models of TGN-L, which try to have a deeper insight into the available information through either UsAttn or SPE, are superior than methods that enlarge the receptive field (e.g., the stacking of GNN layers or an increase in the number of first-order neighbors) in both prediction performance and training speed. In order to achieve a comparable result to the M2-STGN-L+U+SPE, it takes TGN-L with 2 TGAT layers at least 4× more time. Obviously, our "breadth-first approach" for excavating the inherent relationships is more efficient.

Together with the tradeoff between training latency and accuracy shown in Fig. 8 and the computational complexity demonstrated in Table IV, the excellent-performing models, M2-STGN-A as well as its variants M2-STGN-A+U and M2-STGN-A+U+SPE, achieve competitive accuracy (and possibly reach a performance plateau even when we adopt *Early Stopping* strategy to alleviate the overfitting issue) at the cost of a higher expenditure of computation and training time, which makes them primarily suitable for the accuracy-centric scenarios. It should be noted here that the application of SPE successfully narrows the performance gaps between the fast-trained lightweight models (i.e., M2-STGN-L+U and M2-STGN-M+U) and the best-performing model (i.e., M2-STGN-A+U) while maintaining the training latency in an acceptable level, as shown in Fig. 8. Given the balance of computational complexity and accuracy, the counterpart lightweight models equipped with SPE emerge as a feasible choice for access nodes with limited computing resources.



(a) 1 TGAT Layer

(b) 2 TGAT Layers

Fig. 7. Test average precision (the bars) and training time (the lines) of TGN-L with different numbers of aggregating neighbors and TGAT layers.

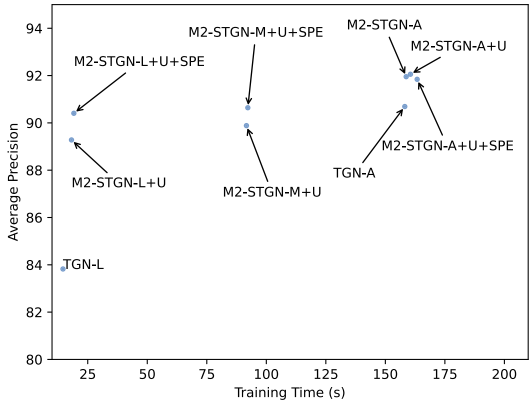


Fig. 8. Tradeoff between accuracy (Average Precision in %) and latency (Training Time per epoch in seconds) of different training strategies.

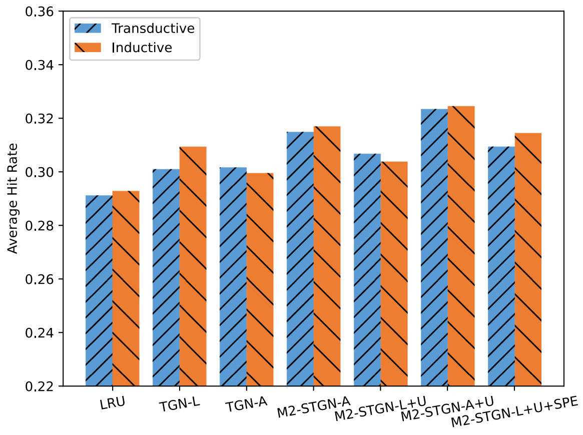
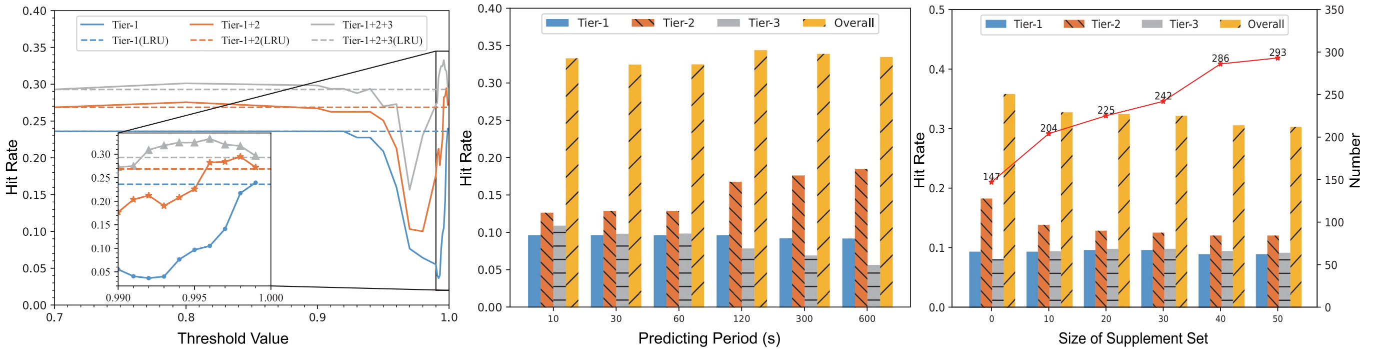


