Toward Knowledge as a Service (KaaS): Predicting Popularity of Knowledge Services Leveraging Graph Neural Networks

Haozhe Lin, Yushun Fan, Jia Zhang, *Senior Member, IEEE,* Bing Bai, Zhenghua Xu, Thomas Lukasiewicz

Abstract—Knowledge services are becoming a rising star in the family of XaaS (Everything as a Service). In recent years, people are more willing to search for answers and share their knowledge directly over the Internet, which drives the knowledge service ecosystem prosperous and quickly evolve. In this paper, we aim to predict the popularity of knowledge services, which will benefit the downstream industries that provide Knowledge as a Service (KaaS). Toward such a task, the spatial interactions (*e.g.*, hyperlinks in *Wikipedia*) and temporal observations (*e.g.*, page views) provide crucial information. However, it is difficult to utilize this information due to: (i) complicated and different usage observations, (ii) intricate and evolutionary spatial interactions, and (iii) small world trait of the network. To tackle such issues, we propose Evolutionary Graph Convolutional Recurrent Neural Networks (E-GCRNNs) to simultaneously model both temporal and spatial dependencies of knowledge services from their evolving networks. Specifically, an elementary unit (called E-GCGRU) is designed to dynamically perceive the evolutionary spatial dependencies, aggregate spatial information of knowledge services, and model the temporal patterns by considering the records of one sequence and its neighbors simultaneously. Additionally, a localized mini-batch training scheme is developed, which allows the E-GCRNNs to work on large-scale knowledge services networks and reduce the prediction bias caused by the small world trait. Extensive experiments on real-world datasets have demonstrated that the proposed E-GCRNNs outperform baselines in terms of prediction accuracy, especially with the prediction range being longer, while remaining computationally efficient.

Index Terms—Knowledge as a Service, Popularity Prediction, Spatiotemporal Prediction, Graph Convolutional Networks

1 Introduction

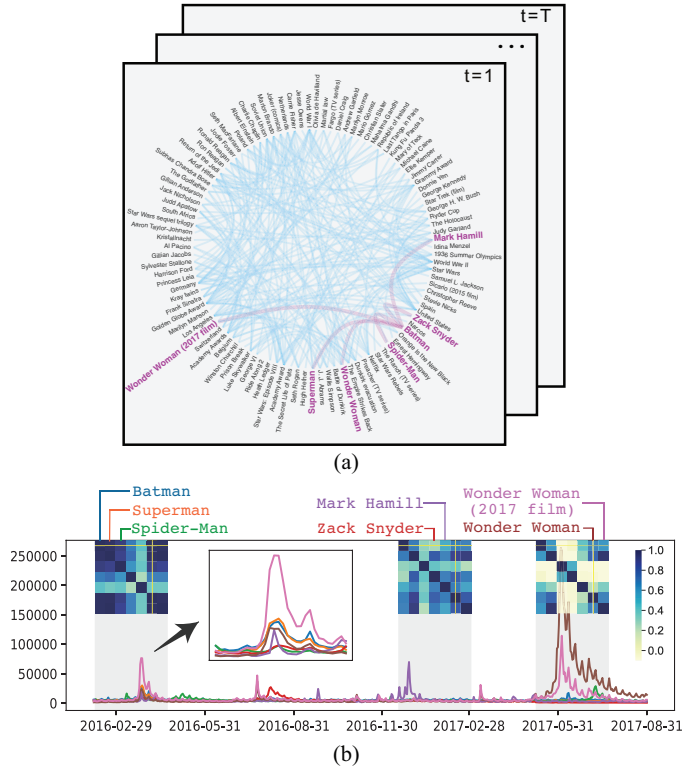
In the era of big data, people are no longer only satisfied with raw Data as a Service. Instead, people desire to gain the insights behind the data – the knowledge. With the penetration of the services computing techniques in the last two decades, increasingly more knowledge has been wrapped up as universally accessible services and published online. In contrast to traditional knowledge sharing over the Internet, such as static web pages, knowledge-oriented services (or knowledge services in short) typically aim to provide dynamic, context-aware, and customized information delivery. For example, when users ask questions on *zhihu.com* or *quora.com*, the popular knowledge-oriented question-and-answer platforms, it will trigger a backend API and generate discriminatingly ranked answer lists (*i.e.*, knowledge) to users. Furthermore, the service may proactively invite users to answer others' questions by identifying their specialty. In such a context, the concept of Knowledge as a Service (KaaS)[1] has been coined and becoming a rising star in the family of Everything as a Service (XaaS), joining Software as a Service (SaaS), Data as a Service (DaaS), Infrastructure as a Service (IaaS), and so on. A number of well-known products of KaaS have emerged in the recent years, such as *Wikipedia*[2], *Google Scholar*[3], and *Quora*[4].

The prosperity in knowledge services provides significant opportunities for service management, discovery, and recommendation. Such visions motivate us to study the predictions of the **popularity of knowledge services**, which may benefit many downstream service industries. For example, if we can identify the potentially popular ones from thousands of academic papers, the quality of paper (knowledge service) recommendations would be improved. In particular, the popularity of knowledge services can be reflected by their usage tendency, like page views in *Wikipedia*, citation counts in *Google Scholar*, and likes in *Quora*. Throughout this paper, we will use two terms interchangeably, *knowledge service* and *service*.

The historical usage observations and the interaction relationships among knowledge services provide useful information for predicting the usage tendency or popularity of the services. However, to fully utilize them, three unique traits demand meticulous considerations:

- *H. Lin, Y. Fan are with the Department of Automation, Tsinghua University, Beijing, China and Beijing National Research Center for Information Science and Technology. E-mail: linhz@mail.tsinghua.edu.cn, fanyus@tsinghua.edu.cn.*
- *J. Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX, USA. E-mail: jiazhang@smu.edu.*
- *B. Bai is with the Cloud and Smart Industries Group, Tencent, Beijing, China. E-mail: icebai@tencent.com.*
- *Z. Xu is with the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, China. E-mail: zhenghua.xu@hebut.edu.cn.*
- *T. Lukasiewicz is with the Department of Computer Science, University of Oxford, UK. E-mail: Thomas.Lukasiewicz@cs.ox.ac.uk.*

1. https://en.wikipedia.org/wiki/Knowledge_as_a_service
2. https://www.wikipedia.org/
3. https://scholar.google.com/
4. https://www.quora.com/



(a) (b)

Fig. 1: Spatial dependencies and temporal usage observations of real-world *Wikipedia* knowledge services. The upper part shows the hyperlinks among 103 random Wikipedia entries. The lower part shows the usage observations of 7 knowledge services and their evolutionary spatial dependencies.

- **Complicated and different temporal observations.** Knowledge services usually present complicated temporal traits, including periodicity, nonlinearity, and (non)stationarity. For example, the service *Christmas*[5] is intensively visited in a one-year periodicity, while presenting complex nonlinearity during the other time. Besides, different knowledge serivces generally present different usage patterns. For example, the observations of the service *Christmas* are distinctly different from those of other services, like *United States presidential election*.
- **Intricate and evolutionary spatial interactions.** The usage tendency or popularity of one knowledge service not only depends on the historical usage observations of the service itself, but also more or less relies on the invocation records of its related services. Take Figure 1(a) as a real-world example, which shows 103 entries and their inter-relationships (*i.e.*, hyperlinks) from *Wikipedia*. Since the service *Batman* is connected with six other services through hyperlinks, its page views could also come from its (one or multiple-step reachable) neighbors. Based on such a fact, the structure of the knowledge service network, presenting non-euclidean and evolutionary, makes it difficult to utilize such information. As shown in Figure 1(b), the service *Batman* highly correlates to the service *Wonder Woman* in February 2016; however, their spatial dependency drops dramatically in May 2017.
- **Small world trait.** Our previous study reveals a service network presents a small world characteristic, meaning that most of the services are connected through several hops, while the others are either isolated or reside in small clusters. Such characteristic also poses more significant challenges for aggregating local spatial information in practice. On the one hand, popular services will cause almost unacceptable footprints, if we consider all their connected neighbors. On the other hand, long-tailed services may waste computational resources since the number of their neighbors are quite small.

5. https://en.wikipedia.org/wiki/Christmas

In summary, given the aforementioned intractable facts, making accurate predictions of the popularity of knowledge services remains an arduous task.

Many existing works have attempted to predict the popularity or usage tendency of knowledge [1], [2], [3], [4]. However, most of them make predictions individually without considering spatial dependencies among knowledge services. In recent years, with the advancement of graph convolutional networks (GCNs) [5], [6], [7], researchers have been able to effectively aggregate information and extract features from non-Euclidean data structures, such as citation networks, traffic networks, (software) service networks, and so on. Based on graph convolutional operators, diffusion convolutional recurrent neural networks (DCRNNs) [8] and spatiotemporal graph convolutional neural networks (STGCNNs) [9] were developed to predict traffic flow and have shown significant improvements, which are the state of the arts for making spatiotemporal predictions. However, different from the relatively static *physical* spatial dependencies in traffic networks, in our problem domain, the interactions among knowledge services, regarded as *virtual* ones, are dynamic and evolve frequently with time. For example, people could frequently edit the content of Wikipedia entries, causing changes in the relations among them. As a result, existing works assuming the interaction graph is fixed cannot solve our evolutionary problem. Besides, to aggregate spatial information for the service vertices of interest, it requires the model to include their neighbors as inputs. Many existing works based on GCNs do not consider the large scale of vertices, and directly train their models in full-batch setting [8]. However, since the number of knowledge services is large and keeps increasing, such methods are impractical for our problem.

