A Proof of Theorem 1

In order to prove this theorem, we first need the following lemma showing the existence of partially symmetric CP decompositions.

Lemma 1. *Any partially symmetric tensor admits a partially symmetric CP decomposition.*

Proof. We show the results for 3-rd order tensors that are partially symmetric w.r.t. their two first modes. The proof can be straightforwardly extended to tensors of arbitrary order that are partially symmetric w.r.t. any subset of modes.

Let $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{m\times m\times n}$ be partially symmetric w.r.t. modes 1 and 2. We have that $\boldsymbol{\mathcal{T}}_{:,:,i}$ is a symmetric tensor for each $i \in [n]$. By Lemma 4.2 in [11], each tensor $\boldsymbol{\mathcal{T}}_{:,:,i}$ admits a symmetric CP decomposition:

$$\boldsymbol{\mathcal{T}}_{:,:,i} = [\![\mathbf{A}^{(i)}, \mathbf{A}^{(i)}]\!], \;\; i \in [n]$$

where $\mathbf{A}^{(i)} \in \mathbb{R}^{m\times R_i}$ and R_i is the symmetric CP rank of $\boldsymbol{\mathcal{T}}_{:,:,i}$.

By defining $R = \sum_{i=1}^{n} R_i$ and $\mathbf{A} = [\mathbf{A}^{(1)}\ \mathbf{A}^{(2)}\ \cdots\ \mathbf{A}^{(n)}] \in \mathbb{R}^{m\times R}$, one can easily check that $\boldsymbol{\mathcal{T}}$ admits the partially symmetric CP decomposition $\boldsymbol{\mathcal{T}} = [\![\mathbf{A}, \mathbf{A}, \boldsymbol{\Delta}]\!]$ where $\boldsymbol{\Delta} \in \mathbb{R}^{m\times R}$ is defined by

$$\Delta_{i,r} = \begin{cases} 1 & \text{if } R_1 + \cdots + R_{i-1} < r \leq R_1 + \cdots + R_i \\ 0 & \text{otherwise.} \end{cases}$$

□

We can now prove Theorem 1.

Theorem. *The function computed by a CP layer (Eq.* (1)*) is permutation-invariant. In addition, any* permutation-invariant *multilinear polynomial* $f : (\mathbb{R}^F)^k \to \mathbb{R}^d$ *can be computed by a CP layer (with linear activation function).*

Proof. The fact that the function computed by a CP layer is permutation-invariant directly follows from the definition of the CP layer and the fact that the Hadamard product is commutative.

We now show the second part of the theorem. Let $f : (\mathbb{R}^F)^k \to \mathbb{R}^d$ be a permutation-invariant multilinear polynomial map. Then, there exists permutation-invariant univariate multilinear polynomials $g_{i,j_1,\cdots,j_k}$ for $j_1, \cdots, j_k \in [F]$ and $i \in [d]$ such that

$$f(\mathbf{x}_1, \cdots, \mathbf{x}_k)_i = \sum_{j_1=1}^{d} \cdots \sum_{j_k=1}^{d} g_{i,j_1,\cdots,j_k}((\mathbf{x}_1)_{j_1}, \cdots, (\mathbf{x}_k)_{j_k}).$$

Moreover, by definition, each such polynomial satisfies

$$g_{i,j_1,\cdots,j_k}(a_1, a_2, \cdots, a_k) = \sum_{i_1=0}^{1} \cdots \sum_{i_k=0}^{1} \tau_{i_1,\cdots,i_k}^{(i,j_1,\cdots,j_k)} a_1^{i_1} a_2^{i_2} \cdots a_k^{i_k}$$

for some scalars $\tau_{i_1,\cdots,i_k}^{(i,j_1,\cdots,j_k)}$. Putting those two expressions together, we get

$$f(\mathbf{x}_1, \cdots, \mathbf{x}_k)_i = \sum_{j_1=1}^{d} \cdots \sum_{j_k=1}^{d} \sum_{i_1=0}^{1} \cdots \sum_{i_k=0}^{1} \tau_{i_1,\cdots,i_k}^{(i,j_1,\cdots,j_k)} (\mathbf{x}_1)_{j_1}^{i_1} \cdots (\mathbf{x}_k)_{j_k}^{i_k}.$$

