Semantics-enhanced Temporal Graph Networks for Content Caching and Energy Saving

Jianhang Zhu*, Rongpeng Li*, Xianfu Chen†, Shiwen Mao‡, Jianjun Wu§ and Zhifeng Zhao*¶
*Dept. of Information Science and Electronic Engineering Zhejiang University, Zheda Road 38, Hangzhou 310027, China
†VTT Technical Research Centre of Finland, 90570 Oulu, Finland
‡Dept. of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201, USA
§Huawei Technologies Company, Ltd., Shanghai 201206, China
¶Zhejiang Lab, Hangzhou 310027, China
Emails: {zhujh20, lirongpeng}@zju.edu.cn, xianfu.chen@vtt.fi, smao@ieee.org,
wujianjun@huawei.com, zhaozf@zhejianglab.com

***Abstract*—The enormous amount of network equipment and users implies a tremendous growth of Internet traffic for multimedia services. To mitigate the traffic pressure, architectures with in-network storage have been proposed to cache popular content at devices in close proximity to users in order to decrease the number of backhaul hops. Meanwhile, the reduced transmission distance also contributes to energy saving. However, due to limited storage, only a fraction of the content can be cached, while caching the most popular content is cost-effective. Correspondingly, it becomes essential to devise an effective popularity prediction method. In this regard, some existing efforts manifest the effectiveness of dynamic graph neural network (DGNN) models, but it remains challenging to tackle sparse datasets. Herein, we first propose a reformative temporal graph network, named STGN, to address the challenge and improve prediction performance. Specifically, the STGN model leverages extra semantic messages to help establish implicit paths within the sparse interaction graph and enhance the temporal and structural learning of a DGNN model. Furthermore, we devise a user-specific attention mechanism to aggregate various semantics in a fine-grained manner. Finally, extensive simulations verify the superiority of our STGN models and demonstrate the potential in terms of energy-saving.**

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

I. Introduction

With the surging demand for high-definition video streaming services, in-network caching is becoming a promising technique to alleviate the burden on the network by deploying storage capability at devices in close proximity to users to cache the popular contents [1]–[3]. Meanwhile, the reduction of backhaul hops also saves energy consumption [3]–[8]. However, it is infeasible to continually increase the device caching capability due to the limitation of reality [9]. This predicament makes the design of effective caching strategies much more desirable. Traditionally, the widely-mentioned reactive caching strategies, such as least recently used and least frequently used, only focus on the patterns of local requests, thus failing to handle the unexpected requests [10]. Accordingly, it becomes inevitable to design proactive caching strategies, wherein the accurate popularity prediction plays a decisive role. Deep neural networks (DNNs) have demonstrated their remarkable potential in popularity prediction. For instance, Ref. [10] uses the long short-term memory (LSTM) to discover the patterns within the temporal content requests, so as to facilitate popularity-assisted proactive content caching. The problem is that due to the lack of historical data, LSTM fails to predict accurately for those inertia users. Fortunately, along with the requests of contents from users, the interactions between users and contents gradually constitute a dynamic bipartite graph. With the help of structural learning, popularity prediction with graph neural networks (GNN) has been proven to be successful [11]. However, most GNN models assume that the inherent graphs are static, which is obviously not the case in practice [12]. Consequently, popularity prediction with dynamic graph neural network (DGNN) has been attracting significant attention. Different from the GNN, a DGNN model is able to jointly learn the structural and temporal patterns of dynamic graphs. Built upon the graph attention network (GAT) [13] and the positional encoding in Transformer [14], Ref. [15] proposes a temporal graph attention mechanism (TGAT) to encode temporal features (i.e., the timestamps of interactions produced by users requesting contents) within the dynamic graphs. TGN in [16] further introduces a temporal learning module before the TGAT for a deeper refinement of the temporal characteristics. Ref. [17] 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 content caching results.

When facing a sparse dataset, it is in general difficult to obtain satisfactory caching performance by deploying the existing models in a straightforward way. In this regard, Ref. [18] introduces a knowledge graph (KG) to incorporate the side information of the requested content into a static graph model, which leverages the implicit associations among the contents and yields superior performance. Nevertheless, this work ignores the dynamics of the interactions between users and contents, while the KG construction also implies a huge

This work was supported in part by the National Natural Science Foundation of China under Grants 62071425, in part by the Zhejiang Key Research and Development Plan under Grants 2022C01093, 2019C01002 and 2019C03131, in part by Huawei Cooperation Project, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005.