In order to model both spatial and temporal dependencies, and adapt to the evolutionary ones, we propose Evolutionary Graph Convolutional Recurrent Neural Networks (E-GCRNNs) to predict the tendency of knowledge service invocation. In more detail, graph convolutional operators first aggregate localized information. In the meantime, a dynamic module identifies the changes of graphs, and triggers the model to make a response. Finally, a Gated Recurrent Unit (GRU)-like structure controls the temporal dependencies and makes predictions. Besides, we propose a localized mini-batch training scheme to efficiently allevi-

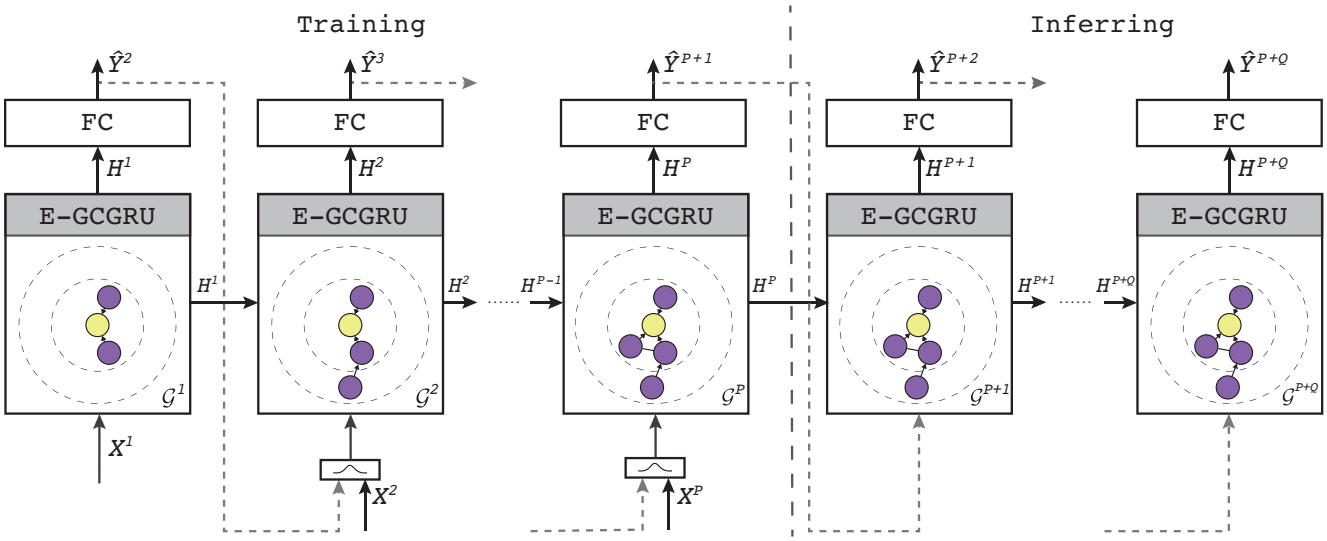


Fig. 2: Overall architecture of E-GCRNNs. The E-GCGRU receives the current observation X^t, previous hidden states H^{t-1}, and evolutionary interaction graphs $\mathcal{G}^t$ as inputs. To predict the trend of the knowledge service of interest, *i.e.*, the yellow node, the past values of it and its neighbors, *i.e.*, the purple nodes, are sampled and aggregated by E-GCGRU as hidden states H^t. At the same time, H^t is regressed by a fully connected layer (FC) to produce the prediction $\hat{Y}^{t+1}$. When training, we applied scheduled sampling to randomly decide whether to replace the observation with the last prediction.

ate the computational burden, using a divide-and-conquer strategy. To our knowledge, this is the first attempt to study the prediction of knowledge service invocation tendency by learning both temporal and dynamic spatial dependencies. Our main contributions are three-fold:

- We have proposed evolutionary graph convolutional recurrent neural networks (E-GCRNNs), which can learn the spatiotemporal dependencies among knowledge service usage sequences, flexibly adapt to evolutionary ones, and eventually make accurate predictions.
- We have developed a localized mini-batch training scheme, where large-scale knowledge services are divided into independent blocks to make the training procedure more efficient.
- We have designed and conducted a collection of experiments on real-world datasets, which show that E-GCRNNs outperform state-of-the-art algorithms in terms of prediction accuracy, especially when the prediction period is longer.

The remainder of this paper is organized as follows. Section 2 gives notations and mathematically restates the knowledge service invocation prediction problem. Section 3 introduces the proposed model, and Section 4 describes the training details. Section 5 reports our experimental results. Section 6 reviews related work. Finally, Section 7 draws conclusions.

2 Preliminaries

In this section, we firstly present the definitions of notations, as well as the spatial dependencies among knowledge services, namely the service dependency graph, then formally introduce the problem definition.

2.1 Notation Definition

Definition 1 (Usage observations of knowledge services)**.** *Given one knowledge service, its usage observations refer to its usage times in the past P units of time. Therefore, we denote the usage observations of knowledge service by a time series $\boldsymbol{x}_i$, where $\boldsymbol{x}_i$ can be broken down to $\boldsymbol{x}_i = \{x_i^1, x_i^2, \ldots, x_i^P\}$. More specifically, x_i^t $(t \in [1, P])$ represents the number of usage times for service i in the past t-th day. In the problem that we consider, there are N services in total, so we denote the usage records of all services by $\boldsymbol{X} = \{\boldsymbol{x}_1; \boldsymbol{x}_2; \ldots; \boldsymbol{x}_N\}$.*

Definition 2 (Usage tendency of knowledge service)**.** *The usage tendency of a knowledge service refers to the usage times of the service in the next Q units of time, which reflects the popularity of knowledge services. In this paper, we denote the usage tendency of knowledge service by $\hat{\boldsymbol{y}}_i = \{\hat{y}_i^{P+1}, \hat{y}_i^{P+2}, \ldots, \hat{y}_i^{P+Q}\}$, where $\hat{y}_i^t$ $(t \in (P+1, P+Q])$ represents the predicted value of invocation times for service i in the next t-th day. Likewise, $\hat{\boldsymbol{Y}} = \{\hat{\boldsymbol{y}}_1; \hat{\boldsymbol{y}}_2; \ldots; \hat{\boldsymbol{y}}_N\}$ denotes the tendency of a collection of knowledge services that is expected to be used.*

Definition 3 (Knowledge service network)**.** *To represent the spatial dependencies among knowledge services, we construct a service dependency graph $\mathcal{G}^t = (\mathcal{V}, \mathcal{E}, \mathbf{W}^t)$, with vertices $\mathcal{V}$ as services set, edges $\mathcal{E}$ as the dependencies set, and weights $\mathbf{W}^t$ as the intensities of corresponding dependencies at different times. Specifically, $\mathbf{W}^t$ can be broken down to w_{ij}^t to represent the interaction intensity between services i and j at time t.*

In this paper, we regard distinct spatial dependencies among knowledge services (*e.g.*, citation in *Google Scholar* and hyperlinks in *Wikipedia*) are continuous values, and try to perceive their changes.

2.2 Problem Restatement

Problem (Predicting the usage tendency of knowledge service)**.** *Given N knowledge services in a service ecosystem, our*

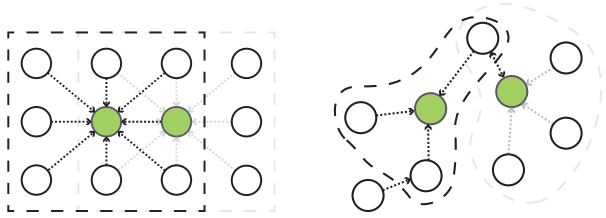
goal is to predict their usage tendency in the next Q units of time, i.e., $\hat{\boldsymbol{Y}} = \{\hat{\boldsymbol{y}}_1; \hat{\boldsymbol{y}}_2; \ldots; \hat{\boldsymbol{y}}_N\}$ *for all services, based on their previous P usage records during the past P units of time observations,* i.e., $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ *for all services and their evolutionary service dependency graph $\mathcal{G}^t$.*

3 Model Architecture

In this section, we introduce our proposed E-GCRNNs for predicting the usage tendency of knowledge services. At a high level, E-GCRNNs apply vanilla RNN structure without encoder-decoder structure, and the recurrent unit – E-GCGRU aggregates information in close proximity, perceives changing intensities, and extracts temporal features. The overall architecture is shown in Figure 2.

3.1 Aggregate Spatial Influence

As mentioned in Section 1, to predict the tendency of one knowledge service (*i.e.*, the yellow node in Figure 2), not only its past records should be considered, the observations of its neighbors (*i.e.*, the purple nodes) also matter. However, unlike data with regular structure (*e.g.*, image where local feature of pixel can be aggregated through canonical convolution in Figure 3 (a)), the structure of knowledge services presents non-Euclidean property, which means we cannot learn unified convolution kernels for the knowledge service network. As shown in Figure 3 (b), for the vertices of interest with green color, they have different numbers of first-order neighbors. Therefore, we introduce advanced spectral graph convolution to E-GCGRU for aggregating local features for knowledge services in such an irregular network.



(a) canonical convolution (b) graph convolution

Fig. 3: Diagram of different convolutions.

To begin with, we represent the knowledge service network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ in a form of matrix[6], namely a Laplacian matrix $\boldsymbol{L}$, ($\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$), where $\boldsymbol{D}$ and $\boldsymbol{A}$ refer to the degree matrix and adjacency matrix of $\mathcal{G}$, respectively. Since we only consider the correlation relationship between knowledge services in this paper[7], which means the adjacency matrix $\boldsymbol{A}$ is symmetric, we have the symmetric normalized Laplacian matrix, *i.e.*, $\boldsymbol{L}^{sym} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$, where $\boldsymbol{I}$ represents the identity matrix. Based on the Laplacian matrix $\boldsymbol{L}$[8], we introduce the Chebyshev graph convolutional operator to aggregate information for service nodes from their neighbors, which can be formulated in Equation (1)[9]:

$$f_\theta \star_{\mathcal{G}} \boldsymbol{Z} = f_\theta(\boldsymbol{L})\boldsymbol{Z} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\boldsymbol{L}})\boldsymbol{Z}, \tag{1}$$

where f_θ represents the graph convolutional filter with θ as trainable parameter; $\boldsymbol{Z}$ represents the inputs of E-GCGRU, and particularly, $\boldsymbol{Z} = [\boldsymbol{X}^t, \boldsymbol{H}^{t-1}]$ with $\boldsymbol{X}^t$ as the observations of all related services nodes at current time, *e.g.*, all colored nodes in Figure 2, and $\boldsymbol{H}^{t-1}$ as the hidden states from the last E-GCGRU; K represents a predefined order, which determines the range of considered neighbors; $\tilde{\boldsymbol{L}}$ represents the re-scaled Laplacian matrix to fit the application condition of Chebyshev polynomial, *i.e.*, $\tilde{\boldsymbol{L}} = \frac{2}{\lambda_{\max}}\boldsymbol{L} - \boldsymbol{I}$, with $\lambda_{\max}$ as the largest eigenvalue of $\boldsymbol{L}$; finally, $T_k(\tilde{L})$ represents a recursive manner to aggregate information at order k-th, and specifically, $T_k(\tilde{\boldsymbol{L}}) = 2\tilde{\boldsymbol{L}}T_{k-1}(\tilde{\boldsymbol{L}}) - T_{k-2}(\tilde{\boldsymbol{L}})$, with $T_1(\tilde{\boldsymbol{L}}) = \tilde{\boldsymbol{L}}$ and $T_0(\tilde{\boldsymbol{L}}) = \boldsymbol{I}$.