Fig. 9. Average hit rate performance of different algorithms in 24 hours.

Fig. 9 compares the average hit rate of the caching strategy based on prediction results and LRU for the whole network architecture in both transductive and inductive tasks during 24 hours. It can be observed that relying on our proposed models, the overall caching performance of the prediction-based strategy is always better than LRU. The improvement of prediction accuracy increases the cache hit rate as well. In particular, the caching strategy based on M2-STGN-A+U surpasses other models, which is in line with the prediction performance. Moreover, even considering the final performance in caching, the lightweight model, M2-STGN-L+U+SPE, is still a promising choice for the resource-limited access node. Actually, we also conduct simulations for the other two baselines, i.e., TGAT and DyRep, but their poor performance results in too many false positive predictions, failing to distinguish the popularity of contents.

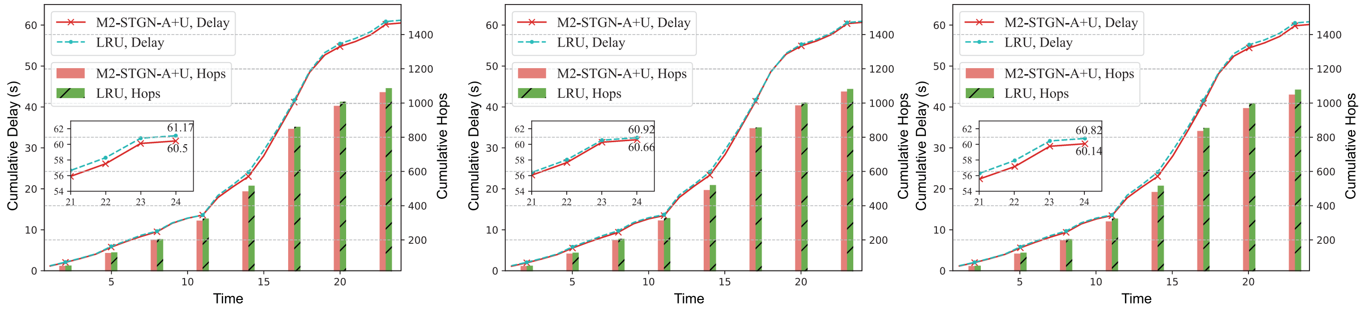
Fig. 10 reveals the hit rate performance in the inductive setting with different hyperparameters for caching. It can be observed in Fig. 10(a) that the caching performance with $p_{\mathtt{thre}} < 0.8$ is equivalent to LRU. Since the caching strategy, introduced in Section III-B, decides the prioritization for contents with the same predicted popularity consistent with LRU, such an abnormal phenomenon implies that our model fails to distinguish the popularity of contents when adopting an improper threshold value. However, for a larger threshold, the caching gain from our model becomes more evident, especially when the threshold is close to 1, the strategy relying on our model outperforms the traditional LRU at all tiers. Surprisingly, a more frequent prediction operation does not always lead to an improvement in the hit rate. The simulation results in Fig. 10(b) present that when $\delta_p = 120$s, our model brings the greatest gain to the cache task. On the other hand, Fig. 10(c) shows that the size of candidate content also affects the final caching results. As the number of candidate content gradually increases, it gives rise to a declined overall hit rate, but is still superior to LRU. To sum up, these experiments demonstrate the robustness of the proposed methods for caching in cases with large number of inactive users.

