Learning the Graphical Structure of Electronic Health Records with Graph Convolutional Transformer

Edward Choi,1 Zhen Xu,1 Yujia Li,2 Michael W. Dusenberry,1 Gerardo Flores,1 Emily Xue,1 Andrew M. Dai1
1Google, Mountain View, USA, 2DeepMind, London, UK

Abstract

Effective modeling of electronic health records (EHR) is rapidly becoming an important topic in both academia and industry. A recent study showed that using the graphical structure underlying EHR data (e.g., relationship between diagnoses and treatments) improves the performance of prediction tasks such as heart failure prediction. However, EHR data do not always contain complete structure information. Moreover, when it comes to claims data, structure information is completely unavailable to begin with. Under such circumstances, can we still do better than just treating EHR data as a flat-structured bag-of-features? In this paper, we study the possibility of jointly learning the hidden structure of EHR while performing supervised prediction tasks on EHR data. Specifically, we discuss that Transformer is a suitable basis model to learn the hidden EHR structure, and propose Graph Convolutional Transformer, which uses data statistics to guide the structure learning process. The proposed model consistently outperformed previous approaches empirically, on both synthetic data and publicly available EHR data, for various prediction tasks such as graph reconstruction and readmission prediction, indicating that it can serve as an effective general-purpose representation learning algorithm for EHR data.

1 Introduction

Large medical records collected by electronic healthcare records (EHR) systems in healthcare organizations enabled deep learning methods to show impressive performance in diverse tasks such as predicting diagnosis (Lipton et al. 2015; Choi et al. 2016a; Rajkomar et al. 2018), learning medical concept representations (Che et al. 2015; Choi, Chiu, and Sontag 2016; Choi et al. 2016b; Miotto et al. 2016), and making interpretable predictions (Choi et al. 2016c; Ma et al. 2017). As diverse as they are, one thing shared by all tasks is the fact that, under the hood, some form of neural network is processing EHR data to learn useful patterns from them. To successfully perform any EHR-related task, it is essential to learn effective representations of various EHR features: diagnosis codes, lab values, encounters, and even patients themselves. EHR data are typically stored in a relational database that can be represented as a hierarchical graph depicted in Figure 1. The common approach for processing EHR data with neural networks has been to treat each encounter as an unordered set of features, or in other words, a bag of features. However, the bag of features approach completely disregards the graphical structure that reflects the physician’s decision process. For example, if we treat the encounter in Figure 1 as a bag of features, we will lose the information that Benzonatate was ordered because of Cough, not because of Abdominal pain.

Recently, motivated by this EHR structure, (Choi et al. 2018) proposed MiME, a model architecture that reflects EHR’s encounter structure, specifically the relationships between the diagnosis and its treatment. MiME outperformed various bag of features approaches in prediction tasks such as heart failure diagnosis prediction. Their study, however, naturally raises the question: when the EHR data do not contain structure information (the red edges in Figure 1), can we still do better than bag of features in learning the representation of the data for various prediction tasks? This question emerges in many occasions, since EHR data do not always contain the entire structure information. For example, some dataset might describe which treatment lead to measuring certain lab values, but might not describe the reason diagnosis for ordering that treatment. Moreover, when it comes to claims data, such structure information is completely unavailable to begin with.
To address this question, we propose Graph Convolutional Transformer (GCT), a novel approach to jointly learn the hidden encounter structure while performing various prediction tasks when the structure information is unavailable. Throughout the paper, we make the following contributions:

- To the best of our knowledge, this is the first work to successfully perform the joint learning of the hidden structure of EHR data and a supervised prediction task.
- We propose a novel modification to the Transformer to guide the self-attention to learn the hidden EHR structure using prior knowledge in the form of attention masks and prior conditional probability based regularization. And we empirically show that unguided self-attention alone cannot properly learn the hidden EHR structure.
- GCT outperforms baseline models in all prediction tasks (e.g. graph reconstruction, readmission prediction) for both the synthetic dataset and a publicly available EHR dataset, showing its potential to serve as an effective general-purpose representation learning algorithm for EHR data.