Remarkably, Equation (1) presents great localized property, which limits the information that can pass to the central vertices to be within its K-step reachable neighbors. That is because by utilizing the Chebyshev polynomial, the gradients recurrently flow through the approximation formula, *i.e.*, $T_k(\tilde{\boldsymbol{L}}) = 2\tilde{\boldsymbol{L}}T_{k-1}(\tilde{\boldsymbol{L}}) - T_{k-2}(\tilde{\boldsymbol{L}})$. Such property will be helpful to design efficient training scheme.

3.2 Perceive Changing Interactions

Different from other problem domains that assume the graph is fixed, we hold a view that the virtual spatial dependencies among knowledge services evolve frequently, which sometimes even lead to a change of graph structure. For example, a breakthrough in Deep Learning, the attention mechanism, could connect the domain knowledge of computer vision and natural language processing more tightly. Under such a situation, if the model cannot perceive such changes in the graph, it may waste up-to-date information or be misled by the outdated data when performing inference. Therefore, it is demanding to study how sequences influence each other dynamically.

In this research, we found the Pearson coefficient can serve as an efficient metric to quantify the correlations between two service usage sequences. Therefore, we update the coefficients dynamically, and the model can timely adapt to the latest spatial dependencies and appropriately aggregate information from one's neighbors:

$$w_{ij}^t = \begin{cases} \frac{\mathbb{E}\left[(\boldsymbol{x}_i^{t-\tau:t} - \mu_i)(\boldsymbol{x}_j^{t-\tau:t} - \mu_j)\right]}{\sigma_i \sigma_j} & \text{if } \{i,j\} \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where w_{ij}^t represents the correlation or spatial dependency between knowledge services i and j at time t; $\boldsymbol{x}_i^{t-\tau:t}$ represents the observation sequence of knowledge service i from $t-\tau$ to t; μ_i and σ_i represent the corresponding average and standard deviation of the observation sequence; and $\mathbb{E}(\cdot)$ represents the mean operator. Specifically, we only quantify the spatial dependencies between two knowledge services if

6. In this section, we omit superscript t of $\mathcal{G}^t$ to simplify the following equations.

7. As for the more challenging problem, namely modeling the causation in service invocation, we will leave it as a future work.

8. The following $\boldsymbol{L}$ refers to symmetric normalized Laplacian $\boldsymbol{L}^{sym}$.

9. Interested readers can refer to [6], [7], [10] for the details of the definition, principle, and approximation of the graph convolution.

there exist distinct connections (*e.g.*, citation in *Google Scholar* and hyperlinks in *Wikipedia*), *i.e.*, $\{i, j\} \in \mathcal{E}$.

Since the spatial dependencies change frequently, directly updating the correlations for all linkages may introduce unacceptable computational costs in neural networks. Consequently, we decide to make a trade-off between the evolution assumption and computational complexity, and further suppose W^t remains fixed in a relatively short time interval, so that the correlations can be updated in an acceptable frequency. Our ideas are summarized in Algorithm 1.

Algorithm 1 Spatial dependencies updating algorithm

Input: service invocation sequences: $\boldsymbol{X}$; sparse adjacency matrix: $\boldsymbol{A}$; current moment: t.
Initialize: fixed interval: τ.
Output: adjacency matrix: $\boldsymbol{A}$.

$row, col \leftarrow \boldsymbol{A}.indices()$
$\boldsymbol{P} \leftarrow \boldsymbol{X}[row, t-\tau : t]$
$\boldsymbol{Q} \leftarrow \boldsymbol{X}[col, t-\tau : t]$
$\mu_P, \mu_Q \leftarrow mean(\boldsymbol{P}), mean(\boldsymbol{Q})$
$\sigma_P, \sigma_Q \leftarrow std(\boldsymbol{P}), std(\boldsymbol{Q})$
$\boldsymbol{W}^t \leftarrow \frac{mean[diag((\boldsymbol{P}-\mu_P)(\boldsymbol{Q}-\mu_Q)^\top)]}{\sigma_P \sigma_Q}$ ▷ Equation (2)
$\boldsymbol{A}^t \leftarrow$ construct_sparse_matrix$(row, col, \boldsymbol{W}^t)$
return $\boldsymbol{A}^t$

Algorithm 1 describes a process to calculate pairwise Pearson correlation coefficients (Equation (2)) for all included nodes in a sub-graph in a matrix form. Specifically, we construct two service invocation records matrices, *i.e.*, $\boldsymbol{P}$ and $\boldsymbol{Q}$, corresponding the indices of the sparse adjacency matrix $\boldsymbol{A}$, and then apply Equation (2) to these two matrices.

3.3 Learn Temporal Dependency

Based on extracted spatial features, we follow the structure of DCGRUs [8] to apply a gated mechanism to capture the long- and short-term temporal dependencies. In contrast to DCGRUs, we consider the evolution of the dependency graph, and then formulate E-GCGRUs by Equation (3):

$$\begin{aligned} \boldsymbol{r}^t &= \sigma(f_r \star_{\mathcal{G}^t} [\boldsymbol{X}^t, \boldsymbol{H}^{t-1}] + \boldsymbol{b}_r) \\ \boldsymbol{u}^t &= \sigma(f_u \star_{\mathcal{G}^t} [\boldsymbol{X}^t, \boldsymbol{H}^{t-1}] + \boldsymbol{b}_u) \\ \boldsymbol{C}^t &= \tanh(f_C \star_{\mathcal{G}^t} [\boldsymbol{X}^t, (\boldsymbol{r}^t \odot \boldsymbol{H}^{t-1})] + \boldsymbol{b}_C) \\ \boldsymbol{H}^t &= \boldsymbol{u}^t \odot \boldsymbol{H}^{t-1} + (1 - \boldsymbol{u}^t) \odot \boldsymbol{C}^t, \end{aligned} \tag{3}$$

where $\boldsymbol{X}^t \in \mathbb{R}^{n \times d_i}$ represents the invocation sequences of all included knowledge services nodes, *e.g.*, all colored nodes in Figure 2 at time t; $\boldsymbol{H}^t$ represents the output of the E-GCGRU at time t; similarly to the structure of GRUs, $\boldsymbol{r}^t$, $\boldsymbol{u}^t$ and $\boldsymbol{C}^t$ represent the output of the reset and update gates, and the temporary states of the unit at time t, respectively; f_r, f_u, and f_C contain graph convolutional filters with different trainable parameters; and particularly, $\mathcal{G}^t$ represents the knowledge service network at time t. Note that in this paper, we only consider the usage observations of knowledge services are with a single dimension, *i.e.*, $d_i = 1$. Furthermore, for some particular cases with more meaningful observations, like *Quora* counting both like and collection of knowledge services, we can easily extend the model with larger d_i.

One major difference between our E-GCGRUs and regular GRUs is that, the multiplications in GRUs are replaced by the graph convolutional operators described in previous sections. Such a change implies that E-GCGRUs can not only capture the temporal dependencies from past records of one sequence, but also consider the observations of one's neighbors. Besides, compared with DCGRUs, the dynamic perception in our E-GCGRUs further promotes the flexibility of learning dependencies from changing structure.

3.4 Predict Trend of Time Series

After obtaining the temporal features $\boldsymbol{H}^t$, we train a fully connected layer (FC in Figure 2) to make final predictions, which is shown in Equation (4):

$$\hat{\boldsymbol{Y}}^{t+1} = \boldsymbol{W}^\top \boldsymbol{H}^t + \boldsymbol{b}, \tag{4}$$

where $\boldsymbol{H}$ represents the hidden states from E-GCGRUs that contains spatiotemporal features; $\boldsymbol{Y}$ represents the predicted values; $\boldsymbol{W}$ and $\boldsymbol{b}$ represent the learnable parameters of the fully connected layer.

Finally, E-GCRNNs iteratively treat the predictions at time t as the observation inputs of the E-GCGRU at time $t+1$ (namely the dashed line shown in Figure 2), and thus multi-step usage tendency of knowledge services can be predicted in an auto-regressive manner.

4 Parameter Learning

In this section, we discuss parameter tuning and optimization.

4.1 Loss Function

To seek the optimal parameters, we use the following loss function:

$$\mathcal{L} = \frac{1}{n} \sum_{i,t} \left| y_i^{t+1} - \hat{y}_i^{t+1} \right|, \tag{5}$$

where n represents the number of samples in a batch; $\hat{y}_i^t$ and y_i^t represent the prediction results of E-GCRNNs and the ground truth, respectively. Since the order of magnitude of popular and long-tailed knowledge services differ significantly from each other, we use logarithmic transformation in practice to alleviate the bias and obtain the residual in a relative meaning.

4.2 Localized Mini-Batch Training Scheme

Not only the first-order neighbors, but also any (infinite steps) reachable ones are informative. Thus, one important problem is which vertices should be included as inputs. As analyzed in Section 3.1, the approximation formula of graph convolution, *i.e.*, Equation (1), is truncated by K, which implies that only K-step reachable neighbors could be valid for the center vertices of interest. Therefore, we can divide all reachable knowledge services into multiple batches for training, which inspires our localized mini-batch training scheme (Algorithm 2). Taking Figure 4 as an example, there are two major rounds of localized sampling:

Algorithm 2 Localized mini-batch training algorithm

Input. Interaction graph: $\mathcal{G}^t$.
Initialization. Hierarchy number: M; Maximum number of central nodes: n_m; Predefined batch size: n_{bs}^m.
Output. Localized mini-batch: $\tilde{\mathcal{G}}^t$.