978-1-5386-7462-8/23/$31.00 ©2023 IEEE

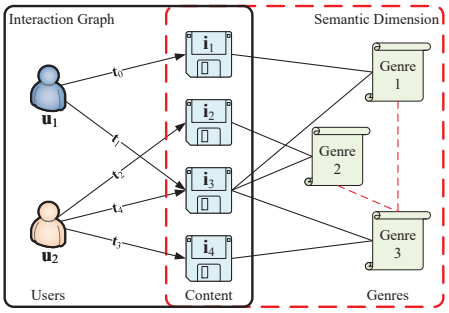

Fig. 1. An example of the dynamic interaction graph and the implicit semantic relationship between contents. The requests of two users, u_1 and u_2, only intersect at content i_3. The sparsity of data makes it intractable for classical GNN-based methods to accurately predict the preference of u_1 for content i_4. On the other hand, the contents' genre information and their underlying similarities, which are indicated by the red dotted lines, in the semantic dimension, reveal a strong correlation between i_4 and i_3 as well as a weak correlation between i_4 and i_1, where i_1 and i_3 are the targets of u_1. Thus, u_1 is likely to request i_4.

demand for side information (e.g., the director and release date of the content), which may not be applicable in many cases.

In this paper, we propose a Semantics-enhanced Temporal Graph Network (STGN) to strengthen the DGNN model performance in dealing with sparse datasets. Different from the KG model in [18], we only adopt the genre information of the content and encode it with the pre-trained natural language processing (NLP) results to establish the implicit relation. In the example shown in Fig. 1, it is intractable for classical GNN-based methods to predict the user preference in a sparse graph. But the attachment of semantics constructs more implicit structural patterns, which facilitates preference inference and may further improve the prediction performance of TGN models [17]. Additionally, a content might possess multiple genres (e.g., a fictional action movie containing both fiction and action genres) and the preferred genre might vary across users as well. Therefore, we further design a user-specific attention mechanism for a finer-grained aggregation of semantics. Although the above popularity-based caching methods have achieved promising performance in terms of cache hit-rate, the energy consumption perspective is overlooked. Hence, in this paper, we also examine the energy consumption by caching the popular contents, which are predicted by the proposed STGN. In summary, the main contributions of this paper are as follows.

- To overcome the limitation of data sparsity and leverage the genre information of requested targets, we propose an STGN to establish the implicit connections between contents and semantic features.
- With the observation that a content possibly carries a variety of semantic information, we devise a user-specific attention mechanism to further exploit the potential semantic relationship and boost the prediction performance.
- Through extensive experiments using a real-world dataset, we verify the prediction accuracy improvement achieved by the STGN model over the conventional TGN models and validate the superiority of the STGN model based proactive caching strategy in terms of the reduced cumulative energy consumption.

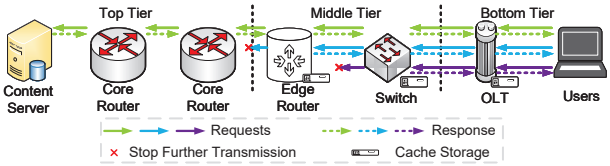

Fig. 2. Caching and response in a multi-tier caching system.

TABLE I
MAJOR NOTATIONS USED IN THE PAPER.

Notation	Definition
u_j, i_k	User j and content k
$\boldsymbol{v}_{u_j}, \boldsymbol{v}_{i_k}, \boldsymbol{e}_{jk}$	Raw features of u_j, i_k and the edge
$p_{jk}(\hat{T}), \tilde{p}_{jk}(\hat{T})$	Real and predicted preferences of u_j for i_k at the future time $\hat{T}$
$\text{Pop}^{i_k}(\hat{T})$	Popularity of i_k at the future time $\hat{T}$
$\mathbf{Msg}_{jk}$	Message that merges all raw features of an interaction
$\mathbf{h}_j(\hat{T})$	Short-term preference of u_j
$\mathbf{Mem}_j, \mathbf{Mem}'_j$	Long-term preference of u_j and the updated formation
$\mathbf{Mem}''_j$	The updated long-term preference of u_j concatenated with the encoded time feature $\Phi_{d_T}(\Delta_t)$
$\mathbf{E}_{u_j}(\hat{T}), \mathbf{E}_{i_k}(\hat{T})$	Final embedding representations for u_j and i_k
$\boldsymbol{E}_{jk}$	User-specific embedding
$\boldsymbol{s}_{kN_s}$	The N_s-th semantic information of i_k
$\boldsymbol{S}_k$	Aggregated semantic feature for semantic messages of all i_k

The remainder of this paper is organized as follows. We introduce system models and formulate the problem in Section II. We elaborate on the details of our proposed prediction model in Section III. In Section IV, we present the experimental results and discussions. Finally, Section V concludes this paper. For convenience, we also list the mainly used notations of this paper in Table I.