2 Related Work

Although there are works on medical concept embedding, focusing on patients (Che et al. 2015; Miotto et al. 2016; Suresh et al. 2017; Nguyen, Tran, and Venkatesh 2018), visits (Choi et al. 2016b), or codes (Tran et al. 2015; Choi et al. 2017; Shang et al. 2019b), the graphical nature of EHR has not been fully explored yet. (Choi et al. 2018) proposed MiME, which derives the visit representation in a bottom-up fashion according to the encounter structure. For example in Figure 1, MiME first combines the embedding vectors of lab results with the Cardiac EKG embedding, which is then combined with both Abdominal Pain embedding and Chest Pain embedding. Then all diagnosis embeddings are pooled together to derive the final visit embedding. By outperforming various bag-of-features models in various prediction tasks, MiME demonstrated the importance of the structure information of encounter records.

Transformer (Vaswani et al. 2017) was proposed for machine translation. It uses a novel method to process sequence data using only attention (Bahdanau, Cho, and Bengio 2014), and recently showed impressive performance in other tasks such as BERT (i.e. pre-training word representations) (Devlin et al. 2018). There are recent works that use Transformer on medical records (Song et al. 2018; Wang and Xu 2019; Shang et al. 2019a; Li et al. 2019), but they either simply replace RNN with Transformer to handle ICU records, or directly apply BERT learning objective on medical records, and do not utilize the hidden structure of EHR. Graph (convolutional) networks encompass various neural network methods to handle graphs such as molecular structures, social networks, or physical experiments. (Kipf and Welling 2016; Hamilton, Ying, and Leskovec 2017; Battaglia et al. 2018; Xu et al. 2019). In essence, many graph networks can be described as different ways to aggregate a given node’s neighbor information, combine it with the given node, and derive the node’s latent representation (Xu et al. 2019).

Some recent works focused on the connection between the Transformer’s self-attention and graph networks (Battaglia et al. 2018). Graph Attention Networks (Veličković et al. 2018) applied self-attention on top of the adjacency matrix to learn non-static edge weights, and (Wang et al. 2018) used self-attention to capture non-local dependencies in images. Although our work also makes use of self-attention, GCT’s objective is to jointly learn the underlying structure of EHR even when the structure information is missing, and better perform supervised prediction tasks, and ultimately serve as a general-purpose EHR embedding algorithm. In the next section, we outline the graphical nature of EHR, then revisit the connection between Transformer and GCN to motivate the EHR structure learning, after which we describe GCT.

3 Method

Electronic Health Records as a Graph

As depicted in Figure 1, the \( t \)-th visit \( V^{(t)} \) starts with the visit node \( v^{(t)} \) at the top. Beneath the visit node are diagnosis nodes \( d_1^{(t)}, \ldots, d_{d^{(t)}}^{(t)} \) which lead to ordering a set of treatments \( m_1^{(t)}, \ldots, m_{m^{(t)}}^{(t)} \), where \( |d^{(t)}|, |m^{(t)}| \) respectively denote the number of diagnosis and treatment codes in \( V^{(t)} \). Some treatments produce lab results \( r_1^{(t)}, \ldots, r_{r^{(t)}}^{(t)} \), which may be associated with continuous values (e.g. blood pressure) or binary values (e.g. positive/negative allergic reaction). Since we focus on a single encounter in this study \(^1\), we omit the time index \( t \) throughout the paper.

If we assume all features \( d_i, m_i, r_j \) can be represented in the same latent space, then we can view an encounter as a graph consisting of \( |d| + |m| + |r| \) nodes with an adjacency matrix \( A \) that describes the connections between the nodes. We use \( c_i \) as the collective term to refer to any of \( d_i, m_i \), and \( r_j \) for the rest of the paper. Given \( c_i \) and \( A \), we can use graph networks or MiME to derive the visit representation \( v \) and use it for downstream tasks such as heart failure prediction. However, if we do not have the structural information \( A \), which is the case in many EHR data and claims data, we typically use feed-forward networks to derive \( v \), which is essentially summing all node representations \( c_i \)’s and projecting it to some latent space.

Transformer and Graph Networks

Even without the structure information \( A \), it is unreasonable to treat \( V \) as a bag of nodes \( c_i \), because obviously physicians must have made some decisions when making diagnosis and ordering treatments. The question is how to utilize the underlying structure without explicit \( A \). One way to view this problem is to assume that all nodes \( c_i \) in \( V \) are implicitly fully-connected, and try to figure out which connections are stronger than the other as depicted in Figure 2. In this work, as discussed in section 2, we use Transformer to learn the underlying encounter structure. To elaborate, we draw a comparison between two cases:

\(^1\)Note that a sequence aggregator such as RNN or 1-D CNN can convert a sequence of individual encounters \( V^{(0)}, \ldots, V^{(T)} \) to a patient-level representation.

