Hierarchical Attention Propagation for Healthcare Representation Learning

Muhan Zhang
muhan@wustl.edu
Washington University in St. Louis

Christopher R King*
christopherking@wustl.edu
Washington University in St. Louis

Michael Avidan*
avidanm@wustl.edu
Washington University in St. Louis

Yixin Chen
chen@cse.wustl.edu
Washington University in St. Louis

ABSTRACT

Medical ontologies are widely used to represent and organize medical terminologies. Examples include ICD-9, ICD-10, UMLS etc. The ontologies are often constructed in hierarchical structures, encoding the multi-level subclass relationships among different medical concepts, allowing very fine distinctions between concepts. Medical ontologies provide a great source for incorporating domain knowledge into a healthcare prediction system, which might alleviate the data insufficiency problem and improve predictive performance with rare categories. To incorporate such domain knowledge, Gram, a recent graph attention model, represents a medical concept as a weighted sum of its ancestors' embeddings in the ontology using an attention mechanism. Although showing improved performance, Gram only considers the unordered ancestors of a concept, which does not fully leverage the hierarchy thus having limited expressibility. In this paper, we propose Hierarchical Attention Propagation (HAP), a novel medical ontology embedding model that hierarchically propagate attention across the entire ontology structure, where a medical concept adaptively learns its embedding from all other concepts in the hierarchy instead of only its ancestors. We prove that HAP learns more expressive medical concept embeddings – from any medical concept embedding we are able to fully recover the entire ontology structure. Experimental results on two sequential procedure/diagnosis prediction tasks demonstrate HAP's better embedding quality than Gram and other baselines. Furthermore, we find that it is not always best to use the full ontology. Sometimes using only lower levels of the hierarchy outperforms using all levels.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; • **Applied computing** → **Health informatics**.

KEYWORDS

attention mechanism, network embedding, medical ontology

ACM Reference Format:
Muhan Zhang, Christopher R King, Michael Avidan, and Yixin Chen. 2020. Hierarchical Attention Propagation for Healthcare Representation Learning. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3394486.3403067

1 INTRODUCTION

In recent years, tremendous effort has been put into developing intelligent healthcare prediction systems leveraging the power of big data and machine learning [13, 17]. Existing systems often face one challenge: how to combine power of data-driven algorithms with **domain knowledge** from human experts. A pure data-driven model often requires a huge data volume to achieve a satisfying performance, and typically has a poor performance when predicting cases rarely present in the training data [4]. A proper incorporation of structured domain knowledge, such as the categorization or grouping relationships among different medical concepts might alleviate such problems. For example, disease a and b are both subclasses of disease class c. Then, we may expect a and b to have similar properties, so that even training data for b are rare, we could still learn to predict b with the help of a.

Luckily, there exist many well-established ontologies of medical concepts encoding such structured domain knowledge. Examples include International Classification of Diseases (ICD) [24], Clinical Classifications Software (CCS) [29], Unified Medical Language System (UMLS) [1], Systematized Nomenclature of Medicine-Clinical Terms (SNOMED-CT) [29], and National Drug Code (NDC) etc. These ontologies often have a hierarchical top-down structure, which systematically organizes medical concepts into categories and subcategories of different levels from general to specific. Such hierarchical structures make identifying particular concepts and searching related concepts much easier. The contained structured domain knowledge can also potentially advance the power of healthcare prediction models. For example, nodes (medical concepts) close to each other in an ontology tend to be assigned to similar patients.

To incorporate the domain knowledge within an ontology into a machine learning algorithm, Gram, a graph-based attention model has been proposed recently [5]. Gram learns an embedding for each medical concept by adaptively summing its ancestors' embeddings via an attention mechanism, so that the parent-child relations along a node's paths to the root are encoded into the node's embedding.

*School of Medicine

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403067

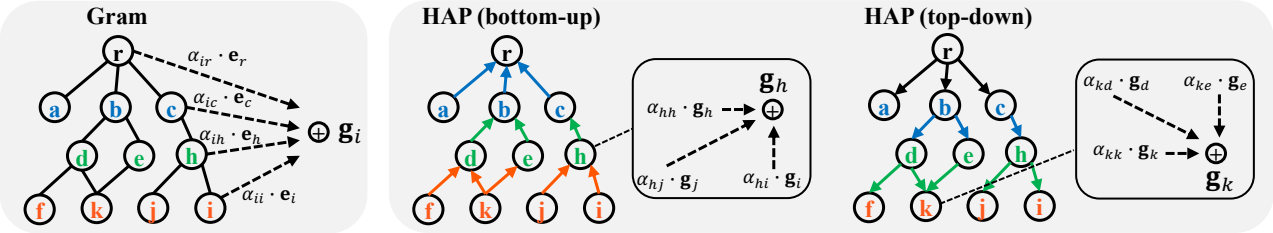

Figure 1: Comparison between Gram and HAP. Gram only considers a node's unordered ancestor set to compute its embedding. HAP hierarchically propagates information across the graph. In the bottom-up round, each parent aggregates information from its children. In the top-down round, each child aggregates information from its parents. The final embedding of each node effectively absorbs information from not only its ancestors, but the entire graph (ancestors, descendants, siblings and others).

Further, two leaf concepts sharing many common ancestors tend to have similar embeddings, which implicitly transfers knowledge between concepts and augments those medical concepts with few occurrences in the training data.