II. System Models and Problem Formulation

A. System Models

Network Model: We concentrate on a multi-tier caching system, where caches are scattered over the devices close to users. We conceptually simplify the network as a three-tier topology as below.

- *Top Tier* – It is composed of *core routers*, which connect the host servers of content providers with other network elements.
- *Middle Tier* – It encompasses *edge routers* and *switches*. In particular, the *switches* communicate with the *core routers* through the *edge routers* and are located at a lower sub-tier than the *edge routers*.
- *Bottom Tier* – It consists of *access nodes*, e.g., the optical line terminals (OLTs), which are deployed to connect users with the *switches*.

In this paper, we primarily consider the in-network caching capability of the *edge routers*, *switches*, and *OLTs*. Moreover, as depicted in Fig. 2, 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.

Energy Model: The energy consumption of data transmission mainly stems from the following two factors [19].

- *Caching Consumption* – Unlike the traditional networks, the energy consumption of the cache hardware plays a non-negligible role. Typically, the power consumption of caching is composed of the baseline P_b for maintaining the working status and the storage-dependent consumption P_s [3], both of which are generally assumed to be proportional to the amount of stored data.

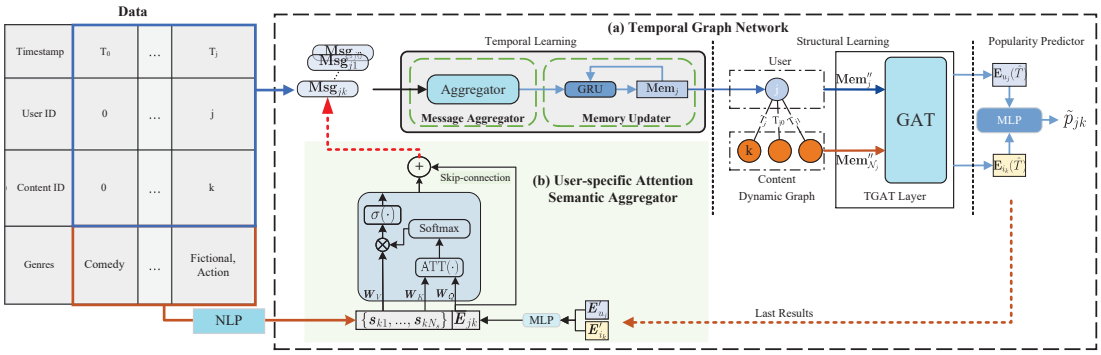

Fig. 3. The illustration of M1−STGN with the user-specific attention mechanism for semantics aggregation.

- *Transport Consumption* – The forwarding of data in the network also leads to certain energy consumption, which is proportional to the amount of transmitted data and usually varies from one device to another.

Thus, the overall consumption of transmitting and caching a content of C_s bits for a duration of t_c is expressed by

$$E^{\text{all}} = C_s[(P_b + P_s)t_c + \sum_{d\in\mathcal{D}} E_d^{\text{tr}} I(d) n_{\text{tr}}], \tag{1}$$

where $d \in \mathcal{D}$ denotes the d-th device in the network, and E_d^{tr} is the corresponding transport consumption. Besides, n_{tr} is the number of requests to the content. $I(\cdot)$ is an indicator function depending on the caching situation, which specifies that if a copy of the content is cached in device d_x, then $I(d) = 0$ for any device d in a higher tier than device d_x.

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 $\mathcal{U} = \{u_0, u_1, ..., u_j\}$ and the set of contents as $\mathcal{I} = \{i_0, i_1, ..., i_k\}$, where u_j and i_k denote the user j and content k, respectively. The raw feature sets of users and contents are denoted as $\mathcal{V}_{\mathcal{U}} = \{\boldsymbol{v}_{u_0}, \boldsymbol{v}_{u_1}, ..., \boldsymbol{v}_{u_j}\}$ and $\mathcal{V}_{\mathcal{I}} = \{\boldsymbol{v}_{i_0}, \boldsymbol{v}_{i_1}, ..., \boldsymbol{v}_{i_k}\}$, 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 can be naturally regarded as the edges, which can be denoted as $\mathcal{E} = \{\boldsymbol{e}_{00}, \boldsymbol{e}_{01}, ..., \boldsymbol{e}_{jk}\}$. Herein, $\boldsymbol{e}_{jk}$ represents the vector of interactions between u_j and i_k. In summary, we represent the dynamic graph as a set of quadruples, $\mathcal{G} = \{(\boldsymbol{v}_{u_0}, \boldsymbol{v}_{i_0}, \boldsymbol{e}_{00}, T_0), ..., (\boldsymbol{v}_{u_j}, \boldsymbol{v}_{i_k}, \boldsymbol{e}_{jk}, T_n)\}$, where T_n denotes the timestamp of the n-th interaction. In addition, we can integrate each quadruple as a piece of historical message, denoted as $\mathbf{Msg}_{jk} = [\boldsymbol{v}_{u_j}||\boldsymbol{v}_{i_k}||\boldsymbol{e}_{jk}||T_n]$, where $||$ is the concatenation operator. Moreover, as each content may contain various semantic genres, we encode all N_s genres of content i_k with some mature NLP methods [20], denoted as $\mathcal{S}_k = \{\boldsymbol{s}_{k1}, ..., \boldsymbol{s}_{kN_s}\}$, so as to fully utilize the inherent semantic characteristics.