\(^2\)If we bucketize the continuous values of \( r_j \), we can treat \( r_j \) as a discrete feature like \( d_i, m_i \).
where the attention mechanism is replaced with the known, 
(Battaglia et al. 2018). Given this connection, it seems natu-
make the comparison with Transformer clearer.

Conversely, Transformer can be seen
node embeddings
and the attention map softmax( ˜
where ˜
node are fully-connected, and arrive at the right, where
meaningful connections are in thicker edges.

• Case A: We know A, hence we can use Graph Convolu-
tional Networks (GCN). In this work, we use multiple
hidden layers between each convolution, motivated by
(Xu et al. 2019).

\[ C^{(j)} = MLP^{(j)}(\bar{D}^{-1} \tilde{A} C^{(j-1)} W^{(j)}), \]  
where \( \tilde{A} = A + I \), \( \bar{D} \) is the diagonal node degree matrix\(^3\) of \( \tilde{A} \), \( C^{(j)} \) and \( W^{(j)} \) are the node embeddings and the trainable parameters of the \( j \)-th convolution respectively, MLP\(^j\) is a multi-layer perceptron of the \( j \)-th convolution with its own trainable parameters.

• Case B: We do not know A, hence we use Transformer, specifically the encoder with a single-head attention, which can be formulated as

\[ C^{(j)} = MLP^{(j)}(\text{softmax}(\frac{Q^{(j)} K^{(j)T}}{\sqrt{d}})V^{(j)}), \] 
where \( Q^{(j)} = C^{(j-1)} W^{(j)}_Q \), \( K^{(j)} = C^{(j-1)} W^{(j)}_K \), \( V^{(j)} = C^{(j-1)} W^{(j)}_V \), and \( d \) is the column size of \( W^{(j)}_K \cdot W^{(j)}_Q \cdot W^{(j)}_V \), and \( W^{(j)}_Q \), \( W^{(j)}_K \), \( W^{(j)}_V \) are trainable parameters of the \( j \)-th Transformer block\(^4\). Note that positional encoding using sine and cosine functions is not required, since features in an encounter are unordered.

Given Eq. 1 and Eq. 2, we can readily see that there is a correspondence between the normalized adjacency matrix \( \bar{D}^{-1} \tilde{A} \) and the attention map softmax(\( \frac{Q^{(j)} K^{(j)T}}{\sqrt{d}} \)), and between the node embeddings \( C^{(j-1)} W^{(j)} \) and the value vectors \( C^{(j-1)} W^{(j)}_V \). Therefore GCN can be seen as a special case of Transformer, where the attention mechanism is replaced with the known, fixed adjacency matrix. Conversely, Transformer can be seen as a graph embedding algorithm that assumes fully-connected nodes and learns the connection strengths during training (Battaglia et al. 2018). Given this connection, it seems natural to take advantage of Transformer as a base algorithm to learn the underlying structure of visits.

\(^3\)(Xu et al. 2019) does not use the normalizer \( \bar{D}^{-1} \) to improve model expressiveness on multi-set graphs, but we include \( \bar{D}^{-1} \) to make the comparison with Transformer clearer.

\(^4\)Since we use MLP in both GCN and Transformer, the terms \( W^{(j)}_Q \) and \( W^{(j)}_V \) are unnecessary, but we put them to follow the original formulations.

Graph Convolutional Transformer

Although Transformer can potentially learn the hidden en-
counter structure, without a single piece of hint, it must search
the entire attention space to discover meaningful connections
between encounter features. Therefore we propose Graph
Convolutional Transformer (GCT), which, based on data
statistics, restricts the search to the space where it is likely to
contain meaningful attention distribution.

Specifically, we use 1) the characteristic of EHR data and
2) the conditional probabilities between features. First, we
use the fact that some connections are not allowed in the
encounter record. For example, we know that treatment codes
can only be connected to diagnosis codes, but not to other
treatment codes. Based on this observation, we can create a
mask \( M \), which will be used during the attention generation
step. \( M \) has negative infinities where connections are not
allowed, and zeros where connections are allowed.

