T- Hop: Tensor representation of paths in graph convolutional networks

Abdulrahman Ibraheem
rahan.ihbraheem1@outlook.com

April 12, 2022

1 Introduction

This draft describes a method for capturing path information in graphs, for use in graph convolutional networks (GCN). Let $G = (V, E)$ represent a graph, where $V = \{v_1, ..., v_n\}$ and $E = \{e_1, ..., e_m\}$ are the nodes and edges of $G$, as usual. The adjacency matrix of $G$ is $A$. Further, $A^L$ denotes the powered adjacency matrix of $G$, while $A^L_{i,j}$ represents the entry on the $i$-th row and $j$-th column of $A^L$. The entry $A^L_{i,j}$ corresponds to the number of paths of length $L$ between node $v_i$ and node $v_j$ in $G$. While the seminal vanilla GCN of Kipf and Welling [1] employs only matrix $A$ as its operator, more recent variants of the GCN, such as MixHop [2] and Variable Power Networks (VPN) [3] employ $A^L$, in their models. The hope of this draft is to improve on the aforementioned methods, by incorporating more information into $A^L$, in hopes that the additional information might lead to higher accuracies in downstream tasks utilizing the resulting representation.

Towards this, let us consider two arbitrary nodes, $v_i$ and $v_j$ in graph $G$. Let $B^L_{i,j,k}$ be the number of paths of length $L$ between $v_i$ and $v_j$ that contain $v_k$. Clearly, we can arrange the values, $B^L_{i,j,k}$, in an $n \times n \times n$ 3-d tensor denoted $B^L$, such that the entry on the $i$-th row, $j$-th column and $k$-th depth of $B^L$ is $B^L_{i,j,k}$. Using $B^L_{i,j,k}$, we now define a 3-d tensor, $T^L$. Where $T^L_{i,j,k}$ is the entry on the $i$-th row, $j$-th column and $k$-th depth of $T^L$, I define $T^L_{i,j,k}$ as follows:

$$T^L_{i,j,k} = \frac{B^L_{i,j,k}}{(L + 1)}$$ (1)

From the above definition, it should be noted that $T^L$ is an $n \times n \times n$ tensor, where $n$ is the number of nodes in the graph. Clearly, working with such a large matrix is computationally demanding. To enhance tractability, I propose applying dimensionality reduction along the depth axis of $T^L$. To underpin the idea, let us fix the row and column indices (i.e $i$ and $j$) of $T^L$, while leaving the depth index to vary. For each pair of indices, $(i, j)$, this results in an $n$-dimensional vector which stretches along the
depth axis of $\mathcal{T}^L$. Let us denote this vector as $t_{ij}^L = \mathcal{T}^L_{ij}$: We would like to compute a dimensionality reduction map, $f : \mathbb{R}^n \to \mathbb{R}^d$, with $d \ll n$, that sends each $t_{ij}^L$ to a $d$-dimensional space. If we apply this map to all $n$-dimensional vectors corresponding to all $(i,j)$ positions in $\mathcal{T}^L$, we get a $n \times n \times d$ tensor, which we denote $\hat{\mathcal{T}}^L$.

Indeed, it is quite interesting to observe that if we choose the dimensionality reduction map to be the summation operation, $f_{\text{sum}} : \mathbb{R}^n \to \mathbb{R}$, which simply outputs the sum of all components of its input vector, then we would obtain the powered adjacency matrix, $A^L$, when we apply $f_{\text{sum}}$ to $\mathcal{T}^L$. I elaborate on this formally below, beginning with the following definition:

**Definition 1.1 (Cardinality of multiset $\mathcal{P}^L$).** Let $G$ be a graph of nodes, $V = \{v_1, \ldots, v_n\}$, and edges. Let $v_i$ and $v_j$ be any two arbitrary nodes in $G$, and let $A^L_{ij}$ be the number of simple paths of length $L$ between $v_i$ and $v_j$. Let $P^L_q = \{v^q_1, v^q_2, \ldots, v^q_{L+1}\}$ denote the $q$-th simple path of length $L$ between $v_i$ and $v_j$, where $v^q_k$ is the $k$-th node in the $q$-th path, $P^L_q$. Let $\mathcal{P}^L = \{P^L_1, P^L_2, \ldots, P^L_{A^L_{ij}}\} = \{v^1_1, v^1_2, \ldots, v^1_{L+1}, v^2_1, v^2_2, \ldots, v^2_{L+1}, \ldots, v^A^L_{ij}, v^A^L_{ij}, \ldots, v^A^L_{ij}\}$ be a multiset containing all the simple paths of length $L$ between $v_i$ and $v_j$. Then, the cardinality of multiset $\mathcal{P}^L$ is defined as the number of nodes in $\mathcal{P}^L$, counting multiplicities of nodes.