B. Problem Formulation

In line with the previous analysis, in order to reduce energy consumption, the more popular contents should be cached at the devices closer to the users [3]. Therefore, it becomes essential to know the popularity of each content in advance. We use $\text{Pop}^{i_k}(\hat{T})$ to represent the popularity of i_k at the future time $\hat{T}$, which can be calculated as

$$\text{Pop}^{i_k}(\hat{T}) = \sum_j \mathbf{1}\left(p_{jk}(\hat{T}) > p_{\text{thre}}\right), \ \forall k \in \mathcal{I}, \tag{2}$$

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 the emergence of such a request, and $\mathbf{1}(\zeta)$ is a function that equals 1 if the condition ζ is satisfied and otherwise, 0.

Though p_{jk} is unknown apriori, we can still calculate a prediction $\tilde{p}_{jk}$ with the embedding representations 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,

$$\tilde{p}_{jk}(\hat{T}) = F\left(\mathbf{E}_{u_j}(\hat{T}), \mathbf{E}_{i_k}(\hat{T})\right), \tag{3}$$

where a multi-layer perceptron (MLP) can be adopted to realize the function $F(\cdot)$. In this paper, our target is to generate feasible representations with the DGNN model from historical messages, so as to minimize the binary cross entropy loss between $\mathbf{p} = \{p_{jk}\}$ and $\tilde{\mathbf{p}} = \{\tilde{p}_{jk}\}$, $\forall u_j \in \mathcal{U}$, $i_k \in \mathcal{I}$,

$$\mathcal{L} = -\sum_{u_j, i_k} \left(p_{jk}\log(\tilde{p}_{jk}) + (1 - p_{jk})\log(1 - \tilde{p}_{jk})\right). \tag{4}$$

III. Semantics-enhanced Temporal Graph Network

In this section, we focus on the design of the STGN, so as to better exploit the dynamic bipartite graph and obtain the 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. The Conventional TGN

As shown in Fig. 3(a), the conventional TGN model is stacked by two prime elements, 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 integrate a user's historical messages into a compressed format. Specifically, the message aggregator aims to leverage u_j's historical messages before the prediction time $\hat{T}$ to obtain the short-term preference $\boldsymbol{h}_j(\hat{T})$, which can be formulated as

$$\boldsymbol{h}_j(\hat{T}) = \text{Agg}\left(\mathbf{Msg}_{j0}, ..., \mathbf{Msg}_{jk}\right), \tag{5}$$

where the filtering and aggregation function $\text{Agg}(\cdot)$ can be implemented diversely. In the remainder of this paper, we take account of three means including filtering the latest message, using the mean value of all messages [16], and attention-based

weighted summation of all messages [17]. We denote them as `TGN-L`, `TGN-M` and `TGN-A`, respectively.

Subsequently, the extracted short-term preference $\boldsymbol{h}_j(\hat{T})$ is used to update the long-term preference $\mathbf{Mem}_j$ by a memory updater, which can be realized by a gated recurrent unit (GRU) with the mathematical formulations as

$$\begin{aligned}
&\mathbf{Mem}'_j = \boldsymbol{Z}\cdot\boldsymbol{H} + (1-\boldsymbol{Z})\cdot\mathbf{Mem}_j,\\
&\boldsymbol{Z} = \sigma\left(\boldsymbol{h}_j(\hat{T})\boldsymbol{W}_{hZ} + \mathbf{Mem}_j\boldsymbol{W}_{MZ} + \boldsymbol{b}_Z\right),\\
&\boldsymbol{F} = \sigma\left(\boldsymbol{h}_j(\hat{T})\boldsymbol{W}_{hF} + \mathbf{Mem}_j\boldsymbol{W}_{MF} + \boldsymbol{b}_F\right),\\
&\boldsymbol{H} = \tanh\left(\boldsymbol{h}_{\mathrm{j}}(\hat{\mathrm{T}})\boldsymbol{W}_{\mathrm{hH}} + (\boldsymbol{F}\cdot\mathbf{Mem}_{\mathrm{j}})\,\boldsymbol{W}_{\mathrm{MH}} + \boldsymbol{b}_{\mathrm{H}}\right),
\end{aligned} \tag{6}$$