As a first attempt to learn from medical ontologies, Gram has several shortcomings. Firstly, Gram does not consider the order of a node's ancestors – a node's lower ancestors and higher ancestors are symmetrically treated in the attention mechanism, which loses the hierarchical information. Secondly, Gram only considers the ancestor information of a node. It oversimplifies the ontology structure by ignoring the descendants and siblings of a node completely. Both of the shortcomings can be addressed by a more advanced graph attention model considering the full hierarchical structure.

In this paper, we propose the Hierarchical Attention Propagation (HAP) model. HAP does two rounds of knowledge propagation to learn embeddings of medical concepts from the entire ontology: first a bottom-up propagation from leaves to root, and second a top-down propagation from root to all leaves. Figure 1 shows its difference from Gram. In the bottom-up propagation, each node updates its embedding by adaptively combining its children's embeddings using an attention mechanism. In the top-down propagation, nodes adaptively combine their parents' embeddings using the same attention mechanism. Such a two-round propagation is inspired by the Belief Propagation algorithm [25] in graphical models, which is widely used to perform exact inference on tree/polytree models. The two-round knowledge propagation process in HAP effectively distributes a node's attention across the graph, making a node's final embedding no longer only a combination of its ancestors but aggregate information over the entire ontology. We prove that HAP is strictly more expressive than Gram, allowing any node embedding to reconstruct the complete ontology. Experimental results on two sequential procedure/diagnosis prediction tasks reveal HAP's improved performance over Gram and other baselines.

2 PRELIMINARIES

2.1 Notations

We use $c_1, c_2, \ldots, c_{|C|} \in C$ to denote the set of all leaf medical codes of a medical ontology $\mathcal{G}$, and use $c_{|C|+1}, c_{|C|+2}, \ldots, c_{|C|+|C'|} \in C'$ to denote the non-leaf nodes (which are ancestors of the leaf nodes). The ontology $\mathcal{G}$ is expressed as a directed acyclic graph (DAG), where nodes are hierarchically arranged in different levels, with the top level consisting of the single root node and the bottom level consisting of all the leaf nodes C. Examples include the ICD-9, ICD-10 and CCS. We use *knowledge DAG* to refer to the ontology $\mathcal{G}$. In the knowledge DAG, a parent represents a related but more general concept over its children, such as the class of a disease or the category of a procedure. We use $A(i)$ to denote the set of c_i's ancestors and c_i itself. We use $P(i)$ and $C(i)$ to denote the parent set and children set of c_i (both including c_i itself), respectively.

We will consider sequential visit data from patients' electronic health records (EHR) over time. The sequential visit data of a patient is denoted by $V_1, V_2, \ldots, V_T$, where each visit contains a subset of medical codes $V_t \subseteq C$, indicating the procedures/diagnoses that the patient receives at the t^{th} visit. V_t can be represented as a binary vector $\mathbf{x}_t \in \{0,1\}^{|C|}$, where the i^{th} element is 1 if $c_i \in V_t$. For ease of presentation, we will propose our algorithms for a single patient in the rest of the paper. The sequential procedure/diagnosis prediction task is to predict the procedure/diagnosis codes V_{t+1} given the past visits $V_1, V_2, \ldots, V_t$.

2.2 Gram for medical ontology embedding

To leverage the parent-child relationships of the ontology, Gram uses an attention mechanism to adaptively combine a node's ancestors' embeddings as its new embedding. More specifically, in the knowledge DAG, every node c_i is first assigned an initial basic embedding $\mathbf{e}_i$ (can be random embeddings or pretrained embeddings from other sources of information). Then, the final embedding $\mathbf{g}_i$ for c_i is given by a weighted sum of $\{\mathbf{e}_j | j \in A(i)\}$:

$$\mathbf{g}_i = \sum_{j \in A(i)} \alpha_{ij} \mathbf{e}_j, \tag{1}$$

where the weights are computed by the attention mechanism:

$$\alpha_{ij} = \frac{\exp(f(\mathbf{e}_i, \mathbf{e}_j))}{\sum_{k \in A(i)} \exp(f(\mathbf{e}_i, \mathbf{e}_k))}. \tag{2}$$

Here, $f(\mathbf{e}_i, \mathbf{e}_j)$ is a multi-layer perceptron (MLP) which outputs a scalar value representing the raw attention between $\mathbf{e}_i$ and $\mathbf{e}_j$. The Softmax normalizes the attention weights so that they sum to 1.

The new embeddings are then used to represent the medical codes in the sequential visit data, which are fed to a RNN to train a sequential diagnosis prediction model in an end-to-end fashion.

By leveraging the ontology, Gram has improved predictive performance, especially for predicting medical codes less observed in the training data. However, since only the ancestors of a node are considered, the domain knowledge within the ontology is **not fully leveraged**. Further, the ancestors from lower levels and higher levels are treated symmetrically in (1). Thus, the **order** of a node's ancestors is completely **ignored**, which might lose important information about the hierarchy.

3 METHODOLOGY

In this paper, we propose Hierarchical Attention Propagation (HAP), a novel medical ontology embedding method which 1) fully leverages the knowledge DAG, and 2) respects the node ordering within the hierarchy. HAP does two rounds of knowledge propagation to iteratively update each level's nodes's embeddings: first a bottom-up propagation and second a top-down propagation.