Maintain a dictionary $\mathcal{D}$ for each nodes, where the key is node ID, and the value is their neighbors within K-order.
Sort the keys into several hierarchies $\mathcal{H} = h_m$ based on their size.
while $\mathcal{H} \neq \emptyset$ **do**
  $m \leftarrow randint(M)$.
  if $h_m \neq \emptyset$ **then**
    $\mathcal{N} \leftarrow$ Randomly sample n_m nodes. ▷ 1st round
    $\mathcal{N} \leftarrow \cup\{\mathcal{D}(n), n \in \mathcal{N}\}$.
    if $|\tilde{\mathcal{G}}^t| < n_{\text{bs}}^m$ **then**
      $\mathcal{N} \leftarrow$ Sample $n_{\text{bs}}^m - |\tilde{\mathcal{G}}^t|$ nodes. ▷ 2nd round
    end if
    $\tilde{\mathcal{G}}^t \leftarrow construct\_graph(\mathcal{N})$.
    return $\tilde{\mathcal{G}}^t$.
  else
    Remove h_m from $\mathcal{H}$.
  end if
end while

- **1st round.** In each training or inference procedure, we first randomly pick up several nodes (*e.g.*, nodes $\#1$ and $\#5$) and sample their neighbors within K-order (*e.g.*, all nodes in green) into a subgraph $\tilde{\mathcal{G}}_t$. As for the inference procedure, we predict the trend of the central nodes, so only one round is required.
- **2nd round.** For training, in order to make full use of computational resources, we casually sample other nodes from the rest (*e.g.*, nodes in purple) to fulfill a predefined batch size (*e.g.*, $n_{\text{bs}} = 16$) in this round. After sampling, if there exists a link between two nodes, we also connect them in the new graph $\tilde{\mathcal{G}}_t$. Note that, during training, the trend of all sampled nodes is predicted and used for updating model parameters.

Recalling the small world trait of the knowledge service network, this algorithm simultaneously considers the popular and long-tailed services. For the popular services, Algorithm 2 limits the number of neighbors step by step. For the long-tailed services, Algorithm 2 complements causal neighbors for training, which also contributes to the data augmentation. Besides, by dividing the services into several hierarchies according to their popularity and iteratively updating the model parameters, the model could be less sensitive to the size of neighbors.

5 Experiments

We have conducted extensive experiments to evaluate the effectiveness and efficiency of our proposed E-GCRNNs. In this section, we first introduce our experimental settings, and then analyze in detail our experimental results. Code and data are available at https://www.simflow.net/Team/linhaozhe/E-GCRNNs.zip

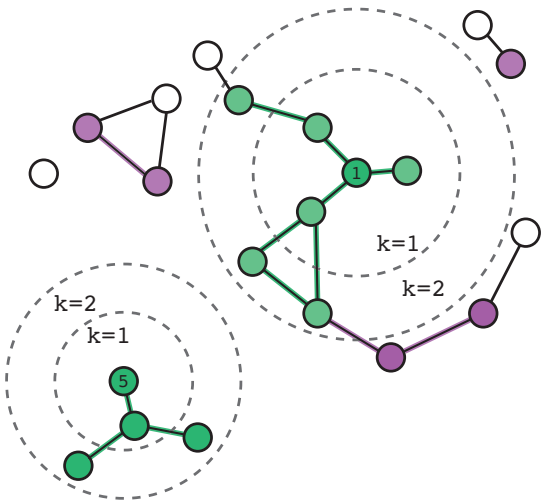


Fig. 4: Diagram of the localized mini-batch training scheme. All colored vertices and edges are sampled, and a new subgraph $\tilde{\mathcal{G}}$ is constructed, where the green are sampled in the 1st round, and the purple are sampled in the 2nd round.

5.1 Experimental Settings

5.1.1 Dataset Selection

As mentioned in Section 1, *Wikipedia* is a representative product of knowledge services, and has been widely studied in public competition[10] and literature [11], [12]. Therefore, we adopt it as our experimental dataset to test and verify our E-GCRNNs. The page views of Wikipedia entries were originated from WikiStat[11], which hourly records the usage counts for thousands of entries from English, German, and other wiki-projects since 2013. To study the interactions among these knowledge services, we crawled the hyperlinks among these entries to construct the knowledge service network.

This real-world dataset perfectly fits our problem in the following three aspects. Firstly, for the temporal dependencies, different page views apparently present different and complicated temporal characteristics. Secondly, for the spatial dependencies, the Wikipedia entries interact with each other. Besides, since the content of Wikipedia entries is frequently altered, the corresponding intensities of the spatial dependencies are different and evolutionary. Thirdly, the network structure of Wikipedia entries presents small world trait. As shown in Figure 5, when we consider 10-step reachable neighbors, *i.e.*, the purple histogram, most of the vertices are connected, while others are isolated. Based on these similarities in all three aspects, we believe it is reasonable to use the WikiStat dataset to test and verify our model.

Since different wiki-projects present gigantic distinctions [13], without losing generality, we selected English and German wiki-projects to simulate the service invocation data. In our experiments, we randomly sampled $4,118$ and $4,321$ entries from English and German Wikipedia projects, and crawled the hyperlinks among these entries to construct the service dependency graphs, respectively. For both

10. https://www.kaggle.com/c/web-traffic-time-series-forecasting
11. https://dumps.wikimedia.org

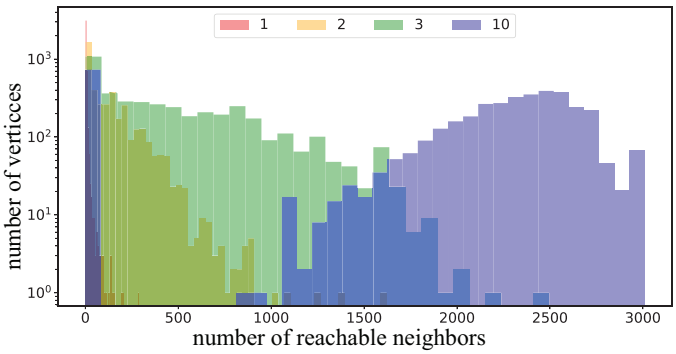


Fig. 5: Histogram of Wiki-EN dataset. Within total 4,118 knowledge services, the red, yellow, green and purple ones represent the histogram of 1, 2, 3 and 10-step reachable neighbors, respectively.

TABLE 1: Numerical properties of datasets

Datasets	Entries	Links	Samples
Wiki-EN	4,118	11,198	3,265,574
Wiki-DE	4,321	8,173	3,426,553

datasets, we used page views from July 1, 2015, to June 30, 2017, for training and validating the models, and those from July 1, 2017, to August 31, 2017, for testing. Table 1 shows the details of the datasets, where samples refer to the number of invocation records for different nodes at different timestamps.

5.1.2 Evaluation Schemes

In our study, due to the huge difference in the order of magnitude of different service invocations or Wikipedia page views, absolute indicators, like Mean Absolute Error (MAE) and Mean Square Error (MSE), cannot appropriately reflect the prediction residual. For example, imagine there are ten sequences, one is with a thousand scale, while the others are with decadal scale. When the one with a thousand scale is predicted inaccurately, no matter how precisely the others are predicted, a model will be deemed a bad one, if MSE or MAE is used for evaluation. Therefore, in this study, we introduce two indicators to evaluate the performance of our model, being the Root Mean Squared Logarithmic Error (RMSLE) [18] and the Symmetric Mean Absolute Percentage Error (SMAPE) [19].

The first indicator RMSLE evaluates the model by shrinking the prediction result to a logarithmic scale, which alleviates the impact caused by order of magnitude. RMSLE can be formulated by Equation (6), and a lower RMSLE represents a higher prediction accuracy.

$$\text{RMSLE} = \sqrt{\frac{1}{n}\sum_{i,t}[\log(y_i^t+1) - \log(\hat{y}_i^t+1)]^2}. \quad (6)$$

The second indicator SMAPE reflects a relative residual of prediction results, which can also solve the problem of the order of magnitude. Besides, an under-forecasting prediction gets a higher value than an over-forecasting one. It is quite suitable for our problem, because a slight redundancy is essential for any kind of role in the service ecosystem, as explained in earlier sections. SMAPE can be formulated by Equation (7), and a lower SMAPE represents a higher prediction accuracy.

$$\text{SMAPE} = \frac{1}{n}\sum_{i,t}\frac{|\hat{y}_i^t - y_i^t|}{(\hat{y}_i^t + y_i^t)/2}. \quad (7)$$

5.1.3 Baselines

We compared our E-GCRNN with five representative baselines, which are described as follows.

- **ARIMA [14].** AutoRegressive Integrated Moving Average (ARIMA) is the most classical time series prediction model, which has been widely used in many industries. ARIMA can easily capture the linearity of one time series. We implemented this method through the *statsmodel* python package[12].
- **VAR [15].** Vector AutoRegressive (VAR) model is an extension of the ARIMA model, which further takes the interaction among sequences into consideration. In the experiments, we set one sequence with all its one-order neighbors as a group, and trained different VAR models for each individual sequence.
- **SVR [16].** Support Vector Regression (SVR) can model the nonlinearity of time series by utilizing different kernels. We implemented this method through the sklearn python package[13].
- **FC-GRU [17].** Recurrent neural networks (RNNs) have been widely used for sequence generating due to their great capability of learning the long-time dependencies of sequences through vast records. In this paper, we replaced the LSTMs with GRUs to keep consistent with the module setup of DCRNNs and our E-GCRNNs, and name it Fully Connected Gated Neural Units (FC-GRUs).
- **DCRNNs [8].** Diffusion Convolutional Recurrent Neural Networks (DCRNNs) is the state-of-the-art model for spatiotemporal prediction, which exploits graph convolutional filters to learn the interaction among sequences, utilizes the GRUs to capture the temporal dependencies, and then makes predictions.

Among these baseline models, ARIMA and SVR make predictions individually; VAR considers the interactions among sequences; FC-GRUs and DCRNNs are RNN-based models, which can learn general features from the whole collection of sequences; and DCRNNs exploits the interactions among sequences and is state of the art in this field.

5.1.4 Hyper-parameters and other settings

All of our experiments were conducted on an Ubuntu server [CPU: Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz, GPU: NVIDIA GTX 1080 Ti]. To make the comparison fair, we apply regular procedures in validating sets, including grid search, early-stopping, and control variates, for hyper-parameters selections. In particular, we have tried different numbers of RNN hidden states in $[\mathbf{128}, 256]$, and initial

12. https://www.statsmodels.org
13. https://pypi.org/project/scikit-learn

TABLE 2: SMAPE and RMSLE of different approaches in page views prediction of English and German Wikipedia.