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

2) Structural Learning Module

The structural learning module aims to keep the representations up-to-date for the inactive users by exchanging features among neighbors in the graph, and map all user's embeddings to their future representations for prediction. Obviously, the timestamp of each interaction also plays a vital role in this procedure. Therefore, we adopt a TGAT model [15] with the time encoding function on the basis of a classical GAT module [13]. In particular, the time encoding function can be formulated as

$$\Phi_{d_T}(\Delta_t) = \sqrt{\frac{1}{d_T}}[\cos(\omega_1\Delta_t), ..., \cos(\omega_{d_T}\Delta_t)]^T, \tag{7}$$

where $\omega_1, \omega_2, ...$ and ω_{d_T} are the trainable parameters, Δ_t denotes the temporal lag between the request-occurring time and the time to predict $\hat{T}$, and d_T is the dimension number of the desired time encoding.

Then, the encoded results are concatenated to the output of the temporal learning module as the input of the subsequent module.

$$\mathbf{Mem}''_j = [\mathbf{Mem}'_j || \Phi_{d_T}(0)], \tag{8}$$

where (8) reformulates the updated long-term preference $\mathbf{Mem}''_j$ for u_j, whose temporal lag is 0. Besides, for u_j's neighbor $k \in \mathcal{N}_j$, its modified preference term $\mathbf{Mem}''_k$ is formulated as

$$\mathbf{Mem}''_k = [\mathbf{Mem}'_k || \Phi_{d_T}(\Delta_{t_k})], \forall k \in \mathcal{N}_j, \tag{9}$$

Notably, $\mathbf{Mem}''_j$ and $\mathbf{Mem}''_k$ are the inputs for the structural learning module. As depicted in Fig. 3, the GAT architecture [13] is the fundamental part of a TGAT layer for exploiting the structural patterns within the dynamic subgraph of u_j, and can be encapsulated as

$$\mathbf{E}_{u_j}(\hat{T}) = \mathrm{GAT}(\mathbf{Mem}''_j, \mathbf{Mem}''_{\mathcal{N}_j}), \tag{10}$$

Similarly, we generate the embedding $\mathbf{E}_{i_k}(\hat{T})$ from the subgraph of i_k as

$$\begin{aligned}
&\mathbf{Mem}''_k = [\mathbf{Mem}'_k || \Phi_{d_T}(0)],\\
&\mathbf{Mem}''_j = [\mathbf{Mem}'_j || \Phi_{d_T}(\Delta_{t_j})], \forall j \in \mathcal{N}_k,\\
&\mathbf{E}_{i_k}(\hat{T}) = \mathrm{GAT}(\mathbf{Mem}''_k, \mathbf{Mem}''_{\mathcal{N}_k}),
\end{aligned} \tag{11}$$

Note that $\mathbf{E}_{u_j}(\hat{T})$ and $\mathbf{E}_{i_k}(\hat{T})$ are exploited as the input of the prediction module in (3).

B. Semantic Enhancement for TGN

Essentially, the temporal learning module can be deemed as a process of refining the commonality among the historical data. However, the randomly initialized raw data makes it complicated to accurately extract the patterns, especially for a sparse dataset. Consequently, we resort to incorporating semantic information into the raw input, so as to establish the implicit relationship among the historical messages.

We use a pre-trained NLP model, such as Glove [20], to encode the content genre information as semantic messages, $\mathcal{S}_k = \{\boldsymbol{s}_{k1}, ..., \boldsymbol{s}_{kN_s}\}$. Meanwhile, in this part, we intuitively adopt the summation to generate an aggregated feature $\boldsymbol{S}_k$ from $\mathcal{S}_k$, which is then incorporated into the raw message.

$$\boldsymbol{S}_k = \sum_{n\in N_s} \sigma(\boldsymbol{W}_s\boldsymbol{s}_{kn} + \boldsymbol{b}_s), \tag{12}$$

$$\mathbf{Msg}'_{jk} = \sigma(\boldsymbol{W}_1^t\mathbf{Msg}_{jk} + \boldsymbol{W}_2^t\boldsymbol{S}_k), \tag{13}$$

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 (5). Hereinafter, we name the semantics-enhanced TGN in a temporal manner as `M1-STGN`.