Conditional probabilities can be useful for determining
potential connections between features. For example, given
chest pain, fever and EKG, without any structure informa-
tion, we do not know which diagnosis is the reason for order-
ing EKG. However, we can calculate from EHR data that
\( p(\text{EKG|chest pain}) \) is typically larger than \( p(\text{EKG|fever}) \), indicating that the connection between the former pair is more
likely than the latter pair. Therefore we propose to use the con-
tditional probabilities calculated from the encounter records
as the guidance for deriving the attention. After calculating
\( p(m|d), p(d|m), p(r|m), \) and \( p(m|r) \) from all encounter records
for all diagnosis codes \( d \), treatment codes \( m \), and lab codes
\( r \), we can create a guiding matrix when given an encounter
record, as depicted by Figure 3. We use \( P \in [0.0, 1.0]^{|d||m||r|} \) to denote the matrix of conditional probabilities of all features, normalized such that each row sums to 1. Note that GCT’s attention softmax(\( \frac{Q^{(j)} K^{(j)T}}{\sqrt{d}} \)), the mask \( M \), and the conditional probabilities \( P \) are of the same size.

Given \( M \) and \( P \), we want to guide GCT to recover the true
table structure as much as possible. But we also want to
allow some room for GCT to learn novel connections that
are helpful for solving given prediction tasks. Therefore GCT
uses the following formulation:

Define \( \hat{A}^{(j)} := \text{softmax}(\frac{Q^{(j)} K^{(j)T}}{\sqrt{d}} + M) \) 

Self-attention:

\[ C^{(j)} = MLP^{(j)}(PC^{(j-1)} W^{(j)}_V) \text{ when } j = 1, \]
\[ C^{(j)} = MLP^{(j)}(\hat{A}^{(j)} C^{(j-1)} W^{(j)}_V) \text{ when } j > 1 \]

Regularization:

\[ L_{\text{reg}}^{(j)} = D_{KL}(P||\hat{A}^{(j)}) \text{ when } j = 1, \]
\[ L_{\text{reg}}^{(j)} = D_{KL}(\hat{A}^{(j)T}||\hat{A}^{(j)}) \text{ when } j > 1 \]
\[ L = L_{\text{pred}} + A \sum_j L_{\text{reg}}^{(j)} \]  

In preliminary experiments, we noticed that attentions were
often uniformly distributed in the first block of Transformer.
Figure 3: Creating the conditional probability matrix $P$ based on an example encounter. The gray cells are masked to zero probability since those connections are not allowed. The green cells are special connections that we know are guaranteed to exist. We assign a pre-defined scalar value (e.g. 1) to the green cells. The white cells are assigned the corresponding conditional probabilities.

This seemed due to Transformer not knowing which connections were worth attending. Therefore we replace the attention mechanism in the first GCT block with the conditional probabilities $P$. The following blocks use the masked self-attention mechanism. However, we do not want GCT to drastically deviate from the informative $P$, but rather gradually improve upon $P$. Therefore, based on the fact that attention is itself a probability distribution, and inspired by Trust Region Policy Optimization (Schulman et al. 2015), we sequentially penalize attention of $j$-th block if it deviates too much from the attention of $j-1$-th block, using KL divergence. As shown by Eq. (4), the regularization terms are summed to the prediction loss term (e.g. negative log-likelihood), and the trade-off is controlled by the coefficient $\lambda$. GCT’s code and datasets are publicly available at https://github.com/Google-Health/records-research.

4 Experiments

Synthetic Encounter Record

MiME (Choi et al. 2018) was evaluated on proprietary EHR data that contained structure information. Unfortunately, to the best of our knowledge, there are no public EHR data that contain structure information (which is the main motivation of this work). To evaluate GCT’s ability to learn EHR structure, we instead generated synthetic data that has a similar structure as EHR data.