Model	Wiki-EN		Wiki-DE	
	SMAPE	RMSLE	SMAPE	RMSLE
ARIMA [14]	60.3986 ± 0.0000	1.0147 ± 0.0000	69.8497 ± 0.0000	1.1893 ± 0.0000
VAR [15]	64.5642 ± 0.0000	1.2483 ± 0.0000	69.4245 ± 0.0000	1.1160 ± 0.0000
SVR [16]	41.2042 ± 0.0000	0.7334 ± 0.0000	41.9852 ± 0.0000	0.7163 ± 0.0000
FC-GRU [17]	32.6536 ± 0.5627	0.5477 ± 0.0229	36.1476 ± 0.7813	0.6132 ± 0.0077
DCRNNs [8]	30.4030 ± 0.8642	0.5298 ± 0.0028	35.6445 ± 0.8306	0.6030 ± 0.0063
GCRNNs	30.3667 ± 0.9487	0.5288 ± 0.0049	35.5654 ± 1.19861	0.6048 ± 0.0079
E-GCRNNs	$\mathbf{29.9606 \pm 0.4559}$	$\mathbf{0.5225 \pm 0.0041}$	$\mathbf{34.7936 \pm 0.2047}$	$\mathbf{0.5978 \pm 0.0023}$

learning rate in $[10^{-2}, \mathbf{10^{-3}}, 10^{-4}]$. We utilized Adam optimizer for all NN-based models and applied early-stopping to control the convergence conditions. Overall, all the hyper-parameters were tuned for different models to achieve their best performance. As for the most sensitive hyper-parameters for the GCN-based models, *i.e.*, the predefined graph convolution order K, we will discuss in detail in Section 5.2.3.

5.2 Experiment Results

5.2.1 Main Results

To compare the overall prediction accuracy of our E-GCRNNs with those of baseline models, we conducted repeated experiments ten times with different initializations. Table 2 records the average and standard deviation of SMAPE and RMSLE from different methods in two-month (*i.e.*, 62 days) prediction. Specifically, GCRNNs refer to a degradation of our E-GCRNN, which considers the correlations among knowledge services are static. Examining the results over both datasets, we noticed five consistent phenomena. First, all RNN-based models, including our E-GCRNNs, significantly outperform the previous ones, which should result from the GRUs efficiently capturing long-term dependencies of sequences. Second, by comparing the accuracy of ARIMA and VAR, we observed that although VAR models the correlations among time series, its errors were not less than those of ARIMA. This phenomenon indicates that simple matrix operations in VAR are not sufficient to learn intricate relationships among knowledge services. Third, three methods (DCRNNs, GCRNNs, and E-GCRNNs), adopting graph convolutional operators to exploit the spatial dependencies, gain significant strides, demonstrating that spectral graph convolution operator is effective for modeling complicated services relationships. Fourth, by comparing the results between DCRNNs and GCRNNs, it seems that the encoder-decoder structure does not contribute to the improvement, since GCRNNs are slightly outperformed. Finally, our E-GCRNNs, learning the changes of the interactions among sequences, performs the best against the baselines. In the datasets, our E-GCRNNs gain around $1.1 \sim 1.3\%$ with a low standard deviation under both metrics.

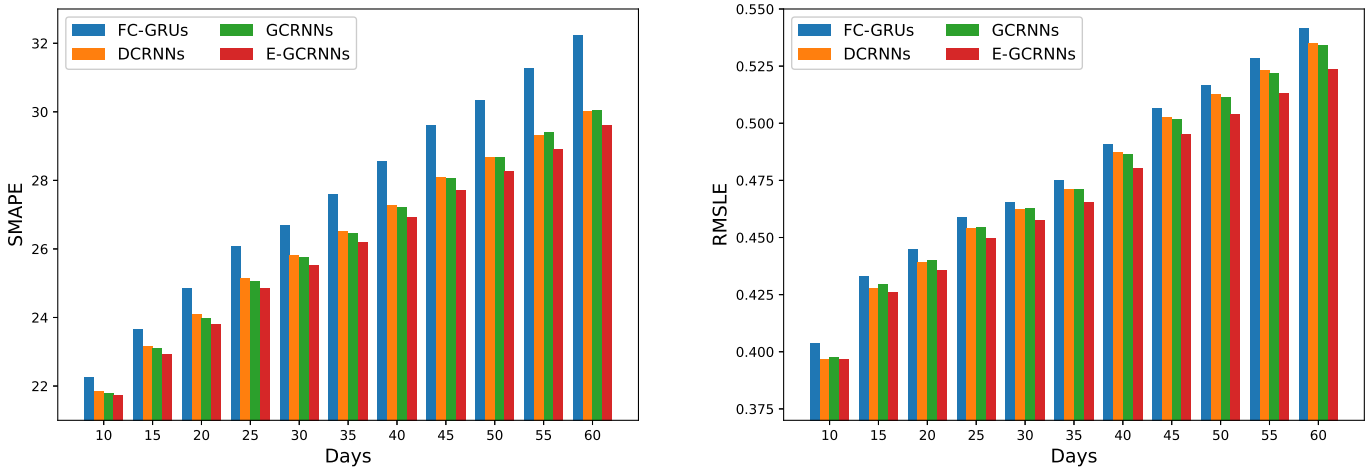
5.2.2 Long-term predictions

The accuracy of long-term prediction is an important property of the models. Therefore, we report the prediction errors changing with increasing prediction lengths in Figure 6. Among these methods, FC-GRUs is the fundamental model only applying a gated mechanism to learn temporal dependencies, while the other three models utilize spectral graph convolution to fuse spatial information with temporal ones. The only difference between DCRNNs and GCRNNs is whether to use an encoder-decoder structure or not. Furthermore, our E-GCRNNs are the only model considering the evolution of spatial dependencies.

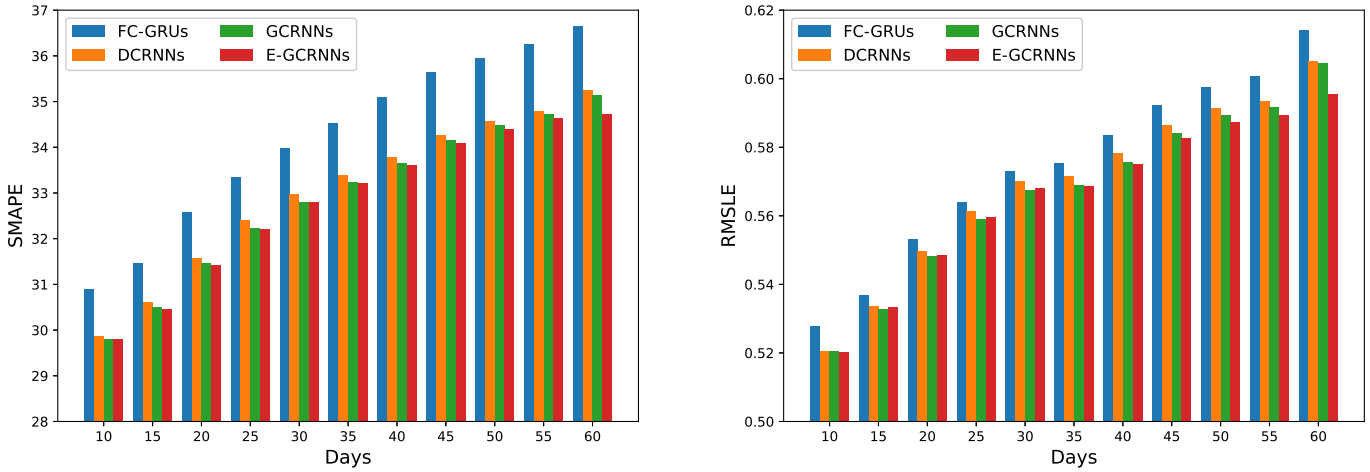
Figure 6 elaborates the performance of all approaches with different ranges of prediction time. Longitudinally, in the beginning, all the models achieve a better prediction accuracy compared to themselves, as the trends in a short time depend more on short-term dependency and thus are easier to be predicted. Among these models, FC-GRUs, being not able to utilize the spatial dependencies among sequences, performs worse than the other three models; while the other three models, *i.e.*, DCRNNs, GCRNNs, and E-GCRNNs, perform similarly at the beginning. Gradually, with the prediction horizon of interest becoming longer, the SMAPE and RMSLE of all the models increase, with DCRNNs and GCRNNs being very similar, demonstrating the encoder-decoder structure is not essential. However, our E-GCRNNs turn increasingly smaller than those of DCRNNs and GCRNNs. Such an observation indicates that our dynamic perception module considering the evolution of the correlations brings improvements for long-term prediction.

5.2.3 Impact of Predefined Order

We noticed that the prediction accuracy of our E-GCRNNs is significantly influenced by the predefined order K, which determines the receptive field of graph convolutional filters. Therefore, we carefully studied the impact of K, and reported the results in Figure 7. When $K = 0$, E-GCRNNs degrades to FC-GRUs, which only utilize the records of one sequence itself to make a prediction, and thus presents the lowest prediction accuracy among these methods. When $K = 1$, the SMAPE and RMSLE in both the training set and the testing set descend greatly, demonstrating the efficacy of our E-GCGRU. However, with growing K, the SMAPE of the training set goes down, while that of the testing set rebounds. In our experiments, we found that $K = 1$ and $K = 2$ are the best hyper-parameters for the English and German datasets, respectively. Consistently, through our experimental results, we found that our E-GCRNNs easily overfit with a large K. Thus, developing an advanced regularization technique for our model will be an important future work.



(a) Wiki-EN



(b) Wiki-DE

Fig. 6: SMAPE and RMSLE of different models with growing prediction length.