Although the fresh features for inactive users can be easily located with the help of the graph structure, the performance still suffers from the data sparsity. To address this issue, we further attach the semantic features to the output of the temporal learning module, establishing implicit semantic pathways between the contents of the dynamic graph. In our experiments, we also discover that concatenation yields superior accuracy than the summation for merging semantics in the structural learning module. Then, (9) is modified as

$$\mathbf{Mem}''_k = [\mathbf{Mem}'_k || \boldsymbol{S}_k || \Phi_{d_T}(\Delta_{t_k})], \forall k \in \mathcal{N}_j, \tag{14}$$

where $\boldsymbol{S}_k$ is calculated by (12). Similarly, we use `M2-STGN` to represent the TGN model that is further facilitated by the structural learning with semantics.

C. User-specific Attention Mechanism for Semantic Aggregation

Although semantic aggregation can be easily achieved with (12), it lacks the capability to distinguish the impacts of different semantics from the same content on different users. Therefore, we aggregate the multiple semantic features with a user-specific attention mechanism, as shown in Fig. 3(b). Mathematically, (12) is reformulated as a linear weighted summation of N_s semantics of content i_k,

$$\begin{aligned}
&\boldsymbol{S}_k = \sigma\left(\sum_{n\in N_s} \alpha_{jn}\boldsymbol{s}_{kn}\boldsymbol{W}_{Vn}\right),\\
&\alpha_{jn} = \frac{\exp(\boldsymbol{E}_{jk}\boldsymbol{W}_Q\cdot\boldsymbol{s}_{kn}\boldsymbol{W}_{Kn})}{\sum_{m=1}^{N_s}\exp(\boldsymbol{E}_{jk}\boldsymbol{W}_Q\cdot\boldsymbol{s}_{km}\boldsymbol{W}_{Km})},
\end{aligned} \tag{15}$$

TABLE II

THE PERFORMANCE OF PREDICTING CONTENT REQUESTS IN BOTH TRANSDUCTIVE AND INDUCTIVE TASKS. TGN−L, TGN−M, AND TGN−A ARE THE CONVENTIONAL TGN MODEL'S VARIANTS WITH DIFFERENT MESSAGE AGGREGATORS. IN OUR M1−STGN AND M2−STGN, SUM AND ATTENTION BELONG TO TWO SEMANTIC AGGREGATORS IN (12) AND (17), RESPECTIVELY. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND THE SECOND-BEST RESULTS ARE HIGHLIGHTED IN UNDERLINED.

Metric		*AUC for Transductive*	*AP for Transductive*	*AUC for Inductive*	*AP for Inductive*
Conventional	TGN−L	85.299	83.824	77.285	76.995
	TGN−M	86.731	86.022	78.953	79.721
	TGN−A	90.507	90.691	83.504	84.999
M1−STGN with Sum	M1−STGN−L	86.386	85.892	79.980	80.439
	M1−STGN−M	88.312	88.095	82.043	82.765
	M1−STGN−A	91.210	91.337	85.247	86.175
M1−STGN with Attention	M1−STGN−L	89.014	88.467	83.096	83.483
	M1−STGN−M	89.721	89.356	83.585	84.148
	M1−STGN−A	91.358	91.572	85.327	86.567
M2−STGN with Sum	M2−STGN−L	87.383	86.806	81.131	81.182
	M2−STGN−M	88.649	88.558	82.805	83.461
	M2−STGN−A	91.773	91.953	85.877	87.019
M2−STGN with Attention	M2−STGN−L	89.749	89.279	84.183	84.387
	M2−STGN−M	90.107	89.884	84.434	84.868
	M2−STGN−A	**91.846**	**92.056**	**86.264**	**87.279**

TABLE III

THE ENERGY CONSUMPTION OF ALL DEVICES IN OUR NETWORK MODEL.

Device	Energy Consumption E_d^{tr} (J/bit)
Server	$2.81 \cdot 10^{-7}$
Core Router	$1.7 \cdot 10^{-8}$
Edge Router	$2.63 \cdot 10^{-8}$
Switch	$8.21 \cdot 10^{-9}$
OLT	$1.4 \cdot 10^{-7}$

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

$$\boldsymbol{E}_{jk} = \text{LeakyReLu}(\boldsymbol{W}_u \boldsymbol{E}'_{u_j} + \boldsymbol{W}_i \boldsymbol{E}'_{i_k} + \boldsymbol{b}_{ui}), \tag{16}$$

where $\boldsymbol{W}_u$, $\boldsymbol{W}_i$ and $\boldsymbol{b}_{ui}$ are the trainable parameters, while $\boldsymbol{E}'_{u_j}$ and $\boldsymbol{E}'_{i_k}$ [1] are the results generated in the last prediction or the initialization values for the first prediction of u_j and i_k, respectively.