Fig. 11 shows the cumulative transmission delay and hop count within 24 hours. It can be observed that caching based on our M2-STGN-A+U model consistently outperforms the strategy with the traditional LRU rule. Compared with caching in *Topology 2*, the performance improvement observed in *Topology 3* stems from the incorporation of the connection between two edge nodes, facilitating a simple collaboration and compensating



(a) Different prediction threshold values. (b) Different popularity predicting periods. (c) Different numbers of candidate content.

Fig. 10. Hit rate performance in the inductive setting with different hyperparameters for caching.



(a) Topology 1 (b) Topology 2 (c) Topology 3

Fig. 11. Cumulative transmission delay and hop count within 24 hours under 3 topologies.

for the lack of cooperation in making caching decision. Consequently, exploring a smarter caching decision strategy for a complicated network topology is a promising direction for our future research.

VII. Conclusions and Future Works

In this paper, we have developed an STGN architecture to improve the performance of popularity prediction in a sparse dataset. We have taken into account the genres of the contents as semantic information and profiled users' intentions by attaching the semantics into the conventional TGN. The proposed STGN models have significantly ameliorated the user preference speculation performance. Furthermore, we have devised a `UsAttn` mechanism for a finer-grained semantic aggregation of diverse genres related to the same content. Meanwhile, an `SPE` function, targeting at assisting the association analysis in the attention-based graph learning, has been adopted as well. Due to the superior prediction performance, the caching strategy based on our STGN model also wins a great improvement in cache hit rate and other metrics from the QoS perspective under extensive simulations.

Our model also provides a paradigm for the fusion of semantics and AI models. Specifically, `UsAttn` suggests a novel method to aggregate multiple semantic information fine-grainedly. Moreover, the improvement with the maintained computational efficiency manifested in the lightweight model (e.g., the `M2-STGN-L+U+SPE`) implies that `SPE` is a feasible answer on how to efficiently incorporate and utilize the aggregated semantic information with AI models.

Finally, apart from utilizing the additional side information, a video embedding obtained through an aggregation of its spatial and temporal features, which embodies the semantics substantially [46], is also viable. Although training a mature video semantic extractor that can harness the inter-correlations among videos is a great challenge, given its fundamental roles in semantic video communications [47], it is also a promising research that warrants future exploration. Meanwhile, besides the application in caching, we also believe that it has the potential to be generalized to other network architectures that desire AI models to be integrated with semantic analysis, like intent-based network (IBN) [48].

References

[1] J. Zhu, R. Li, X. Chen, S. Mao, J. Wu, and Z. Zhao, "Semantics-enhanced temporal graph networks for content caching and energy saving," in *Proc. IEEE Int. Conf. Commun.*, Roma, Italy, 2023, pp. 1724–1729.

[2] "Cisco annual Internet report (2018–2023) white paper," Cisco, 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11--741490.pdf

[3] Z. Hajiakhondi Meybodi, A. Mohammadi, E. Rahimian, S. Heidarian, J. Abouei, and K. N. Plataniotis, "TEDGE-Caching: Transformer-based edge caching towards 6G networks," in *Proc. IEEE Int. Conf. Commun.*, Seoul, South Korea, 2022, pp. 613–618.

[4] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge AI: Algorithms and systems," *IEEE Commun. Surv. Tutor.*, vol. 22, no. 4, pp. 2167–2191, Fourth Quarter, 2020.

[5] "Mobile edge computing (MEC); technical requirements," ETSI, 2016. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/mec/001_099/002/01.01.01_60/gs_mec002v010101p.pdf

[6] C. Yang, Y. Yao, Z. Chen, and B. Xia, "Analysis on cache-enabled wireless heterogeneous networks," *IEEE Trans. Wirel. Commun.*, vol. 15, no. 1, pp. 131–145, Jan. 2016.