5.2.4 Training Efficiency

We also studied the efficiency and utility of the proposed localized mini-batch training scheme for a static and dynamic setup, and reported the average training time per batch and parameter numbers in Table 3. Focusing on fixed batch size (512 or 1024), our E-GCRNNs are slightly more time-consuming than the DCRNNs and GCRNNs, when they show a similar performance. The extra cost of the E-GCRNN is the result of its dynamic perception module frequently updating the correlations. However, as shown in Table 2 and Figure 6, the dynamic perception module significantly increases prediction accuracy. Thus, we deem such slightly extra time cost acceptable. Additionally, it is noticeable that the amount of model parameters of the DCRNNs is almost twice those of the GCRNN and E-GCRNNs, due to the encoder-decoder structure. While in this case, we found that our E-GCRNNs, which lack an encoder-decoder structure, can provide comparable performance in terms of prediction accuracy with fewer parameters.

TABLE 3: Time cost and model size of different models.

Model	Time (Sec./Batch)		# parameters
	$n_{\text{bs}} = 512$	$n_{\text{bs}} = 1024$	
DCRNNs	1.663	1.691	201,345
GCRNNs	1.658	1.694	100,737
E-GCRNNs	1.859	2.113	

5.2.5 Case Studies

After observing the prediction results of many cases, we noticed some interesting traits of our E-GCRNNs. In this section, we select three of the most representative ones to vividly present these features.

Case 1. Figure 8 (a) shows the ground truth and prediction results of Wikipedia entry *Olivia Munn*, which has two neighbors in our dataset. Based on the knowledge that actress *Olivia Munn* played a popular role *Psylocke* in the movie *X-Men: Apocalypse*, it is easy to identify a strong connection between the page views of the entry *Olivia Munn* with those of its neighbors. For this typical case, the SMAPE

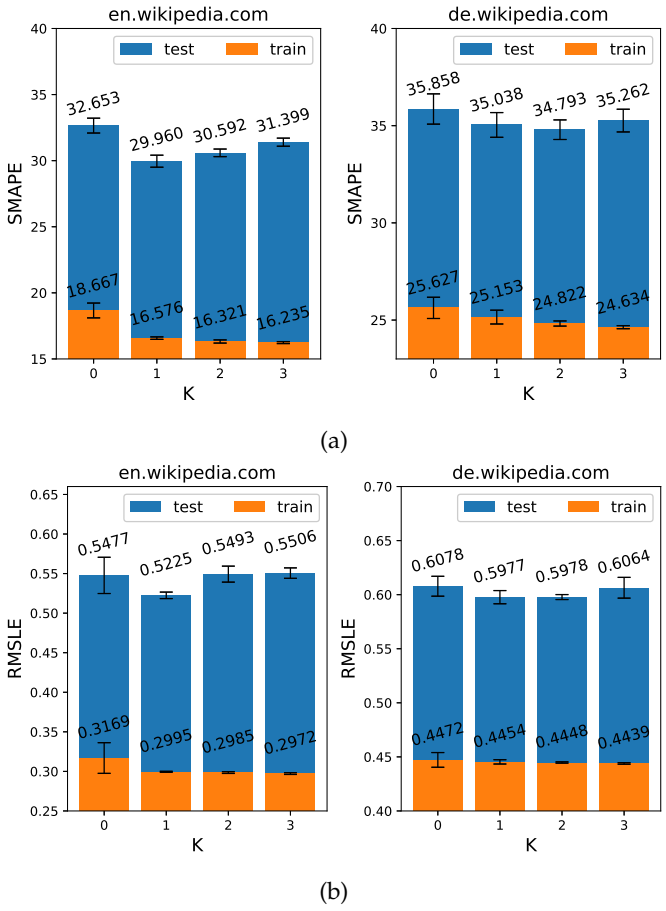


Fig. 7: SMAPE and RMSLE changes with different predefined order K.

of the FC-GRU, DCRNN, GCRNN and our E-GCRNN is 32.1421, 26.9466, 26.2038, 25.5751, respectively, where the performance of all the GCN-based models are better than that of the FC-GRU. After observing many cases like this one, we believe the graph convolution filters can utilize the historical records of the neighbors to improve the prediction accuracy of the central nodes.

Case 2. Figure 8 (b) shows the ground truth and prediction results of Wikipedia entry *The OA*, which also has two neighbors in our dataset but the correlation between these three entries evolve largely with time. The SMAPE of the FC-GRU, DCRNNs, GCRNNs and our E-GCRNNs are 56.5899, 96.5173, 65.5465, 15.7331, respectively. From cases like Figure 8 (b), we can conclude that the performance of GCN-based models greatly relies on the graph. More specifically, with the outdated correlation between sequences, DCRNNs and GCRNNs, two GCN-based models, perform even worse than the simple FC-GRU model, while our E-GCRNN model perceiving the dynamic correlation among sequences can better utilize the graph and make remarkably more accurate predictions than other models.

Case 3. Figure 8 (c) shows the ground truth and prediction results of the Wikipedia entry *Avengers: Infinity War*, which has 18 neighbors in our dataset. In this typical case, the SMAPE of the FC-GRU, DCRNNs, GCRNNs and our E-GCRNNs are 34.299, 84.7189, 100.8242, 52.5599, respectively. From this case, we suspect that for a popular node, *i.e.*, a node with many neighbors, the GCN-based models may be more likely to be misled by its not-so-important neighbors. Here, we see the entry *Avengers: Infinity War* once shows a strong correlation with all the plotted neighbors. However, when we attached attention to the tendency of the page views at the end of the training set, we found the tendency of entries *Avengers: Infinity War*, *Marvel Cinematic Universe*, *Chris Hemsworth* present ascending trends, while those of the entries *Zoe Saldana*, *Thanos* present descending trends. Therefore, for such popular nodes, the ability of the GCN-based model to identify the noisy neighbors and utilize the important ones to make predictions requires careful study in the future.

In summary, as illustrated in the above cases, the GCN-based models are able to utilize the (virtual) spatial dependencies and capture the temporal dependencies among sequences. In particular, our E-GCRNN performs the best in the evolving sequences due to its ability to adapt to the changing correlation among sequences dynamically. However, the GCN-based models also show a limitation in the overfitting problem, which will be one of our important future works.

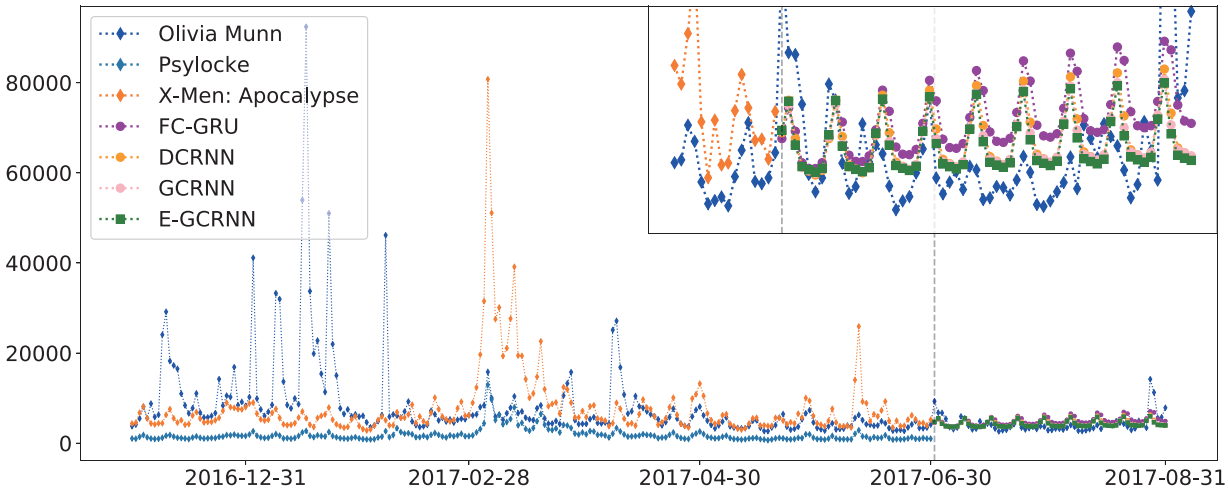
6 Related Work

In this section, we compare our work with related work in the literature from three aspects: service networks, time series prediction, and spectral-based graph convolutional networks.

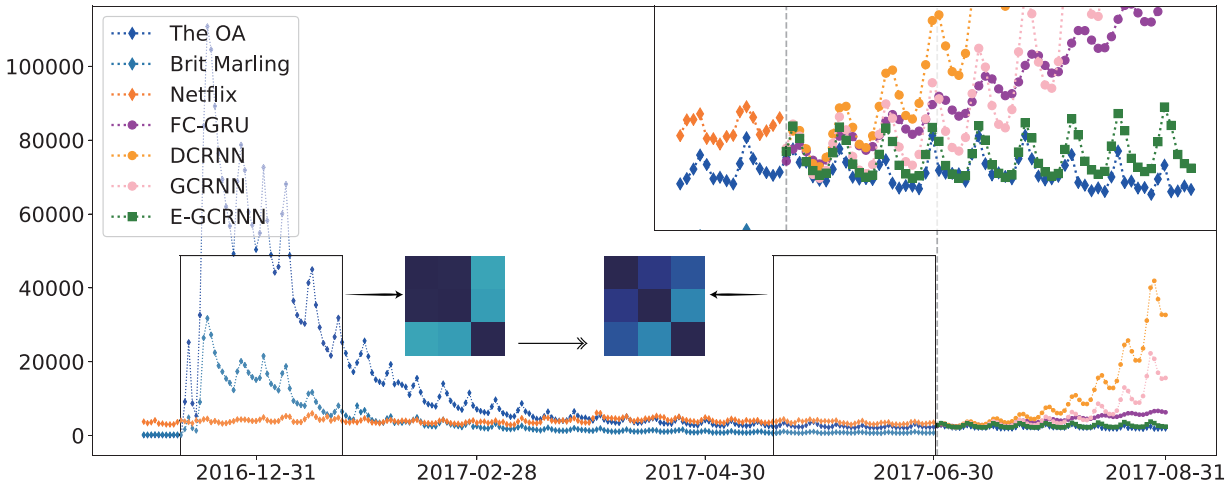
6.1 Service Networks

In recent years, with the development of cloud computing, big data, and the Internet of Things, Service-Oriented Architecture (SOA) has been widely accepted as a mainstream paradigm in the domain of software engineering [20], [21], [22], [23]. As a consequence, a great amount of services, especially knowledge services, have been developed and published into the service ecosystems, which subsequently construct intricate service networks [24], [25], [26], [27], [28].