To avoid the over-smoothing caused by the stacking of various DNN layers and augment the effect of semantics, we further leverage the skip-connection in GAT [13] to improve the overall performance. Specifically,

$$\boldsymbol{S}_k \leftarrow N_s \cdot \boldsymbol{S}_k + \boldsymbol{E}_{jk} \tag{17}$$

which is the final representation that we use in (13), so as to further optimize M1−STGN or the temporal learning module of M2−STGN.

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

A. Experimental Settings

In this paper, the experiments use the public Netflix[2] dataset, which covers a set of user behaviors on Netflix in UK. We filter those records associated with users who have more than 4 requests and view each requested content for more than 3 minutes as the valid input data for prediction. The preprocessed dataset includes 86,889 interactions, which involve 11,254 different users and 4,057 pieces of content. Obviously, it is even much sparser than the dataset used in [17]. Afterwards, we perform a 60%-20%-20% chronological split of the dataset for training, validation, and testing, respectively. In order to verify the effectiveness of our proposed models, we conduct experiments in both transductive task and inductive task. Compared with the transductive task, the validation set and test set in inductive task may contain some vertices that have not been inferred by the models in the training phase. For both tasks, we choose the *average precision* (AP) and the *area under the ROC curve* (AUC) as evaluation metrics.

As for caching, we randomly select 24 hours of the data in the test set of inductive task, which includes some unseen vertices as in practice. We predict the per-hour popularity and update the caching once an hour, i.e., the t_c in (1) is 1 hour. Moreover, we configure a network architecture with two core routers, one edge router, one switch, and one OLT, as depicted in Fig. 2. In Table III, we summarize the per-bit energy consumption E_d^{tr}. Typically, the devices closer to users in the network usually have a smaller storage capacity [3]. Hence, we assume that the OLT can store 5 contents with each taking 3 GB storage, while 7 and 8 contents can be cached at the switch and the edge router, respectively. Furthermore, we assume that all caches use high-speed solid-state drive (SSD) technology. And the storage-dependent power of an SSD is $P_s = 6.25 \cdot 10^{-12}$ W/bit, while its baseline power is $P_b = 0.025$ W/GB [19].

Moreover, we also investigate the impact of the request number (i.e., n_{tr} in (1)) of a content, while maintaining the popularity distribution. Specifically, we assume that each request in the dataset is made by a cluster consisting of multiple users with the same request pattern. We compare the energy consumption of different caching strategies.

B. Results

Table II demonstrates the prediction performance of our proposed models (i.e., M1−STGN, M2−STGN), and their vari-

[1] For simplicity, we omit the time information $\hat{T}'$ in $\boldsymbol{E}'_{u_j}(\hat{T}')$ and $\boldsymbol{E}'_{i_k}(\hat{T}')$.

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

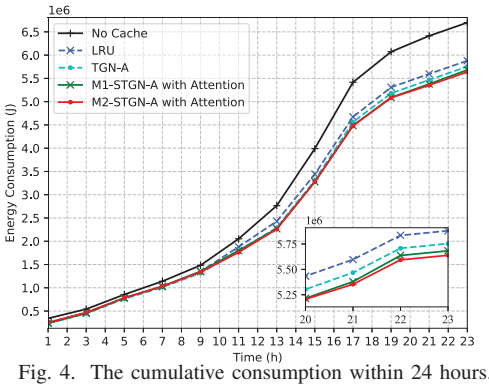

Fig. 4. The cumulative consumption within 24 hours.

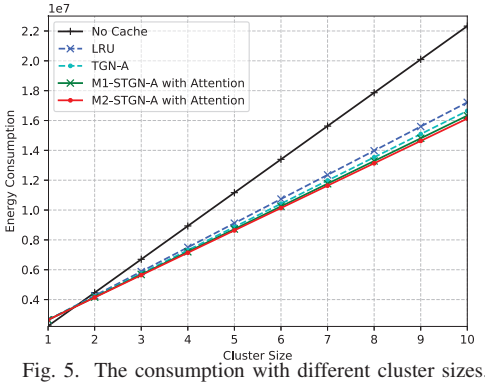

Fig. 5. The consumption with different cluster sizes.

ants, as well as the original TGNs in [16], [17]. With a sparse dataset, it is clear that our models can yield better results in both transductive task and inductive task, even when we aggregate the semantic information only by summation. Moreover, the superiority of `M1-STGN` and `M2-STGN` with attention also proves the effectiveness of our proposed semantic aggregator. In addition, if we further consider the computational complexity, the satisfactory results of `M2-STGN-L` and `M2-STGN-M` with attention, which have lower computational complexity [17], also imply the potential in energy-saving.