The synthetic data has the same visit-diagnosis-treatment-lab results hierarchy as EHR data, and was generated in a top-down fashion. Each level was generated conditioned on the previous level, where the probabilities were modeled with the Pareto distribution, which follows the power law that best captures the long-tailed nature of medical codes. Using 1,000 diagnosis, treatment and lab codes each, we initialized $p(D)$, $p(D|D)$, $p(M|D)$, $p(R|M, D)$ to follow the Pareto distribution, where $D$, $M$, and $R$ respectively denote diagnosis, treatment, and lab random variables. $p(D)$ is used to draw independent diagnosis codes $d_i$, and $p(D|D)$ is used to draw $d_j$ that are likely to co-occur with the previously sampled $d_i$. $p(M|D)$ is used to draw a treatment code $m_j$, given some $d_i$. $p(R|M, D)$ is used to draw a lab code $rk$, given some $m_j$ and $d_i$. A further description of synthetic records is provided in Appendix A. Table 1 summarizes the data statistics.

|                      | Synthetic | eICU |
|----------------------|-----------|------|
| # of encounters      | 50,000    | 41,026 |
| # of diagnosis codes | 1,000     | 3,093 |
| # of treatment codes | 1,000     | 2,132 |
| # of lab codes       | 1,000     | N/A  |
| Avg. # of diagnosis per visit | 7.93 | 7.70 |
| Avg. # of treatment per visit | 14.59 | 5.03 |
| Avg. # of lab per visit | 21.31 | N/A  |

eICU Collaborative Research Dataset

To test GCT on real-world EHR records, we use Philips eICU Collaborative Research Dataset (Pollard et al. 2018). eICU consists of Intensive Care Unit (ICU) records collected from multiple sites in the United States between 2014 and 2015. From the encounter records, medication orders and procedure orders, we extracted diagnosis codes and treatment codes (i.e. medication, procedure codes). Since the data were collected from an ICU, a single encounter can last several days, where the encounter structure evolves over time, rather than being fixed as Figure 1. Therefore we used encounters that lasted less than 24 hours, and removed duplicate codes (i.e. medications administered multiple times). Additionally, we did not use lab results as their values change over time in the ICU (i.e. blood pH level). We leave as future work how to...
Table 2: Graph reconstruction and diagnosis-treatment classification performance. Parentheses denote standard deviations. We report the performance measured in AUROC in Appendix D.

| Model       | Graph reconstruction | Diagnosis-Treatment classification |
|-------------|----------------------|------------------------------------|
|             | Validation AUCPR     | Test AUCPR                         | Validation AUCPR     | Test AUCPR                        |
| GCN         | 1.0 (0.0)            | 1.0 (0.0)                          | 1.0 (0.0)            | 1.0 (0.0)                          |
| GCN<sub>p</sub> | 0.5807 (0.0019)     | 0.5800 (0.0021)                    | 0.8439 (0.0166)     | 0.8443 (0.0214)                    |
| GCN<sub>random</sub> | 0.5644 (0.0018)     | 0.5635 (0.0021)                    | 0.7839 (0.0144)     | 0.7804 (0.0214)                    |
| Shallow     | 0.5443 (0.0015)     | 0.5441 (0.0017)                    | 0.8530 (0.0181)     | 0.8555 (0.0206)                    |
| Deep        | -                   | 0.8210 (0.0096)                    | 0.8198 (0.0046)     |                                 |
| Transformer | 0.5755 (0.0020)     | 0.5752 (0.0015)                    | 0.8329 (0.0282)     | 0.8380 (0.0178)                    |
| GCT         | 0.5972 (0.0027)     | 0.5965 (0.0031)                    | 0.8686 (0.0103)     | 0.8671 (0.0247)                    |

Prediction Tasks

To evaluate the impact of jointly learning the encounter structure, we use prediction tasks based on a single encounter, rather than a sequence of encounters, which was the experiment setup in (Choi et al. 2018). However, GCT can be readily combined with a sequence aggregator such as RNN or 1-D CNN to handle a sequence of encounters, and derive patient representations for patient-level prediction tasks. Specifically, we test the models on the following tasks. Parentheses indicate which dataset is used for each task.

- **Graph reconstruction (Synthetic):** Given an encounter with <i>N</i> features, we train models to learn <i>N</i> feature embeddings. The prevalence is approximately 17%. Further details are provided in Appendix B. This is a multi-label prediction task using the visit representation <i>v</i>.