[7] Q. Chen, W. Wang, W. Chen, F. R. Yu, and Z. Zhang, "Cache-enabled multicast content pushing with structured deep learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2135–2149, Jul. 2021.

[8] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6G: Vision, enabling technologies, and applications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 5–36, Jan. 2022.

[9] D. Xu et al., "Edge intelligence: Empowering intelligence to the edge of network," in *Proc. IEEE*, vol. 109, no. 11, pp. 1778–1837, Nov. 2021.

[10] O. Serhane, K. Yahyaoui, B. Nour, and H. Moungla, "A survey of ICN content naming and in-network caching in 5G and beyond networks," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4081–4104, Mar. 2021.

[11] W. Liu, J. Zhang, Z. Liang, L. Peng, and J. Cai, "Content popularity prediction and caching for ICN: A deep learning approach with SDN," *IEEE Access*, vol. 6, pp. 5075–5089, 2018.

[12] Z. Zhang and M. Tao, "Deep learning for wireless coded caching with unknown and time-variant content popularity," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 2, pp. 1152–1163, Feb. 2021.

[13] T. Zong, C. Li, Y. Lei, G. Li, H. Cao, and Y. Liu, "Cocktail edge caching: Ride dynamic trends of content popularity with ensemble learning," *IEEE/ACM Trans. Netw.*, vol. 31, no. 1, pp. 208–219, Feb. 2023.

[14] J. Zhou et al., "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[15] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, 2022.

[16] J. Skarding, B. Gabrys, and K. Musial, "Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey," *IEEE Access*, vol. 9, pp. 79143–79168, 2021.

[17] Y. Fu, L. Salaün, X. Yang, W. Wen, and T. Q. S. Quek, "Caching efficiency maximization for device-to-device communication networks: A recommend to cache approach," *IEEE Trans. Wirel. Commun.*, vol. 20, no. 10, pp. 6580–6594, Oct. 2021.

[18] J. Zhu et al., "AoI-based temporal attention graph neural network for popularity prediction and content caching," *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 2, pp. 345–358, Apr. 2023.

[19] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–19.

[20] J. Liang et al., "Multi-head attention based popularity prediction caching in social content-centric networking with mobile edge computing," *IEEE Commun. Lett.*, vol. 25, no. 2, pp. 508–512, Feb. 2021.

[21] Y. Liu, S. Yang, Y. Xu, C. Miao, M. Wu, and J. Zhang, "Contextualized graph attention network for recommendation with item knowledge graph," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 181–195, Jan. 2023.

[22] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics - Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[23] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Doha, Qatar, 2014, pp. 1532–1543.

[24] A. Vaswani et al., "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017.

[25] T. Zhang, X. Fang, Z. Wang, Y. Liu, and A. Nallanathan, "Stochastic game based cooperative alternating q-learning caching in dynamic D2D networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13255–13269, Dec. 2021.

[26] X. Xu, C. Feng, S. Shan, T. Zhang, and J. Loo, "Proactive edge caching in content-centric networks with massive dynamic content requests," *IEEE Access*, vol. 8, pp. 59906–59921, 2020.

[27] M. Amadeo, C. Campolo, G. Ruggeri, and A. Molinaro, "Beyond edge caching: Freshness and popularity aware IoT data caching via NDN at internet-scale," *IEEE Trans. Green Commun. Netw.*, vol. 6, no. 1, pp. 352–364, Mar. 2022.

[28] D. Lee et al., "LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies," *IEEE Trans. Comput.*, vol. 50, no. 12, pp. 1352–1361, Dec. 2001.

[29] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "DyRep: Learning representations over dynamic graphs," in *Proc. Int. Conf. Learn. Representations*, New Orleans, LA, USA, 2019, pp. 1–25.

[30] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[31] Y. Zhang, H. An, Y. Xing, Y. Liu, and T. Zhang, "Learning temporal and spatial features jointly: A unified framework for space-time data prediction in industrial IoT networks," *IEEE Sens. J.*, vol. 23, no. 16, pp. 18752–18764, Aug. 2023.