Figs. 4 and 5 show the energy consumption from different popularity prediction strategies. Fig. 4 shows the 24-hour cumulative energy consumption with 3 users in a cluster. The curves demonstrate that due to the stronger prediction capability of STGN, a more considerable gain of energy consumption can be expected along with the increase in the number of requests compared with `TGN-A` based caching. In addition, Fig. 5 shows the impact of the cluster size, which implies the recurring number of the same content request, on energy consumption. It can be observed that the gain in energy consumption benefits from the superior prediction accuracy of our methods on the repeated requests of popular contents.

V. Conclusions

In this paper, we have proposed an STGN architecture to improve the performance of popularity prediction for in-network caching. By attaching semantic information to the temporal learning and structural learning modules of TGN, the proposed STGN models have demonstrated superior prediction accuracy with a sparse dataset. In addition, by considering that a piece of content may contain multiple semantics, we have devised a user-specific attention mechanism for a more efficient semantic aggregation, which further enhances the prediction accuracy. The in-network caching based on our STGN models also consumes less energy compared to the existing baselines.

References

[1] "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

[2] Z. Hajiakhondi Meybodi, A. Mohammadi, E. Rahimian, *et al.*, "TEDGE-Caching: Transformer-based edge caching towards 6G networks," in *Proc. ICC*, Seoul, South Korea, May 2022.

[3] O. Ayoub, F. Musumeci, M. Tornatore, *et al.*, "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.

[4] F. Guo, H. Zhang, X. Li, *et al.*, "Joint optimization of caching and association in energy-harvesting-powered small-cell networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6469–6480, Jul. 2018.

[5] F. H. Panahi, F. H. Panahi, and T. Ohtsuki, "Energy efficiency analysis in cache-enabled D2D-aided heterogeneous cellular networks," *IEEE Access*, vol. 8, pp. 19 540–19 554, Jan. 2020.

[6] M. Amadeo, C. Campolo, G. Ruggeri, *et al.*, "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.

[7] M. I. A. Zahed, I. Ahmad, D. Habibi, *et al.*, "A review on green caching strategies for next generation communication networks," *IEEE Access*, vol. 8, pp. 212 709–212 737, Nov. 2020.

[8] Q. N. Nguyen, M. Arifuzzaman, K. Yu, *et al.*, "A context-aware green information-centric networking model for future wireless communications," *IEEE Access*, vol. 6, pp. 22 804–22 816, Apr. 2018.

[9] O. Serhane, K. Yahyaoui, B. Nour, *et al.*, "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.

[10] A. Narayanan, S. Verma, E. Ramadan, *et al.*, "Making content caching policies 'smart' using the deepcache framework," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 48, no. 5, pp. 64–69, Oct. 2018.

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

[12] 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. 79 143–79 168, May 2021.

[13] P. Veličković, G. Cucurull, A. Casanova, *et al.*, "Graph attention networks," in *Proc. ICLR*, Vancouver, BC, Canada, Apr./May 2018.

[14] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Proc. NeurIPS*, Long Beach, CA, USA, Dec. 2017.

[15] D. Xu, C. Ruan, E. Korpeoglu, *et al.*, "Inductive representation learning on temporal graphs," in *Proc. ICLR*, Virtual Edition, Apr./May 2020.

[16] E. Rossi, B. Chamberlain, F. Frasca, *et al.*, "Temporal graph networks for deep learning on dynamic graphs," in *Proc. ICML 2020 Workshop on Graph Representation Learning*, Virtual Edition, Jul. 2020.

[17] J. Zhu, R. Li, G. Ding, *et al.*, "AoI-based temporal attention graph neural network for popularity prediction and content caching," *arXiv preprint arXiv:2208.08606*, 2022.

[18] Y. Liu, S. Yang, Y. Xu, *et al.*, "Contextualized graph attention network for recommendation with item knowledge graph," *IEEE Trans. Data Knowl. Eng.*, May 2021, Early Access.

[19] N. Choi, K. Guan, D. C. Kilper, *et al.*, "In-network caching effect on optimal energy consumption in content-centric networking," in *Proc. ICC*, Ottawa, Canada, Jun. 2012.

[20] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. EMNLP*, Doha, Qatar, Oct. 2014.