Suppose the ontology has L levels of nodes, where level 1 consists of the single root node and level L consists of only leaf medical codes. Level 2, 3, ..., $L-1$ can contain either intermediate category nodes or leaf medical codes (because some medical codes do not have a full L levels of hierarchy). In the beginning, every node embedding $\mathbf{g}_i$ is initialized using a basic embedding $\mathbf{e}_i$. In the bottom-up propagation round, we sequentially update the embeddings of nodes from level $L-1$, level $L-2$, ..., until level 1. For node c_i from level $l-1$, we update its embedding by adaptively combining its current embedding with its children's embeddings from level l using an attention mechanism, given by:

$$\mathbf{g}_i^{(l-1)} = \sum_{j \in C(i)} \alpha_{ij} \mathbf{g}_j^{(l)}, \tag{3}$$

where $\mathbf{g}_j^{(l)} \in \mathbb{R}^{d_g}$ denotes the embedding of node j before we start updating nodes from level $l-1$. We use d_g to denote the embedding size. The attention weight α_{ij} is given by:

$$\alpha_{ij} = \frac{\exp(f(\mathbf{g}_i^{(l)}, \mathbf{g}_j^{(l)}))}{\sum_{k \in C(i)} \exp(f(\mathbf{g}_i^{(l)}, \mathbf{g}_k^{(l)}))}, \tag{4}$$

where $f(\mathbf{g}_i^{(l)}, \mathbf{g}_j^{(l)})$ is an MLP to compute the scalar raw attention between $\mathbf{g}_i^{(l)}$ and $\mathbf{g}_j^{(l)}$. In this work, we use a two layer neural network following [5]:

$$f(\mathbf{g}_i^{(l)}, \mathbf{g}_j^{(l)}) = \mathbf{u}_a^\top \tanh(\mathbf{W}_a \cdot \text{concat}(\mathbf{g}_i^{(l)}, \mathbf{g}_j^{(l)}) + \mathbf{b}_a), \tag{5}$$

where $\mathbf{W}_a \in \mathbb{R}^{d_a \times 2d_g}$ is the weight matrix for the column concatenation of $\mathbf{g}_i^{(l)}$ and $\mathbf{g}_j^{(l)}$, $\mathbf{b}_a \in \mathbb{R}^{d_a}$ is the bias, and $\mathbf{u}_a \in \mathbb{R}^{d_a}$ is the weight vector for generating the scalar raw attention. Here, we use d_a to denote the hidden size of f.

The bottom-up propagation starts from the second-to-last level, and goes all the way up to the root. The updating of nodes from the same level can be performed in parallel, while the updating of an upper level of nodes must wait until all its lower levels have been updated.

Given the embeddings computed by the bottom-up propagation, HAP performs the second round of propagation in a top-down manner. Specifically, we sequentially update the embeddings of nodes from level 2, level 3, ..., until level L. For node c_i from level $l+1$, we update its embedding using a similar attention mechanism by adaptively combining its own embedding with its parents' embeddings from level l:

$$\mathbf{g}_i^{(l+1)} = \sum_{j \in P(i)} \alpha_{ij} \mathbf{g}_j^{(l)}, \tag{6}$$

where $\mathbf{g}_j^{(l)}$ denotes the embedding of node j before we start updating nodes from level $l+1$. The attention weight α_{ij} is:

$$\alpha_{ij} = \frac{\exp(f(\mathbf{g}_i^{(l)}, \mathbf{g}_j^{(l)}))}{\sum_{k \in P(i)} \exp(f(\mathbf{g}_i^{(l)}, \mathbf{g}_k^{(l)}))}, \tag{7}$$

where f has the same form as in Equation (5), with a different set of parameters.

Finally, after the two rounds of propagation, each node has propagated its "attention" across the entire knowledge DAG. Thus, the final embedding of each node effectively absorbs knowledge from not only its ancestors, but also its descendants, siblings, and even some distant nodes. Furthermore, as the propagation order is strictly aligned with the hierarchy, the node ordering information is kept. For instance, in the top-down propagation phase, the ancestors of a node sequentially pass their information down level by level, rather than passing them in one shot as in (1). This enables HAP to discriminate ancestors/descendants from different levels and encode the ordering information.

The final medical code embeddings are used in the sequential procedure/diagnosis prediction tasks. Following [5], we adopt an end-to-end RNN framework. The final embeddings $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{|C|}$ are concatenated column-wise to form an embedding matrix $\mathbf{G} \in \mathbb{R}^{d_g \times |C|}$. Remember each visit record V_t can be represented as a multi-hot vector $\mathbf{x}_t$. To get an embedding vector $\mathbf{v}_t$ for all medical codes in V_t, we multiply $\mathbf{G}$ with $\mathbf{x}_t$ and apply a nonlinear transformation by:

$$\mathbf{v}_t = \tanh(\mathbf{G}\mathbf{x}_t). \tag{8}$$

Then, we sequentially feed $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_T$ into a RNN, which outputs the hidden state for each visit. The hidden state $\mathbf{h}_t$ for $\mathbf{v}_t$ is given by feeding the visit embeddings from all timestamps up to t:

$$\mathbf{h}_t = \text{RNN}(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t). \tag{9}$$

Then the prediction for the next timestamp $t+1$ is given by:

$$\hat{\mathbf{y}}_t = \hat{\mathbf{x}}_{t+1} = \text{Softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b}), \tag{10}$$

where $\mathbf{W} \in \mathbb{R}^{|C| \times d_h}$ and $\mathbf{b} \in \mathbb{R}^{d_h}$ are the weight and bias of the final prediction network, and d_h is the dimension of the RNN's hidden states. The prediction $\hat{\mathbf{y}}_t$ is a vector of dimension $|C|$, indicating the probability of each medical code in visit $t+1$. Note that following Gram, we use Softmax instead of dimension-wise sigmoid to predict multiple medical codes in the next visit as it showed better performance.