[32] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018, pp. 1–12.

[33] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, "Temporal graph networks for deep learning on dynamic graphs," in *Proc. Int. Conf. Mach. Learn. Workshop Graph Representation Learn.*, Virtual Edition, 2020, pp. 1–9.

[34] H. Wang et al., "RippleNet: Propagating user preferences on the knowledge graph for recommender systems," in *Proc. Conf. Inf. Knowl. Manage.*, Torino, Italy, 2018, pp. 417–426.

[35] Y. Li et al., "Learning dynamic user interest sequence in knowledge graphs for click-through rate prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 647–657, Jan. 2023.

[36] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, Honolulu, Hawaii, USA, 2019, pp. 5329–5336.

[37] H. Mezni, D. Benslimane, and L. Bellatreche, "Context-aware service recommendation based on knowledge graph embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5225–5238, Nov. 2022.

[38] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–12.

[39] Y. Li, S. Si, G. Li, C.-J. Hsieh, and S. Bengio, "Learnable fourier features for multi-dimensional spatial positional encoding," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 15816–15829.

[40] N. J. Parmar et al., "Image transformer," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4052–4061.

[41] O. Ayoub, F. Musumeci, M. Tornatore, and A. Pattavina, "Energy-efficient video-on-demand content caching and distribution in metro area networks," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 1, pp. 159–169, Mar. 2019.

[42] B. Chen and C. Yang, "Caching policy for cache-enabled d2d communications by learning user preference," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6586–6601, Dec. 2018.

[43] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2014, pp. 1–9.

[44] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2007, pp. 1177–1184.

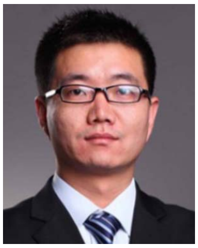
[45] G. Jawahar, B. Sagot, and D. Seddah, "What does BERT learn about the structure of language?" in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 3651–3657.

[46] S. Tong, X. Yu, R. Li, K. Lu, Z. Zhao, and H. Zhang, "Alternate learning based sparse semantic communications for visual transmission," in *Proc. Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, Toronto, ON, Canada, 2023, pp. 1–6.

[47] S. Wang et al., "Wireless deep video semantic transmission," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 214–229, Jan. 2023.

[48] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *IEEE Commun. Surv. Tuts.*, vol. 25, no. 1, pp. 625–655, First Quarter, 2023.

Jianhang Zhu (Graduate Student Member, IEEE) received the BS degree in communication engineering from Jilin University, Changchun, China, in 2020. He is currently working toward the EngD degree with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou. His research interest includes graph neural network, multi-agent reinforcement learning, and edge computing.



Rongpeng Li (Member, IEEE) is currently an associate professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He was a research engineer with the Wireless Communication Laboratory, Huawei Technologies Company, Ltd., Shanghai, China, from 2015 to 2016. He was a visiting scholar with the Department of Computer Science and Technology, University of Cambridge, Cambridge, U.K., from February to August 2020. His research interest currently focuses on networked intelligence for communications evolving (NICE). He received the Wu Wenjun Artificial Intelligence Excellent Youth Award in 2021. He serves as an editor for China Communications.

Xianfu Chen (Member, IEEE) received the PhD degree (with Hons.) from Zhejiang University, Hangzhou, China, in 2012. Since 2012, he has been with the VTT Technical Research Centre of Finland, Oulu, Finland, where he is currently a senior scientist. His research interests include wireless communications and networking, with emphasis on human-level and artificial intelligence for resource awareness in next-generation communication networks. He was the recipient of the 2021 IEEE Communications Society Outstanding Paper Award and the 2021 *IEEE Internet of Things Journal* Best Paper Award. He is the editor of *IEEE Transactions on Cognitive Communications and Networking and Microwave and Wireless Communications*, an academic editor for *Wireless Communications and Mobile Computing*, an associate editor for China Communications, and was a member of the First Editorial Board of Journal of Communications and Information Networks. He was the guest editor for several international journals, including *IEEE Wireless Communications Magazine*. He was the TPC Chair/Track co-chair/TPC member for a number of IEEE ComSoc flagship conferences.