- **Diagnosis-Treatment classification (Synthetic):** We assign labels to an encounter if there are specific diagnosis (<i>d</i><sub>1</sub> and <i>d</i><sub>2</sub>) and treatment code (<i>m</i><sub>1</sub>) connections. Specifically, we assign “1” if it contains a <i>d</i><sub>1</sub>-<i>m</i><sub>1</sub> connection, and “2” if it contains a <i>d</i><sub>2</sub>-<i>m</i><sub>1</sub> connection. We intentionally made the task difficult so that the models could not achieve a perfect score by just basing their prediction on whether <i>d</i><sub>1</sub>, <i>d</i><sub>2</sub> and <i>m</i><sub>1</sub> exist in an encounter. The prevalence for both labels are approximately 3.3%. Further details are provided in Appendix B. This is a multi-label prediction task using the visit representation <i>v</i>.

- **Masked diagnosis code prediction (Synthetic, eICU):** Given an encounter record, we mask a random diagnosis code <i>d</i> and predict if there is a d<i>1</i>-m<i>1</i> connection. Specifically, we assign “1” if it contains a <i>d</i><sub>1</sub>-m<i>1</i> connection, and “2” if it contains a <i>d</i><sub>2</sub>-m<i>1</i> connection. We intentionally made the task difficult so that the models cannot achieve a perfect score by just basing their prediction on whether <i>d</i><sub>1</sub>, <i>d</i><sub>2</sub> and <i>m</i><sub>1</sub> exist in an encounter. The prevalence for both labels are approximately 3.3%. Further details are provided in Appendix B. This is a multi-label prediction task using the visit representation <i>v</i>.

- **Readmission prediction (eICU):** Given an encounter record, we mask a random diagnosis code <i>d</i>. We train models to learn the embedding of the masked code to predict its identity, i.e. a multi-class prediction. For Shallow and Deep, we use the visit embedding <i>v</i> as a proxy for the masked code representation. The row and the column of the conditional probability matrix <i>P</i> that correspond to the masked diagnosis were also masked to zeroes.

Note that eICU does not contain structure information, e.g. we know *cough* and *acetaminophen* in Figure 1 occur in the same visit, but do not know if *acetaminophen* was prescribed due to *cough*. Table 1 summarizes the data statistics.
Table 3: Masked diagnosis code prediction performance on the two datasets. Parentheses denote standard deviations.

| Model       | Synthetic Validation Accuracy | Synthetic Test Accuracy | eICU Validation Accuracy | eICU Test Accuracy |
|-------------|------------------------------|-------------------------|--------------------------|--------------------|
| GCN         | 0.2862 (0.0048)              | 0.2834 (0.0065)         | -                        | -                  |
| GCN<sub>p</sub> | 0.2002 (0.0024)              | 0.1954 (0.0064)         | 0.7434 (0.0072)          | 0.7432 (0.0086)    |
| GCN<sub>random</sub> | 0.1868 (0.0031)              | 0.1844 (0.0058)         | 0.7129 (0.0044)          | 0.7186 (0.0067)    |
| Shallow     | 0.2084 (0.0043)              | 0.2032 (0.0068)         | 0.7313 (0.0026)          | 0.7364 (0.0017)    |
| Deep        | 0.1958 (0.0043)              | 0.1938 (0.0038)         | 0.7309 (0.0050)          | 0.7344 (0.0043)    |
| Transformer | 0.1969 (0.0045)              | 0.1909 (0.0074)         | 0.7190 (0.0040)          | 0.7170 (0.0061)    |
| GCT         | **0.2220** (0.0033)          | **0.2179** (0.0071)     | **0.7704** (0.0047)      | **0.7704** (0.0039) |

Table 4: Readmission prediction and mortality prediction performance on eICU. Parentheses denote standard deviation. We report the performance measured in AUROC in Appendix D.

| Model       | Readmission prediction Validation AUCPR | Readmission prediction Test AUCPR | Mortality prediction Validation AUCPR | Mortality prediction Test AUCPR |
|-------------|----------------------------------------|-----------------------------------|--------------------------------------|--------------------------------|
| GCN<sub>p</sub> | 0.5121 (0.0154)                          | 0.4987 (0.0105)                   | 0.5808 (0.0331)                      | 0.5647 (0.0201)                |
| GCN<sub>random</sub> | 0.5078 (0.0116)                          | 0.4974 (0.0173)                   | 0.5717 (0.0571)                      | 0.5435 (0.0644)                |
| Shallow     | 0.3704 (0.0123)                          | 0.3509 (0.0144)                   | 0.6041 (0.0253)                      | 0.5795 (0.0258)                |
| Deep        | 0.5219 (0.0182)                          | 0.5050 (0.0126)                   | 0.6119 (0.0213)                      | 0.5924 (0.0121)                |
| Transformer | 0.5104 (0.0159)                          | 0.4999 (0.0127)                   | 0.6069 (0.0291)                      | 0.5931 (0.0211)                |
| GCT         | **0.5313** (0.0124)                      | **0.5244** (0.0142)               | **0.6196** (0.0259)                  | **0.5992** (0.0223)            |