We use batch gradient descent to minimize the prediction loss of all timestamps (except timestamp 1). The prediction loss for a single patient is given by:

$$\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) = -\frac{1}{T-1} \sum_{t=1}^{T-1} \left[\mathbf{x}_{t+1}^\top \log(\hat{\mathbf{y}}_t) + (1 - \mathbf{x}_{t+1})^\top \log(1 - \hat{\mathbf{y}}_t) \right]. \tag{11}$$

Following Gram, we not only train the model weights, but also train the basic embeddings $\mathbf{e}_i$. The initialization of the basic embeddings follow the same procedure as in Gram, i.e., using the GloVe [26] embeddings learned from the cooccurrence matrix of augmented codes within the visit records. See section 2.4 of [5] for more details.

Time complexity Since the attention computation is performed on each edge twice, plus the self-attention on each node twice, the time complexity of HAP is $O(|\mathcal{E}| + |C| + |C'|)$ where $\mathcal{E}$ denotes the edge set of the graph. In comparison, Gram computes attention between each leaf node and all its ancestors. Since each leaf can have at most $|C'|$ ancestors, Gram theoretically has a complexity of $O(|C|\cdot|C'|)$. Thus, which of HAP or Gram has a lower complexity depends on the ontology structure used. One specific case (but also a common DAG structure) is tree, where each node has at most one parent. In this case, since $|\mathcal{E}| = |C| + |C'| - 1$, the complexity of HAP becomes $O(|C| + |C'|)$. For Gram, assuming the height of the tree is h. Then the complexity of Gram is $O(|C|\cdot h)$.

4 THEORETICAL ANALYSIS

In this section, we theoretically analyze the properties of our proposed Hierarchical Attention Propagation model, and show it has strictly higher expressive ability than Gram in terms of encoding knowledge DAGs.

Firstly, we use a counter example to show that recovering the knowledge DAG from Gram embeddings is not always possible due to its not encoding the order of the ancestors.

Proposition 1. *With the embeddings $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{|C|}$ computed by Gram (Equation (1)), we **cannot** always reconstruct the knowledge DAG.*

Proof. Consider two DAGs, $A \to B \to C$ and $B \to A \to C$ (illustrated in Figure 2). In both DAGs, the leaf node C gets the same embedding using Equation (1). Thus, from C's embedding we cannot differentiate which is the original DAG. □

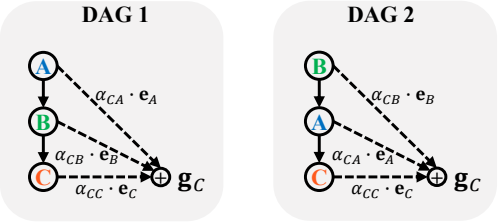

Figure 2: In the two DAGs above, Gram will encode node C into the same embedding, ignoring the order of its ancestors.

This counter-example reveals the limited expressive ability of Gram due to its ignorance of the node ordering and insufficient use of the hierarchy. It indicates that, after running Gram, the ontology information is not fully kept in the medical code embeddings. Thus, the Gram embeddings are essentially a lossy encoding of the knowledge DAG. Next, we study the expressive ability of HAP. In contrast to Gram, we show that from the HAP embeddings we can perfectly reconstruct the knowledge DAG.

Theorem 2. *Assume the basic embeddings $\mathbf{e}_i$ are unique identifiers of the medical concepts they represent. Then, from any embedding $\mathbf{g}_i$ computed by HAP we can always perfectly reconstruct the knowledge DAG, given that every update in (3) and (6) is injective.*

Proof. In the bottom-up propagation, since every update in (3) is injective, from the result $\mathbf{g}_i^{(l-1)}$ of (3) we can reversely infer $\mathbf{g}_i^{(l)}$ and $\{\mathbf{g}_j^{(l)} | j \in C(i)\}$. Consider any node i from level $L-1$. From its updated embedding $\mathbf{g}_i^{(L-1)}$ we can always reconstruct the rooted sub-DAG formed by itself and its children. Now suppose every embedding $\mathbf{g}_j^{(l)}$ from level l injectively encodes the sub-DAG formed by j and all of its descendants. Since Equation (3) is injective, any embedding $\mathbf{g}_i^{(l-1)}$ from level $l-1$ also injectively encodes the sub-DAG formed by i and all of its descendants. Applying structural induction, we get the conclusion that the root embedding at the end of the bottom-up propagation injectively encodes the entire knowledge DAG.

In the top-down propagation, since every update in (6) is also injective, any node embedding will injectively encode the sub-DAG formed by itself and all of its ancestors (including the root) represented by their new embeddings from the bottom-up propagation. Thus, after the top-down propagation, we can reconstruct the entire knowledge DAG from any node embedding (by recovering the root embedding). □

The above theorem demonstrates that the medical code embeddings learned by HAP are strictly more expressive than those learned by Gram. It indicates that HAP embeddings can keep all the knowledge from the ontology – they are an lossless encoding of the knowledge DAG. Note that Theorem 2 requires the update functions in (3) and (6) to be injective. We now prove that the attention mechanism can be an injective mapping. First, we need the following lemma.