Based on the preceding definition, the following is a fact:

**Fact 1.2 (Cardinality of multiset $\mathcal{P}^L$ equals $\sum_k B^L_{i,j,k}$).** The cardinality of multiset $\mathcal{P}^L$ defined in Definition 1.1 above is equal to $\sum_k B^L_{i,j,k}$.

**Proof.** The proof is best sketched with an example. As an example, let us use the graph $G$ of five nodes depicted in Figure 1. Without loss of generality, let us consider all simple paths of length $L = 3$ between two arbitrary nodes, $v_1$ and $v_5$ in the graph. From the graph, we see that there are 2 simple paths of length 3 between $v_1$ and $v_5$, so that $A^3_{1,5} = 2$. These paths are $P^3_1 = \{v_1, v_2, v_4, v_5\}$ and $P^3_2 = \{v_1, v_3, v_4, v_5\}$. Hence, we may write $\mathcal{P}^3 = \{P^3_1, P^3_2\} = \{v_1, v_2, v_4, v_5, v_1, v_3, v_4, v_5\}$. Upon sorting $\mathcal{P}^3$, we now have $\mathcal{P}^3 = \{v_1, v_1, v_2, v_3, v_4, v_4, v_5, v_5\}$. Now, given any node, $v_k$ in $G$, when we count the multiplicity of $v_k$ in the sorted version of $\mathcal{P}^3$, we see it corresponds to the number of simple paths of length 3 between $v_1$ and $v_5$ that contain $v_k$. For example, we see clearly that node $v_4$ has multiplicity of 2, because it is contained in two different paths of length 3 between $v_1$ and $v_5$, whereas node $v_2$ has multiplicity of 1, because it is contained in a single path of length 3 between $v_1$ and $v_5$. Generalizing this observation, we see that, if $\mathcal{P}^L$ is the multiset of nodes that constitute the simple paths of length $L$ between any two arbitrary nodes, $v_i$ and $v_j$, then for any node, $v_k \in \mathcal{P}^L$, the multiplicity of $v_k$ in $\mathcal{P}^L$ corresponds to the number of paths of length $L$ between $v_i$ and $v_j$ that contain $v_k$, which in turn, by definition, is equal to $B^L_{i,j,k}$. **In summary, for any node, $v_k \in \mathcal{P}^L$, the multiplicity of $v_k$ in $\mathcal{P}^L$ equals $B^L_{i,j,k}$.** Based on this, we now consider the quantity $\sum_k B^L_{i,j,k}$. It should be clear that the quantity $\sum_k B^L_{i,j,k}$ simply equals the sum of multiplicities of all nodes in $\mathcal{P}^L$, which, in turn, equals the cardinality of $\mathcal{P}^L$.  

Next, using Fact 1.2, we have the following proposition:
Proposition 1.3 \((f_{\text{sum}} \text{ recovers } A^L \text{ from } T^L)\). Let \(f_{\text{sum}} : \mathbb{R}^n \to \mathbb{R}\) be the function that takes a vector \(u \in \mathbb{R}^n\) as input and returns as output the summation of all components of \(u\). Then, with \(t_{ij}^L\) denoting the \(n\)-dimensional vector that stretches along the depth-axis of the 3-d tensor, \(T^L\), at a given \(i\)-th row, \(j\)-th column position of \(T^L\), we have that \(f_{\text{sum}}(t_{ij}^L) = A_{ij}^L\).

Proof. To proceed, from Definition 1.1 above, we recall the meaning of the multiset \(P^L = \{P^L_1, P^L_2, ..., P^L_{A_{ij}}\} = \{v_1^1, v_2^1, ..., v_{L+1}^1, v_1^2, v_2^2, ..., v_{L+1}^2, ..., v_1^{A_{ij}}, v_2^{A_{ij}}, ..., v_{L+1}^{A_{ij}}\}\); we also bring to mind the definition of the multiset’s cardinality given therein. In particular, the number of paths in \(P^L\) is \(A_{ij}\) and each path contains \(L + 1\) nodes, so that the cardinality of \(P^L\) is equal to \((L + 1)A_{ij}^L\). Hence, we have \(|P^L| = (L + 1)A_{ij}^L\). But, we already know from Fact 1.2 above that \(|P^L| = \sum_k B_{i,j,k}^L\). Hence, we have \(\sum_k B_{i,j,k}^L = (L + 1)A_{ij}^L\), implying \(\sum_k \frac{B_{i,j,k}^L}{(L + 1)} = A_{ij}^L\). Further, by definition, we know \(\frac{B_{i,j,k}^L}{(L + 1)} = T_{i,j,k}^L\). Thus, \(\sum_k T_{i,j,k}^L = A_{ij}^L\). Now, it is clear that \(\sum_k T_{i,j,k}^L\) is tantamount to applying \(f_{\text{sum}}\) to the vector \(t_{ij}^L = T_{i,j,k}^L\), which completes the proof.