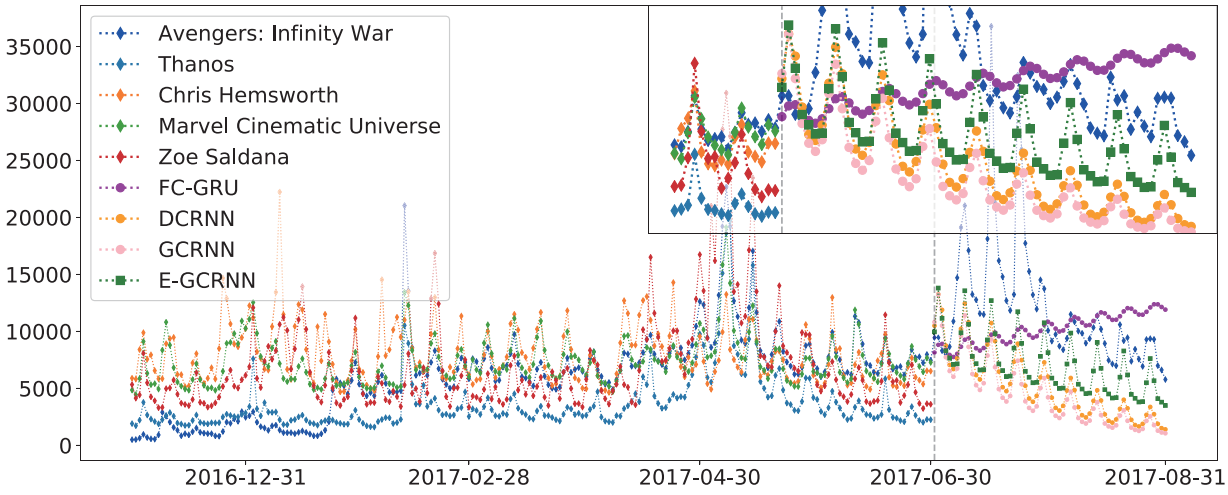
In such contexts, networked services have posed many new challenges in the traditional services computing paradigm. For example, in terms of service recommendation, traditionally, people mainly counted on the functional descriptions of services, namely Web Service Description Language (WSDL), to make recommendations through a keyword search or topic modeling [29], [30], [31], [32], while ignoring the interactions among services. In recent years, non-negative matrix decomposition and neural network-based models are developed to exploit the collaborative relationships among services [33], but these methods all face the problem of cold start. Some recent works try to utilize service networks to make recommendations and gain extra improvement in recommendation accuracy [34], [35]. However, such work, including service recommendation and service tendency prediction, can only learn general patterns of service invocation, which are not able to directly make use of the relationships among services [11]. By comparison, our work applies graph convolution to capture the evolving interactions among services in service networks, which can



(a) Olivia Munn



(b) The OA



(c) Avengeres: Infinity War

Fig. 8: Typical cases. Each figure represents page views of one Wikipedia entry, with dash lines before the gray dash line (July 1, 2017) representing the ground truth and the others after the dash line representing the prediction results of different models. Specifically, the blue dash line represents the ground truth of the central entry, the purple one, orange one, pink one, and green one represent the prediction results of the FC-GRU, DCRNN, GCRNN and E-GCRNN, respectively, and the others represent the true page views of the neighbors of the central entry.

become a strider to improve the prediction accuracy of trends of service invocation.

6.2 Time Series Prediction

The prediction of time series has been an enduring problem for decades. In the beginning, researchers focused on predicting an individual time series. For example, the autoregressive integrated moving average (ARIMA) and support vector regression (SVR) [16] was proposed to model the linearity and non-linearity of one sequence. Similarly, the point process model is also a classical model for modeling individual sequences. In particular, related to our work, Xiao *et al.* and Liu *et al.* use it to successfully predict the usage tendency of knowledge service (paper/patent citation counts) [3], [4]. After that, people were also interested in constructing the relationships among multiple sequences. Representative examples are the vector autoregressive model (VAR) [15] and multiple-output SVR [36].

In recent years, using recurrent neural networks (RNNs), a number of models have been developed to predict large-scale time series [37], [38]. They substantially improve previous work, due to their ability to capture long-term dependencies of RNNs, especially because of their unique units — Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). As an example related to our work, Wen *et al.* applies RNNs to predict the citation counts [39]. However, traditional RNNs can only learn general changing patterns from a broad range of sequences. As a result, the connections among sequences are not well exploited. Therefore, based on RNNs, people have started to study how to utilize the interactions among large-scale time series to improve prediction accuracy. Among the trails, Goel *et al.* combined VAR with RNN [40], and Lai *et al.* used 2D convolutional filters to extract the relations among sequences [41]. However, these models, either based on matrix decomposition or fixed-size convolutional filters, only work when the number of related time series is small.

Recently, with the development of graph neural networks, Li *et al.* proposed diffusion convolutional recurrent neural networks (DCRNNs), which introduces a spectral-based graph convolutional filter to model the spatial dependencies among roads, to predict the sequential traffic stream [8]. The main differences between our E-GCRNN and DCRNNs are two aspects. Firstly, we use a plain RNN structure (shown in Fig. 2) instead of an encoder-decoder one in DCRNNs to reduce model parameters, as well as to remain model performance. Secondly, DCRNNs assume that the interactions among sequences are time-invariant, which is unsatisfactory in our problem domain, namely evolutionary service invocation tendency. In contrast, we consider the evolution of the interactions among time series in this work, which has been proved effective in remarkably improving prediction accuracy in our targeted knowledge services domain.

We realize that software service usage tendency prediction [11], [30], [33] may also imply some similar characteristics as knowledge service usage tendency prediction. However, compared with the software services ecosystem, the knowledge services ecosystem represents a much larger-scale network with many more nodes and much more complex relationships. This is good for establishing deep learning models, and our findings on knowledge services may be applied to software services, which will be our future work.

6.3 Spectral-based Graph Convolutional Networks

Graph convolutional networks (GCNs) have shown the potential to aggregate information from networks with complicated and irregular structures to utilize the interactions among sequences. The GCNs were first proposed in [5], where spectral-based GCNs, due to their interpretable physical property [42], have drawn significant attention. However, it carries some practical issues.

On the one hand, the graph convolutional operator is time-consuming. To reduce the complexity, Defferrard *et al.* combined Chebyshev expansion [43] with graph signal theory [44] to calculate the graph convolution in a recursive manner [6]. Kipf *et al.* proposed to stack graph convolutional layers to reduce model parameters [7]. These methods are a great basis for our research. However, their ideas consider the Laplacian matrix is static, which cannot solve our problem under the evolutionary assumption.

On the other hand, the application of graph convolutional operators shows a memory bottleneck. In many previous works, GCNs are trained on all training and testing nodes simultaneously, which is impractical in many real-world industries comprising thousands of nodes. Recently, Hamilton *et al.* and Chen *et al.* proposed GraphSAGE and FastGCN to sample related nodes into a mini-batch, to improve efficiency [45], [46]. However, those models only utilize the sampled central nodes to train models, with their neighbors as features for the central nodes, which cannot fully make use of the computational resources. To solve the remained problem, our localized mini-batch training scheme can not only reduce the complexity by separating large graphs into multiple small ones, but also make full use of the GPU memories by considering all included neighbors as the central nodes. Besides, we apply two rounds of sampling in our localized mini-batch training scheme, which fits well for our problem that presents a small world trait. Furthermore, we have proved that our scheme can efficiently reduce the computational complexity as well as be beneficial for model regularization.

7 Conclusions

In recent years, knowledge services have become one of the most important forms for supporting Internet-based innovations. As increasingly more knowledge services are published onto the Internet, how to accurately predict the usage tendency of knowledge services has become a significant topic. However, three unique facts make this problem intractable, including (i) different and complicated temporal dependencies, (ii) intricate and evolutionary spatial dependencies, and (iii) small world trait. To tackle such issues, we have presented evolutionary graph convolutional recurrent neural networks (E-GCRNNs) to predict the long trend of large-scale service invocation with dynamic interactions. E-GCRNNs count on their E-GCGRU component to aggregate spatial information, perceive changing patterns,

and learn temporal dependencies. Furthermore, E-GCRNNs rely on a localized mini-batch training scheme to improve the efficiency and utility of computational resources significantly. Extensive experimental results over real-world datasets have demonstrated that E-GCRNNs outperform baselines, especially when the prediction period becomes longer.

In our future work, we plan to focus on the following four aspects: (i) to study the sensitivity of E-GCRNNs and further improve their prediction accuracy; (ii) to figure out the physical meaning of complex eigenvalues and take causation of sequences into consideration, so that the directed invocation relationship among mashups could be modeled; (iii) to solve the overfitting issue of E-GCRNNs and develop an advanced regularization technique; and (iv) to study the applicability of our E-GCRNNs on predicting the usage tendency of software services.

ACKNOWLEDGEMENTS

This research has been partially supported by the National Key Research and Development Program of China (No. 2018YFB1402500) and the National Nature Science Foundation of China (No.62173199). Yushun Fan is the corresponding author.

REFERENCES

[1] N. Zhang, J. Wang, and Y. Ma, "Mining domain knowledge on service goals from textual service descriptions," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 488–502, 2017.

[2] M. Bano, D. Zowghi, N. Ikram, and M. Niazi, "What makes service oriented requirements engineering challenging? a qualitative study," *IET software*, vol. 8, no. 4, pp. 154–160, 2014.

[3] S. Xiao, J. Yan, C. Li, B. Jin, X. Wang, X. Yang, S. M. Chu, and H. Zha, "On modeling and predicting individual paper citation count over time." in *IJCAI*, 2016, pp. 2676–2682.

[4] X. Liu, J. Yan, S. Xiao, X. Wang, H. Zha, and S. Chu, "On predictive patent valuation: Forecasting patent citations and their types," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[5] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of International Conference on Learning Representations*, 2014.

[6] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 3844–3852.

[7] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[8] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," *arXiv preprint arXiv:1707.01926*, 2017.

[9] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 3634–3640.

[10] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[11] H. Lin, Y. Fan, J. Zhang, and B. Bai, "MSP-RNN: multi-step piecewise recurrent neural network for predicting the tendency of services invocation," *IEEE Transactions on Services Computing*, 2020.