Lemma 3. *Suppose a, b, a', b' are positive integers, $a \neq b$, $a' \neq b'$. Then $2^a - 2^b = 2^{a'} - 2^{b'}$ if and only if $a = a'$, $b = b'$.*

Proof. Firstly, it is straightforward that $a = a', b = b' \Rightarrow 2^a - 2^b = 2^{a'} - 2^{b'}$. We now prove that $2^a - 2^b = 2^{a'} - 2^{b'}$ only if $a = a'$, $b = b'$.

Since $a \neq b$ and $a' \neq b'$, if $a > b$ and $a' < b'$ or $a < b$ and $a' > b'$, the equation $2^a - 2^b = 2^{a'} - 2^{b'}$ will not hold. Thus, w.l.o.g., we assume $a > b$ and $a' > b'$. We have

$$2^b(2^{a-b} - 1) = 2^{b'}(2^{a'-b'} - 1). \tag{12}$$

We will prove $a = a'$ and $b = b'$ by contradiction. Assume $b \neq b'$. Without loss of generality, we let $b > b'$. Then,

$$2^{b-b'}(2^{a-b} - 1) = (2^{a'-b'} - 1). \tag{13}$$

In the above equation, LHS is a product of an even number and an odd number which is even, while RHS is an odd number. Thus we have reached a contradiction, which means that $b = b'$. Eliminating b, b' from $2^a - 2^b = 2^{a'} - 2^{b'}$, we have $a = a'$ too. □

Given Lemma 3, now we prove that the update functions in (3) and (6) can indeed be injective.

Theorem 4. *There exists a function f such that the update function*

$$\mathcal{U}(\mathbf{g}_i, \{\mathbf{g}_j | j \in S\}) := \sum_{j \in S \cup i} \alpha_{ij} \mathbf{g}_j, \tag{14}$$

$$\text{where } \alpha_{ij} = \frac{\exp(f(\mathbf{g}_i, \mathbf{g}_j))}{\sum_{k \in S \cup i} \exp(f(\mathbf{g}_i, \mathbf{g}_k))}, \tag{15}$$

is injective w.r.t. its inputs $(\mathbf{g}_i, \{\mathbf{g}_j | j \in S\})$, where $\mathbf{g}_i$ and $\mathbf{g}_j$ are unique rational embeddings of the knowledge DAG nodes and $S \neq \emptyset, i \notin S$.

Proof. Since an ontology has a limited number of concepts, $\mathbf{g}_i$ and $\mathbf{g}_j$ are from a countable universe. Thus, we can construct a function ϕ such that $\phi(i, j)$ maps every ordered $(\mathbf{g}_i, \mathbf{g}_j)$ pair to a unique positive integer. Assume there are n different $\mathbf{g}_i$ choices in the countable universe. Then, $\phi(i, j)$ has n^2 possible outputs.

Having $\phi(i, j)$, we will construct f such that α_{ij} is unique for every combination of $(\mathbf{g}_i, \mathbf{g}_j)$ and $\{\mathbf{g}_k | k \in S \cup i\}$. We let

$$f(\mathbf{g}_i, \mathbf{g}_j) = 2^{\phi(i,j)}. \tag{16}$$

Then, the attention weight α_{ij} becomes

$$\begin{aligned} \alpha_{ij} &= \frac{\exp(2^{\phi(i,j)})}{\sum_{k \in S \cup i} \exp(2^{\phi(i,k)})} \\ &= \frac{1}{1 + \sum_{k \in S \cup i, k \neq j} \exp(2^{\phi(i,k)} - 2^{\phi(i,j)})}. \end{aligned} \tag{17}$$

According to Lemma 3, $2^{\phi(i,k)} - 2^{\phi(i,j)}$ is unique for each ordered tuple of (i, j, k) when $j \neq k$. Considering the linear independence among integer powers of e, the summation in the denominator of the above equation constitutes a unique irrational representation for $(i, j, \{k | k \in S\})$. This means α_{ij} is a unique irrational number for each different (i, j, S) (the reciprocal of an irrational number is also irrational). Under fixed i and S, a unique irrational α_{ij} is associated with each $\mathbf{g}_j$. Besides, α_{ij} for different j are linearly independent using rational coefficients (only multiplying α_{ij} by integer powers of e can we recover other α_{ij}). Thus, the summation $\sum_{j \in S \cup i} \alpha_{ij} \mathbf{g}_j$ is a unique representation for $(i, S, \{j \in S \cup i\})$, which means it is a unique representation for (i, S). Therefore, $\mathcal{U}(\mathbf{g}_i, \{\mathbf{g}_j | j \in S\})$ is an injective function. □

To model such an f, we use an MLP (5) with trainable weights thanks to the universal approximation theory [12]. Note that Theorem 4 requires the node embeddings $\mathbf{g}_i$ to be rational, which can be easily satisfied for the initial basic embeddings. For intermediate embeddings, the update functions in (3) and (6) will output irrational vectors. We can apply another MLP to the updated vectors to map them to rational vectors. In practice, however, this MLP is not necessary due to the finite digits of computer when representing numbers. We also find that the current scheme without another MLP already works well.