- **Mortality prediction (eICU):** Given an encounter record, we train models to learn the visit embedding $v$ to predict patient death during the ICU admission, i.e., a binary prediction. The prevalence is approximately 7.3%.

For each task, data were randomly divided into train, validation, and test set in 8:1:1 ratio for 5 times, yielding 5 trained models, and we report the average performance. Note that the conditional probability matrix $P$ was calculated only with the training set. Further training details and hyperparameter settings are described in Appendix C.

**Prediction Performance**

Table 2 shows the graph reconstruction performance and the diagnosis-treatment classification performance of all models. Naturally, GCN shows the best performance since it uses the true adjacency matrix $A$. Given that GCN<sub>p</sub> is outperformed only by GCT, we can infer that the conditional probability is indeed indicative of the true structure. GCT, which combines the strength of both GCN<sub>p</sub> and Transformer shows the best performance, besides GCN. It is noteworthy that GCN<sub>random</sub> outperforms Shallow. This seems to indicate that for graph reconstruction, attending to other features, regardless of how accurately the process follows the true structure, is better than individually embedding each feature. Diagnosis-treatment classification, on the other hand, clearly penalizes randomly attending to the features, since GCN<sub>random</sub> shows the worst performance. GCT again shows the best performance.

Table 3 shows the model performance for masked diagnosis prediction for both datasets. GCN could not be evaluated on eICU, since eICU does not have the true structure. However, GCN naturally shows the best performance on the synthetic dataset. Interestingly, Transformer shows comparable performance to GCN<sub>random</sub>, indicating the opposite nature of this task compared to graph reconstruction, where simply each feature attending to other features significantly improved performance. Note that the task performance is significantly higher for eICU than for the synthetic dataset. This is mainly due to eICU having a very skewed diagnosis code distribution. In eICU, more than 80% of encounters have diagnosis codes related to whether the patient has been in an operating room prior to the ICU admission. Therefore randomly masking one of them does not make the prediction task as difficult as for the synthetic dataset.

Table 4 shows the readmission prediction and mortality prediction performance of all models on eICU. As shown by GCT’s superior performance, it is evident that readmission prediction benefits from using the latent encounter structure. Mortality prediction, on the other hand, seems to rely little on the encounter structure, as can be seen from the marginally superior performance of GCT compared to Transformer and Deep. Even when the encounter structure seems unnecessary, however, GCT still outperforms all other models, demonstrating its potential to be used as a general-purpose EHR modeling algorithm. These two experiments indicate that not all prediction tasks require the true encounter structure, and it is our future work to apply GCT to various prediction tasks to evaluate its effectiveness.

**Evaluating the Learned Encounter Structure**

In this section, we analyze the learned structure of both Transformer and GCT. As we know the true structure $A$ of synthetic records, we can evaluate how well both models learned $A$ via self-attention $\tilde{A}$. Since we can view the normalized
Table 5: KL divergence between the normalized true adjacency matrix and the attention map. We also show the entropy of the attention map to indicate the sparseness of the attention distribution. Parentheses denote standard deviations.

| Model    | Graph Reconstruction | Diagnosis-Treatment Classification | Masked Diagnosis Code Prediction |
|----------|----------------------|-------------------------------------|---------------------------------|
|          | KL Divergence        | Entropy                             | KL Divergence                   | Entropy                                      |
| GCNP     | 8.4844 (0.0140)      | 1.5216 (0.0044)                     | 8.4844 (0.0140)                 | 1.5216 (0.0044)                              |
| Transformer | 19.6268 (2.9114)  | 1.7798 (0.1411)                     | 14.3178 (0.2084)                | 1.9281 (0.0368)                              |
| GCT      | 7.6490 (0.0476)      | 1.8302 (0.0135)                     | 8.0363 (0.0305)                 | 1.6003 (0.0244)                              |

Figure 4: Attentions from each self-attention block of Transformer trained for graph reconstruction. Code starting with ‘D’ are diagnosis, ‘T’ treatment, and ‘L’ lab codes. The diagnosis code with the red background D199 is attending to the other features. The red bars indicate the codes that are actually connected to D199, and the blue bars indicate the attention given to each code.