[12] R. Sen, H.-F. Yu, and I. Dhillon, "Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting," *arXiv preprint arXiv:1905.03806*, 2019.

[13] M. Graham, B. Hogan, R. K. Straumann, and A. Medhat, "Uneven geographies of user-generated information: Patterns of increasing informational poverty," *Annals of The Association of American Geographers*, vol. 104, no. 4, pp. 746–764, 2014.

[14] G. E. P. Box and G. M. Jenkins, *Time series analysis forecating and control*. HOlden-Day, 1970.

[15] H. Lütkepohl, *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[16] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[17] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, vol. 2, 2014, pp. 3104–3112.

[18] S. Jachner, G. Van den Boogaart, T. Petzoldt *et al.*, "Statistical methods for the qualitative assessment of dynamic models with time delay (r package qualv)," *Journal of Statistical Software*, vol. 22, no. 8, pp. 1–30, 2007.

[19] C. Tofallis, "A better measure of relative prediction accuracy for model selection and model estimation," *Journal of the Operational Research Society*, vol. 66, no. 8, pp. 1352–1362, 2015.

[20] Y. Wei and M. B. Blake, "Service-oriented computing and cloud computing: challenges and opportunities," *IEEE Internet Computing*, vol. 14, no. 6, pp. 72–75, 2010.

[21] A. Jula, E. Sundararajan, and Z. Othman, "Cloud computing service composition: a systematic literature review," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3809–3824, 2014.

[22] L. Zhang, J. Zhang, J. Fiaidhi, and J. Chang, "Hot topics in cloud computing," *IEEE IT Professional*, vol. 12, no. 5, pp. 17–19, 2010.

[23] A. Vakili and N. J. Navimipour, "Comprehensive and systematic review of the service composition mechanisms in the cloud environments," *Journal of Network and Computer Applications*, vol. 81, pp. 24–36, 2017.

[24] A. Bouguettaya, M. Singh, M. Huhns, Q. Z. Sheng, H. Dong, Q. Yu, A. G. Neiat, S. Mistry, B. Benatallah, B. Medjahed *et al.*, "A service computing manifesto: the next 10 years," *Communications of the ACM*, vol. 60, no. 4, pp. 64–72, 2017.

[25] X. Xu, G. Motta, Z. Tu, H. Xu, Z. Wang, and X. Wang, "A new paradigm of software service engineering in big data and big service era," *Computing*, vol. 100, no. 4, pp. 353–368, 2018.

[26] Z. Wu, J. Yin, S. Deng, J. Wu, Y. Li, and L. Chen, "Modern service industry and crossover services: Development and trends in china," *IEEE Transactions on Services Computing*, vol. 9, pp. 1–1, 04 2015.

[27] W. Tan, J. Zhang, R. Madduri, I. Foster, D. De Roure, and C. Goble, "Servicemap: providing map and GPS assistance to service composition in bioinformatics," in *Proceedings of IEEE International Conference on Services Computing*, 2011, pp. 632–639.

[28] W. Tan, Y. Fan, A. Ghoneim, M. A. Hossain, and S. Dustdar, "From the service-oriented architecture to the web api economy," *IEEE Internet Computing*, vol. 20, no. 4, pp. 64–68, 2016.

[29] K. P. Sycara, M. Paolucci, A. Ankolekar, and N. Srinivasan, "Automated discovery, interaction and composition of semantic web services," *Journal of Web Semantics*, vol. 1, no. 1, pp. 27–46, 2003.

[30] Y. Zhong, Y. Fan, K. Huang, W. Tan, and J. Zhang, "Time-aware service recommendation for mashup creation," *IEEE Transactions on Services Computing*, vol. 8, no. 3, pp. 356–368, 2015.

[31] B. Xia, Y. Fan, C. Wu, K. Huang, W. Tan, J. Zhang, and B. Bai, "Domain-aware service recommendation for service composition," in *2014 IEEE International Conference on Web Services*. IEEE, 2014, pp. 439–446.

[32] Z. Gao, Y. Fan, C. Wu, W. Tan, J. Zhang, Y. Ni, B. Bai, and S. Chen, "SeCo-LDA: Mining service co-occurrence topics for composition recommendation," *IEEE Transactions on Services Computing*, vol. 12, no. 3, pp. 446–459, 2018.

[33] B. Bai, Y. Fan, W. Tan, and J. Zhang, "DLTSR: A deep learning framework for recommendations of long-tail web services," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 73–85, 2020.

[34] F. Xie, L. Chen, Y. Ye, Z. Zheng, and X. Lin, "Factorization machine based service recommendation on heterogeneous information networks," in *2018 IEEE International Conference on Web Services (ICWS)*. IEEE, 2018, pp. 115–122.

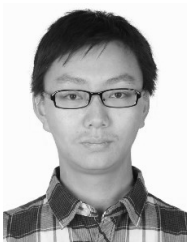
[35] F. Xie, L. Chen, D. Lin, Z. Zheng, and X. Lin, "Personalized service recommendation with mashup group preference in heterogeneous information network," *IEEE Access*, vol. 7, pp. 16 155–16 167, 2019.

[36] S. B. Taieb, A. Sorjamaa, and G. Bontempi, "Multiple-output modeling for multi-step-ahead time series forecasting," *Neurocomputing*, vol. 73, no. 10-12, pp. 1950–1957, 2010.
[37] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, 2019.
[38] X. Cheng, R. Zhang, J. Zhou, and W. Xu, "Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting," in *Proceedings of International Joint Conference on Neural Networks*, 2018, pp. 1–8.
[39] J. Wen, L. Wu, and J. Chai, "Paper citation count prediction based on recurrent neural network with gated recurrent unit," in *2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, 2020.
[40] H. Goel, I. Melnyk, and A. Banerjee, "R2n2: Residual recurrent neural networks for multivariate time series forecasting," *arXiv preprint arXiv:1709.03159*, 2017.
[41] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *Proceedings of International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 95–104.
[42] F. R. Chung and F. C. Graham, *Spectral graph theory*. American Mathematical Soc., 1997, no. 92.
[43] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.
[44] D. I. Shuman, M. Faraji, and P. Vandergheynst, "A multiscale pyramid transform for graph signals," *IEEE Transactions on Signal Processing*, vol. 64, no. 8, pp. 2119–2134, 2016.
[45] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.
[46] J. Chen, T. Ma, and C. Xiao, "FastGCN: fast learning with graph convolutional networks via importance sampling," in *Proceedings of International Conference on Learning Representations*, 2018.

Haozhe Lin received the B.S degree from Central South University, and the Ph.D. degree from the Department of Automation at Tsinghua University, China. He is currently a postdoc with the Department of Automation at Tsinghua University, China. He received the Best Student Paper Award from the 2018 IEEE International Conference on Web Services. His research interests include services computing, time series prediction, and data mining.

Yushun Fan received a Ph.D. degree in control theory and application from Tsinghua University, China, in 1990. He is currently a tenure professor with the Department of Automation, Vice Director of National CIMS Engineering Research Center of China, Director of the System Integration Institute, and Director of the Networking Manufacturing Laboratory, Tsinghua University. He is a member of IFAC TC5.1 and TC 5.2, Vice director of the China Standardization Committee for Automation System and Integration. Editorial member of the International Journal of Computer Integrated Manufacturing. From September 1993 to 1995, he was a visiting scientist, supported by Alexander von Humboldt Stiftung, with the Fraunhofer Institute for Production System and Design Technology (FHG/IPK), Germany. He has authored 10 books in enterprise modeling, workflow technology, intelligent agent, object-oriented complex system analysis, and computer integrated manufacturing. He has published more than 500 research papers in journals and conferences. His research interests include enterprise modeling methods, system integration, modern service science and technology.

Jia Zhang received the M.S. and B.S. degrees in computer science from Nanjing University, China and the Ph.D. degree in computer science from the University of Illinois at Chicago. She is currently Cruse C. and Marjorie F. Calahan Centennial Chair in Engineering, Professor at the Department of Computer Science, Southern Methodist University. Her recent research interests center on data science infrastructure, with a focus on scientific workflows, software discovery, and knowledge graph. She has co-authored one textbook titled "Services Computing" and has published more than 170 refereed journal papers, book chapters, and conference papers. She is currently an associate editor of the IEEE Transactions on Services Computing (TSC). She is a senior member of the IEEE.



Bing Bai received the B.S. and Ph.D. degree in control theory and application from Tsinghua University, China, in 2013 and 2018 respectively, and he is currently a senior researcher with the Cloud and Smart Industries Group, Tencent, Beijing, China. He received the Best Paper Award from the 14th IEEE International Conference on Services Computing (2017). His research interests include data mining and recommender systems.

Zhenghua Xu received the B.Eng. degree from Beijing University of Posts and Telecommunications, China, the M.Phil. in Computer Science degree from The University of Melbourne, Australia, in 2012, and the D.Phil in Computer Science degree from University of Oxford, United Kingdom, in 2018. From 2017 to 2018, he worked as a Research Associate at the Department of Computer Science, University of Oxford, he is now a full Professor at the State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, China. His research focuses on topics within the scope of artificial intelligence and data mining, especially deep learning, medical artificial intelligence, health data mining, and reinforcement learning. He has published more than 20 papers in top AI and database conferences and journals, and is currently serving as the PC member and Area Chair of several top AI conferences, e.g., AAAI, IJCAI, ECAI, etc.

Thomas Lukasiewicz received the Ph.D. degree in computer science from the University of Augsburg, Germany, in 1996, and the Dozent degree (venia docendi) in practical and theoretical computer science from TU Vienna, Austria, in 2001. He is currently a Professor of Computer Science and the Head of the Intelligent Systems Lab at the Department of Computer Science of the University of Oxford, U.K. He is also Turing Fellow at the Alan Turing Institute, London, U.K. He is currently funded by the AXA Research Fund with an AXA Chair in Explainable Artificial Intelligence for Healthcare. His research interests are in artificial intelligence and machine learning, in particular for healthcare applications. He received the Artificial Intelligence journal's Prominent Paper Award 2013 and the 2019 ACM PODS Alberto O. Mendelzon Test-of-Time Award. He is an associate editor for the Journal of Artificial Intelligence Research and the Artificial Intelligence journal.