5 EXPERIMENTS

5.1 Experimental setup

5.1.1 Prediction tasks: We use two datasets to evaluate the performance of HAP: 1) We conduct a sequential procedure prediction task using the ACTFAST datset, which contains procedure codes of 13.7K patients who received surgery with anesthesia at Barnes-Jewish Hospital between June 2012 and August 2016. Given the history visit records of a patient's ICD9 procedure codes, we aim to predict all the procedure codes she/he will receive in the next visit. 2) We conduct a sequential diagnosis prediction task using the open-source MIMIC-III dataset [15], which contains the medical records of 7.5K intensive care unit (ICU) patients over 11 years. Given the history of a patient's ICD9 diagnosis codes in each visit, we aim to predict all the diagnosis codes she/he will receive in the next visit. A summary of the two datasets are provided in Table 1. Note that our setting is **different** from the setting of the Gram paper [5], the tasks of which were to predict CCS single-level groups of medical codes instead of the exact codes, where the cardinality of the target space was much smaller than ours. In other words, our tasks are more difficult and test a method's exact code prediction ability instead of its group prediction ability. Our setting is also more general – after getting the exact predictions, we can infer the group predictions, while the inverse is not true.

Table 1: Statistics of ACTFAST and MIMIC-III datasets.

Dataset	ACTFAST	MIMIC-III
# of patients	13,658	7,499
# of visits	36,484	19,911
Avg. # of visits per patient	2.67	2.66
# of unique ICD9 codes	2,236	4,893
Avg. # of codes per visit	5.10	13.1
Max # of codes per visit	57	39

For both datasets, we filter out patients with less than two visits.

We calculate *Accuracy@k* for each medical code. For each visit V_t, we get a 1 if the target medical code appears in the top k predictions and 0 otherwise. Then, we report the overall *Accuracy@k* for all medical codes as well as *Accuracy@k* for grouped medical codes. To calculate grouped *Accuracy@k*, we first sort all possible target codes by their counts in the training data. Then, we divide all target codes into four groups $[0, 25]$, $[25, 50]$, $[50, 75]$, $[75, 100]$ with each group's codes having the same summed counts. That is, group $[0, 25]$ contains the the rarest medical codes which constitute 25% of the code counts in the training data. Group $[75, 100]$, on the other hand, contains the most frequent codes which in total constitute 25% of the training codes. We calculate *Accuracy*@10 for ACTFAST and *Accuracy*@20 for MIMIC-III, considering their respective average number of ICD9 codes per visit.

We use the CCS multi-level procedure hierarchy[1] ($L = 5$) as our knowledge DAG for the ACTFAST dataset, and use the CCS multi-level diagnosis hierarchy[2] ($L = 6$) as our knowledge DAG for the MIMIC-III dataset. We randomly split each dataset into the training, validation and test sets using 0.7:0.1:0.2 ratio. We train a model with the training set for 50 epochs, and use the model parameters at the epoch with the smallest validation loss to evaluate on the test set. We repeat each experiment for five times with different random seeds (thus using five different data splits in total). The average test accuracies and standard deviations are reported in the paper.

[1] https://www.hcup-us.ahrq.gov/toolssoftware/ccs/AppendixDMultiPR.txt
[2] https://www.hcup-us.ahrq.gov/toolssoftware/ccs/AppendixCMultiDX.txt

Table 2: Grouped and overall Accuracy@10 of sequential procedure prediction on ACTFAST data.

Model	0-25	25-50	50-75	75-100	Overall
HAP	**0.2135±0.0109**	**0.4993±0.0143**	0.6243±0.0147	0.8001±0.0172	0.5372±0.0045
HAP (lv3)	0.2041±0.0042	0.4973±0.0141	0.6306±0.0123	0.7887±0.0084	0.5331±0.0060
HAP (lv2)	0.2088±0.0075	**0.4993±0.0128**	0.6285±0.0113	**0.8127±0.0134**	**0.5401±0.0051**
Gram	0.1701±0.0065	0.4607±0.0074	0.6137±0.0082	0.8095±0.0122	0.5165±0.0015
Gram (lv3)	0.1933±0.0088	0.4766±0.0141	0.6134±0.0123	0.7743±0.0186	0.5174±0.0056
Gram (lv2)	0.1971±0.0078	0.4987±0.0194	0.6276±0.0028	0.7972±0.0080	0.5331±0.0043
RNN+	0.1994±0.0073	0.4973±0.0102	0.6358±0.0121	0.7967±0.0103	0.5352±0.0031
RNN	0.1936±0.0060	0.4992±0.0066	**0.6409±0.0109**	0.8012±0.0092	0.5367±0.0045
Rollup+	0.1387±0.0063	0.4087±0.0052	0.5743±0.0107	0.7974±0.0106	0.4825±0.0026
Rollup	0.1449±0.0015	0.4123±0.0052	0.5703±0.0148	0.7886±0.0141	0.4817±0.0034

Table 3: Grouped and overall Accuracy@20 of sequential diagnosis prediction on MIMIC-III data.