In the above, the application of \(f_{\text{sum}}\) to the a \(n \times n \times n\) 3-d tensor, \(T^L\), to obtain the \(n \times n\) 2-d matrix, \(A^L\), can be viewed as a dimensionality reduction process. However, it seems that applying the summation operation, embodied in \(f_{\text{sum}}\), to \(T^L\) might result in too much loss of the information encoded in \(T^L\). It is only natural to hope that a softer approach which compresses the original information contained in \(T^L\) into a \(n \times n \times d\) tensor, with \(d \ll n\), might result in lesser loss of information, and might thereby lead to improved accuracies in downstream tasks. This is why a key proposal of this draft is to explore the use of dimensionality techniques to compress the information encoded in \(T^L\) into a \(n \times n \times d\) tensor, namely \(\hat{T}^L\). For dimensionality reduction, one could explore the variational autoencoder and its variants [4] [5].

After dimensionality reduction, we take the resulting tensor, \(\hat{T}^L\), and employ it within a larger GCN framework such as MixHop. In MixHop, the \((l + 1)\)-th layer of GCN performs the following operation:

![Figure 1: An illustrational graph of five nodes](image-url)
\[ H^{l+1} = \bigoplus_{L \in P} \sigma(A^L H^l W^L_L) \]  

In the preceding equation, \( P \) is an index set, which acts as an hyperparameter of the model. For example, one could have \( P = \{0, 1, 2\} \). Further, \( H^l \in \mathbb{R}^{n \times s_l} \) represents the input to the \( l \)-th GCN layer, with \( n \) being the number of nodes in the underlying graph. \( A^L \in \mathbb{R}^{n \times n} \) denotes the \( L \)-power adjacency matrix, \( W^L_L \in \mathbb{R}^{s_l \times \hat{s}_l + 1} \) is a learnable parameter of the model, \( \sigma(.) \) denotes an activation function, and \( H^{l+1} \in \mathbb{R}^{n \times \hat{s}_l + 1} \) denotes the output from the \( l \)-th GCN layer. Also, \( \bigoplus \) denotes concatenation of \( \sigma(A^L H^l W^L_L) \in \mathbb{R}^{n \times \hat{s}_l + 1} \) along the column dimension. Due to this concatenation operation, we have \( s_{l+1} = |P|\hat{s}_{l+1} \), where \( |P| \) is the cardinality of set \( P \).

I now turn to define my proposed **Tensor Hop** (T-Hop) model. To proceed, let \( \hat{T}^L_{i:j,k} \) denote the \( n \times n \) matrix obtained at the \( k \)-depth of \( \hat{T}^L \). My proposal is to use \( \hat{T}^L_{i:j,k} \) in place of \( A^L \) in the MixHop layer of Equation 2 above. To formalize this, I define my proposed T-Hop layer as:

\[ H^{l+1} = \bigoplus_{L \in P} \bigoplus_{k=1}^d \sigma(\hat{T}^L_{i:j,k} H^l W^L_L) \]  

The notation in Equation 3 above is similar to that in Equation 2, except that \( \bigoplus \) denotes a generic aggregation operation and \( d \) denotes the number of depth in the \( n \times n \times d \) tensor, \( \hat{T}^L \). An example aggregation operation that could be used for implementing \( \bigoplus \) is the element-wise averaging operation of two or more matrices. Comparing the proposed T-Hop layer in Equation 3 with the MixHop layer of Equation 2, we see that there is a very high degree of correspondence between the two of them. In particular, in Equation 3, the matrix \( \hat{T}^L_{i:j,k} \in \mathbb{R}^{n \times n} \) plays the role which \( A^L \in \mathbb{R}^{n \times n} \) plays in Equation 2. Moreover, the same set of parameters, \( W^L_L \), are shared for all \( k = 0, 1, ..., d \) in the computation of \( \sigma(\hat{T}^L_{i:j,k} H^l W^L_L) \) in Equation 3. Notice that, as long as \( d \ll n \) is satisfied, this parameter sharing scheme makes the computational complexity of the proposed T-Hop layer to be on the order of that of the MixHop layer. Moreover, the memory footprint of the proposed T-Hop layer is also on the order of that required for MixHop, as long as \( d \ll n \) is satisfied.

References

[1] Kipf, T., and Welling, M. Semi-supervised classification with graph convolutional networks, ICLR, 2017.

[2] Abu-El-Haija S., Perozzi B., Kapoor A., Harutyunyan H., Alipourfard N., Lerman K., Ver Steeg G., and Galstyan A. MixHop: Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing, ICML, 2019.

[3] Jin, M., Chang, H., Zhu, W., and Sojoudi, S. Power up! Robust Graph Convolutional Network via Graph Powering. AAAI, 2021.
[4] Hinton, G. E.; Salakhutdinov, R.R. Reducing the Dimensionality of Data with Neural Networks”. Science. 313 (5786): 504–507, July, 2006.

[5] Kingma D., and Welling M., Auto-Encoding Variational Bayes. ICLR, 2014