Shiwen Mao (Fellow, IEEE) received the PhD degree in electrical engineering from Polytechnic University, Brooklyn, NY, in 2004. After joining Auburn University, Auburn, AL in 2006, he held the McWane Endowed Professorship from 2012 to 2015 and the Samuel Ginn Endowed Professorship from 2015 to 2020 in the Department of Electrical and Computer Engineering. Currently, he is a professor and Earle C. Williams Eminent Scholar Chair, and director of the Wireless Engineering Research and Education Center with Auburn University. His research interest includes wireless networks, multimedia communications, and smart grid. He is a distinguished lecturer of *IEEE Communications Society* and the IEEE Council of RFID. He is the editor-in-chief of *IEEE Transactions on Cognitive Communications and Networking* and an area editor of ACM GetMobile. He is the General chair of IEEE INFOCOM 2022, a TPC Chair of IEEE INFOCOM 2018, the TPC vice-chair of IEEE INFOCOM 2015, and the TPC vice chair of IEEE GLOBECOM 2022. He received the SEC (Southeastern Conference) 2023 Faculty Achievement Award for Auburn, the IEEE ComSoc TC-CSR Distinguished Technical Achievement Award in 2019, the Auburn University Creative Research & Scholarship Award in 2018, the NSF CAREER Award in 2010, and several service awards from IEEE ComSoc. He is a co-recipient of the 2022 Best Journal Paper Award of IEEE ComSoc eHealth Technical Committee (TC), the 2021 Best Paper Award of Elsevier/KeAi Digital Communications and Networks Journal, the 2021 IEEE Internet of Things Journal Best Paper Award, the 2021 IEEE Communications Society Outstanding Paper Award, the IEEE Vehicular Technology Society 2020 Jack Neubauer Memorial Award, the 2018 Best Journal Paper Award and the 2017 Best Conference Paper Award from IEEE ComSoc Multimedia TC, and the 2004 IEEE Communications Society Leonard G. Abraham Prize in the Field of Communications Systems. He is a co-recipient of the Best Paper Awards from IEEE ICC 2022 and 2013, IEEE GLOBECOM 2023, 2019, 2016, and 2015, and IEEE WCNC 2015, and the Best Demo Awards from IEEE INFOCOM 2022 and IEEE SECON 2017.

Jianjun Wu the chief researcher and director of Future Network Laboratory of Huawei Technologies Company Ltd., the main research direction is future wireless network architecture include 6G network architecture definition, 5G E2E slicing solution research, standards, and industry development. He was the director of the European Research Center Branch of Huawei 2012 Laboratories and led the local team to fully participate in the definition and research of 5G origins such as 5GIA and 5GPPP. He initiated and successfully established the 5GAA and 5GACIA industry alliances.

Zhifeng Zhao (Member, IEEE) received the BE degree in computer science, the ME degree in communication and information systems, and the PhD degree in communication and information systems from the PLA University of Science and Technology, Nanjing, China, in 1996, 1999, and 2002, respectively. From 2002 to 2004, he acted as a postdoctoral researcher with Zhejiang University, Hangzhou, China, where his researches were focused on multimedia next-generation networks (NGNs) and soft switch technology for energy efficiency. From 2005 to 2006, he acted as a senior researcher with the PLA University of Science and Technology, where he performed research and development on advanced energy-efficient wireless router, ad-hoc network simulator, and cognitive mesh networking test-bed. From 2006 to 2019, he was an associate professor with the College of Information Science and Electronic Engineering, Zhejiang University. Currently, he is with the Zhejiang Lab, Hangzhou as the chief engineering officer. His research interests include software defined networks (SDNs), wireless network in 6G, computing networks, and collective intelligence. He is the Symposium Co-Chair of ChinaCom 2009 and 2010. He is the Technical Program Committee (TPC) Co-Chair of the 10th IEEE International Symposium on Communication and Information Technology (ISCIT 2010).