Model	0-25	25-50	50-75	75-100	Overall
HAP	0.0414±0.0062	**0.2179±0.0103**	0.3813±0.0119	0.7983±0.0171	0.3619±0.0027
HAP (lv3)	**0.0434±0.0033**	0.2119±0.0070	0.3884±0.0102	0.8006±0.0118	0.3633±0.0020
HAP (lv2)	0.0428±0.0062	0.2168±0.0091	**0.3905±0.0128**	0.7970±0.0128	**0.3640±0.0027**
Gram	0.0426±0.0055	0.2042±0.0081	0.3733±0.0118	0.8084±0.0045	0.3591±0.0044
Gram (lv3)	0.0417±0.0058	0.2127±0.0130	0.3858±0.0148	0.7965±0.0167	0.3614±0.0029
Gram (lv2)	0.0399±0.0058	0.2082±0.0124	0.3838±0.0188	0.8031±0.0116	0.3609±0.0031
RNN+	0.0399±0.0064	0.2167±0.0147	0.3806±0.0112	0.8044±0.0160	0.3625±0.0028
RNN	0.0298±0.0029	0.1983±0.0070	0.3847±0.0114	**0.8104±0.0107**	0.3580±0.0028
Rollup+	0.0308±0.0071	0.1786±0.0146	0.3663±0.0164	0.8022±0.0101	0.3465±0.0039
Rollup	0.0314±0.0050	0.1795±0.0076	0.3674±0.0151	0.8056±0.0036	0.3480±0.0036

5.1.2 Models for comparison: We include the following models for comparison.

- HAP: The Hierarchical Attention Propagation model.
- HAP (lv3): The proposed HAP model using only the lowest 3 levels of the hierarchy. That is, the bottom-up propagation stops in level L-2, and the top-down propagation begins from level L-2 too. The assumption is using only lower levels of the hierarchy can sometimes already provide sufficient domain knowledge while reducing the computation complexity.
- HAP (lv2): HAP using the lowest 2 levels of hierarchy.
- Gram: The Graph-based Attention Model described in Preliminaries. A leaf code's embedding $\mathbf{g}_i$ is a weighted sum of the basic embeddings of itself and its ancestors.
- Gram (lv3): The Gram model using only the lowest 3 levels of the hierarchy. A leaf code's embedding $\mathbf{g}_i$ is a weighted sum of the basic embeddings of itself and its ancestors within the lowest 3 levels.
- Gram (lv2): Gram using the lowest 2 levels of hierarchy.
- RNN+: A leaf embedding $\mathbf{g}_i$ takes its own basic embedding without considering the hierarchy. The basic embeddings are initialized using the GloVe embeddings learned from the cooccurrence matrix of leaf concepts, and are trained together with the RNN.
- RNN: The same as RNN+ without pretrained initialization.
- Rollup+: The same as RNN+, except that we replace all leaf concepts with their direct parents in CCS multi-level hierarchy. In other words, all leaf concepts of a parent share the same parent embedding. This is to compare HAP with a common grouping scheme.
- Rollup: The same as Rollup+ without pretrained initialization.

We omit recent state-of-the-art methods such as Dipole [20], KAME [21], since they enhance Gram by using more advanced sequence modeling techniques in the prediction phase (without modifying Gram's graph embedding part), and are orthogonal to our work. The follow-up work of Gram, MiME [7], is also not compared since it leverages the additional hierarchical structure between diagnosis and procedure which is not available in our data. All models are implemented with Theano and optimized using Adadelta with a mini-batch size of 100 patients. All models use the same GRU-based RNN with a hidden size $d_h = 400$ and a dropout rate of 0.4. The embedding size d_g is 400 for all models. For HAP and Gram, the attention weights $\mathbf{W}_a, \mathbf{b}_a, \mathbf{u}_a$ have a dimension $d_a = 100$.

5.2 Prediction performance

We present the sequential procedure prediction results on ACTFAST in Table 2, and present the sequential diagnosis prediction results on MIMIC-III in Table 3. In both tasks, HAP and its variants show advantages over other models. The gain is greater for

less frequent codes. For example, in MIMIC-III, the HAP (lv3) is nearly 50% more accurate than the RNN baseline in the 0-25 percentile range, indicating that learning embeddings from ontology structures benefits the prediction of rare codes.

Both HAP and Gram leverage the ontology structure, yet Gram seems to be more sensitive to the number of hierarchy levels used. By varying the number of hierarchy levels, Gram shows larger performance variances. Especially for ACTFAST, we observe that Gram has a large drop of accuracies compared to Gram (lv3) and Gram (lv2). This indicates that Gram is not robust to the number of hierarchy levels used and requires carefully selecting this number. In contrast, HAP shows robust performance across different numbers of hierarchy levels used. One possible explanation is that the use of higher levels of hierarchy makes a Gram's leaf embedding confused about its parent and grandparent (since the attention scheme in Gram is order-unaware), which on the contrary adds ambiguity to the leaf embedding. On the other hand, HAP is order-aware, which means adding more ancestors of a leaf will still respect their ordering without confusing the leaf embedding.

We also find that HAP and Gram do not always perform the best using the full hierarchy. This might imply that lower levels of the hierarchy **provide the most important domain knowledge** for predicting leaf codes, which is reasonable since higher levels of the hierarchy correspond to very broad categories of procedures or diseases that might not be as useful as those fine categories in lower levels. For example, knowing that two diagnoses both belong to the "Infection" category provides very limited information. Knowing they are both related to "Viral Infection" provides a little more information. And knowing that they both belong to "HIV Infection" provides the most information.

We did not observe the huge advantage of Gram as in the task of predicting CCS single-level groups [5]. This might indicate that Gram's grouping effect is more suitable for predicting coarse medical concept groups than predicting exact codes, while HAP keeps enough distinctness of individual leaf codes during the embedding learning.

6 RELATED WORK

Attention mechanism is a widely used framework in neural networks to adaptively learn the importance of each component w.r.t. the target component. It has been successfully used in computer vision [33], machine translation [31], speech recognition [8], semi-supervised node classification [32] etc. There has been work that applies attention mechanism to healthcare problems [5, 6, 20, 21]. Our work is closely related to the Gram model [5] by generalizing the attention mechanism used to aggregate ancestor information to a hierarchical mulit-level propagation framework that learns from the entire DAG.