As shown by Table 5, the conditional probabilities are closer to the true structure than what Transformer has learned, in all three tasks. GCT shows similar performance to GCNP in all tasks, and was even able to improve upon P in both graph reconstruction and diagnosis-treatment classification tasks. It is notable that, despite having attentions significantly different from the true structure, Transformer demonstrated strong graph reconstruction performance in Table 2. This again indicates the importance of simply attending to other features in graph reconstruction, which was discussed in Section 4 regarding the performance of GCN_{random}. For the other two tasks, regularizing the models to stay close to P helped GCT outperform Transformer as well as other models.

**Attention Behavior Visualization**

In this section, we show the attention behavior of GCT when trained for graph reconstruction. We randomly chose an encounter record from the test set of the synthetic dataset, which had less than 30 codes in order to enhance readability. To show the attention distribution of a specific code, we chose the first diagnosis code connected to at least one treatment. The red bars indicate the codes that are actually connected to D199, and the blue bars indicate the attention given to each code.

As can be seen from the first self-attention block, GCT starts with a very specific attention distribution, as opposed to Transformer, which can be seen in Figure 5 in Appendix E. The first two attentions given to the placeholder Visit node, and to itself are determined by the scalar value from Figure 3. However, the attentions given to the treatment codes, especially T939 are derived from the conditional probability matrix P. Then in the following self-attention blocks, GCT starts to deviate from P, and the attention distribution becomes more similar to the true adjacency matrix. This nicely shows the benefit of using P as a guide to learning the encounter structure. We further compare and analyze the attention behaviors of both Transformer and GCT under two different contexts, namely graph reconstruction and masked diagnosis code prediction, in Appendix E.
5 Conclusion

Learning effective patterns from raw EHR data is an essential step for improving the performance of many downstream prediction tasks. In this paper, we addressed the issue where the previous state-of-the-art method required the complete encounter structure information, and proposed GCT to capture the underlying encounter structure when the structure information is unknown. Experiments demonstrated that GCT outperformed various baseline models on encounter-based tasks on both synthetic data and a publicly available EHR dataset, demonstrating its potential to serve as a general-purpose EHR modeling algorithm. In the future, we plan to combine GCT with a sequence aggregator (e.g., RNN) to perform patient-level prediction tasks such as heart failure diagnosis prediction or unplanned emergency admission prediction, while working on improving the attention mechanism to learn more medically meaningful patterns.

References

Ba, J.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. arXiv preprint arXiv:1607.06450.
Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational inductive biases, deep learning, and graph networks. arXiv:1806.01261.
Che, Z.; Kale, D.; Li, W.; Bahadori, M. T.; and Liu, Y. 2015. Deep computational phenotyping. In SIGKDD.
Choi, E.; Bahadori, M. T.; Schuetz, A.; Stewart, W. F.; and Sun, J. 2016a. Doctor ai: Predicting clinical events via recurrent neural networks. In Machine Learning for Healthcare Conference.
Choi, E.; Bahadori, M. T.; Searles, E.; Coffey, C.; Thompson, M.; Bost, J.; Tejedor-Sojo, J.; and Sun, J. 2016b. Multi-layer representation learning for medical concepts. In SIGKDD.
Choi, E.; Bahadori, M. T.; Sun, J.; Kulas, J.; Schuetz, A.; and Stewart, W. 2016c. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In NIPS.
Choi, E.; Bahadori, M. T.; Song, L.; Stewart, W. F.; and Sun, J. 2017. Gram: Graph-based attention model for healthcare representation learning. In SIGKDD.
Choi, E.; Xiao, C.; Stewart, W.; and Sun, J. 2018. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. In NIPS.
Choi, Y.; Chiu, C. Y.-I.; and Sontag, D. 2016. Learning low-dimensional representations of medical concepts. AMIA Summits on Translational Science Proceedings.
Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In NIPS.
He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In CVPR.
Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.