Our method is related to recent graph representation learning works such as network embedding [10, 27, 30] and graph neural networks (GNNs) [2, 9, 16, 18, 28]. Network embedding methods learn transductive node embeddings through optimizing some loss functions such that nearby/connected nodes have similar embeddings. Network embedding is unsupervised. The learned node embeddings can be used in downstream node classification or link prediction tasks, while the embedding training is separated from the downstream tasks.

Graph neural networks (GNNs) iteratively pass messages between a node and its neighbors to extract multiple-hop local substructure features for nodes. GNNs are typically supervised. The extracted node features are used directly in later tasks so that the supervision signals can train parameters in the message passing layers in an end-to-end fashion and guide GNNs on how to extract node features (embeddings). GNNs have gained great popularity in recent years, achieving state-of-the-art performance on semi-supervised node classification [16], graph classification [35], network embedding [11], and link prediction [34], etc. Despite the success, little work has been done on applying GNNs to healthcare. Our model can be seen as combining an attention-based graph convolution layer [32] with a particular message passing order respecting the multi-level hierarchy of the medical ontology, where the design is inspired by the Belief Propagation algorithm [25] and ensures incorporating structural information of the entire knowledge DAG. We did not compare with the extensive literature of GNNs, since exitsing GNNs are mainly designed to assign similar embeddings to nearby nodes in an undirected graph by symmetrically passing messages between nodes, without really considering the hierarchical structure of medical ontologies.

One noteworthy line of related research is learning hyperbolic graph embeddings [3, 19, 22, 23]. However, these works only consider the implicit hierarchical structure of real-world graphs to learn more efficient embeddings, without explicitly leveraging an ontology. For example, our used medical ontologies have clearly defined root, leaves, and different levels of intermediate nodes, while such hierarchies are assumed to be latent in the graphs of hyperbolic embeddings. There is also recent work studying GNNs for trees/DAGs [14, 36] with a focus on generating DAGs instead of learning embeddings of DAG nodes.

7 CONCLUSION

In this paper, we have proposed Hierarchical Attention Propagation (HAP), a graph attention-based method to learn medical concept embeddings based on medical ontologies. HAP propagates information hierarchically across the graph. The bottom-up propagation sends all leaf information to the root node, and the top-down propagation propagates entire structure information back to leaves. HAP learns highly expressive embeddings by learning from the full ontology hierarchy, addressing previous work Gram's limited expressibility. We have theoretically proved that from any HAP embedding we can recover the entire knowledge DAG, which strictly outperforms Gram embedding's expressibility. Experiments on two sequential procedure/diagnosis prediction tasks verified HAP's superior healthcare representation learning and prediction performance.

ACKNOWLEDGMENTS

The work is supported in part by the National Science Foundation under award numbers III-1526012 and SCH-1622678, and by the National Institute of Health under award number 1R21HS024581.

REFERENCES

[1] Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research* 32, suppl_1 (2004),

D267–D270.
[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).
[3] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*. 4869–4880.
[4] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. 301–318.
[5] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. 2017. GRAM: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 787–795.
[6] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*. 3504–3512.
[7] Edward Choi, Cao Xiao, Walter Stewart, and Jimeng Sun. 2018. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. In *Advances in Neural Information Processing Systems*. 4547–4557.
[8] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*. 577–585.
[9] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3837–3845.
[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.
[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.
[12] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural networks* 2, 5 (1989), 359–366.
[13] Fei Jiang, Yong Jiang, Hui Zhi, Yi Dong, Hao Li, Sufeng Ma, Yilong Wang, Qiang Dong, Haipeng Shen, and Yongjun Wang. 2017. Artificial intelligence in healthcare: past, present and future. *Stroke and vascular neurology* 2, 4 (2017), 230–243.
[14] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction Tree Variational Autoencoder for Molecular Graph Generation. In *Proceedings of the 35th International Conference on Machine Learning*. 2323–2332.
[15] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3 (2016), 160035.
[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[17] Hian Chye Koh, Gerald Tan, et al. 2011. Data mining applications in healthcare. *Journal of healthcare information management* 19, 2 (2011), 65.
[18] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493* (2015).
[19] Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*. 8228–8239.
[20] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. 2017. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 1903–1911.
[21] Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. 2018. Kame: Knowledge-based attention model for diagnosis prediction in healthcare. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 743–752.
[22] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*. 6338–6347.
[23] Maximilian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *arXiv preprint arXiv:1806.03417* (2018).
[24] World Health Organization. 2004. *International statistical classification of diseases and related health problems*. Vol. 1. World Health Organization.
[25] Judea Pearl. 1982. *Reverend Bayes on inference engines: A distributed hierarchical approach*.
[26] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.
[28] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20, 1 (2009), 61–80.
[29] Michael Q Stearns, Colin Price, Kent A Spackman, and Amy Y Wang. 2001. SNOMED clinical terms: overview of the development process and project status.. In *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 662.
[30] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.
[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).
[33] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.
[34] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*. 5165–5175.
[35] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. 2018. An End-to-End Deep Learning Architecture for Graph Classification. In *AAAI*. 4438–4445.
[36] Muhan Zhang, Shali Jiang, Zhicheng Cui, Roman Garnett, and Yixin Chen. 2019. D-VAE: A variational autoencoder for directed acyclic graphs. In *Advances in Neural Information Processing Systems*. 1586–1598.