We can then group together the coefficients $\tau_{i_1,\cdots,i_k}^{(i,j_1,\cdots,j_k)}$ corresponding to the same monomials $(\mathbf{x}_1)_{j_1}^{i_1} \cdots (\mathbf{x}_k)_{j_k}^{i_k}$. E.g., the coefficients $\tau_{1,0,\cdots,0}^{(i,j_1,\cdots,j_k)}$ all correspond to the same monomial $(\mathbf{x}_1)_{j_1}$ for any values of $j_2, j_3, \cdots, j_k$. Similarly, all the coefficients $\tau_{0,1,\cdots,1}^{(i,j_1,\cdots,j_k)}$ all correspond to the same monomial $(\mathbf{x}_2)_{j_2} \cdots (\mathbf{x}_k)_{j_k}$ for any values of j_1. By grouping and summing together these coefficients into a tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{(F+1)\times\cdots\times(F+1)\times d}$, where $\boldsymbol{\mathcal{T}}_{j_1,\cdots,j_k,i}$ is equal to the sum $\sum_{j_\ell\in[F]:j_\ell=F+1\wedge i_\ell=0} \tau_{i_1,\cdots,i_k}^{(i,j_1,\cdots,j_k)}$, we obtain

$$f(\mathbf{x}_1, \cdots, \mathbf{x}_k)_i = \sum_{j_1=1}^{F+1} \sum_{j_2=1}^{F+1} \cdots \sum_{j_k=1}^{F+1} \boldsymbol{\mathcal{T}}_{j_1,j_2,\cdots,j_k,i} \left(\begin{bmatrix}\mathbf{x}_1\\1\end{bmatrix}\right)_{j_1} \left(\begin{bmatrix}\mathbf{x}_2\\1\end{bmatrix}\right)_{j_2} \cdots \left(\begin{bmatrix}\mathbf{x}_k\\1\end{bmatrix}\right)_{j_k}.$$

Since f is permutation-invariant, the tensor $\boldsymbol{\mathcal{T}}$ is partially symmetric w.r.t. its first k modes. Thus, by Lemma 1, there exist matrices $\mathbf{A} \in \mathbb{R}^{R \times F}$ and $\mathbf{B} \in \mathbb{R}^{d \times R}$ such that

$$\begin{aligned} f(\mathbf{x}_1, \cdots, \mathbf{x}_k) &= \boldsymbol{\mathcal{T}} \times_1 \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \times_2 \cdots \times_k \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \\ &= [\![\mathbf{A}, \cdots, \mathbf{A}, \mathbf{B}]\!] \times_1 \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \times_2 \cdots \times_k \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \\ &= \mathbf{B}\, \sigma \left(\mathbf{A}^\top \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \odot \cdots \odot \mathbf{A}^\top \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \right) \end{aligned}$$

with σ being the identity function, which concludes the proof.

□

B Proof of Theorem 2

Theorem. *A CP layer or rank $F \cdot k$ can compute the sum and mean aggregation functions for k vectors in $\mathbb{R}^F$.*

Consequently, for any $k \geq 1$ and any GNN $\mathcal{N}$ using mean or sum pooling with feature and embedding dimensions bounded by F, there exists a GNN with CP layers of rank $F \cdot k$ computing the same function as $\mathcal{N}$ over all graphs of uniform degree k.

Proof. We will show that the function $f : \left(\mathbb{R}^F\right)^k \to \mathbb{R}^d$ defined by

$$f(\mathbf{x}_1, \cdots, \mathbf{x}_k) = \sum_{i=1}^{k} \alpha \mathbf{x}_i$$

where $\alpha \in \mathbb{R}$, can be computed by a CP layer of rank Fk, which will show the first part of the theorem. The second part of the theorem directly follows by letting $\alpha = 1$ for the sum aggregation and $\alpha = 1/k$ for the mean aggregation.

Let $\tilde{\mathbf{e}}_1, \cdots, \tilde{\mathbf{e}}_{F+1}$ be the canonical basis of $\mathbb{R}^{F+1}$, and let $\mathbf{e}_1, \cdots, \mathbf{e}_F$ be the canonical basis of $\mathbb{R}^F$. We define the tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{(F+1) \times \cdots \times (F+1) \times d}$ by

$$\boldsymbol{\mathcal{T}} = \sum_{j=1}^{d} \sum_{\ell=1}^{k} \alpha \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{\ell - 1 \text{ times}} \circ \tilde{\mathbf{e}}_j \circ \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{k - \ell \text{ times}} \circ \mathbf{e}_j$$

where $\circ$ denotes the outer (or tensor) product between vectors. We start by showing that contracting k vectors in homogeneous coordinates along the first k modes of $\boldsymbol{\mathcal{T}}$ results in the sum of those vectors weighted by α. For any vectors $\mathbf{x}_1, \cdots, \mathbf{x}_k \in \mathbb{R}^F$, we have

$$\begin{aligned} \boldsymbol{\mathcal{T}} \times_1 \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \times_2 \cdots \times_k \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} &= \left(\sum_{j=1}^{d} \sum_{\ell=1}^{k} \alpha \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{\ell - 1 \text{ times}} \circ \tilde{\mathbf{e}}_j \circ \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{k - \ell \text{ times}} \circ \mathbf{e}_j \right) \times_1 \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \times_2 \cdots \times_k \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \\ &= \sum_{j=1}^{d} \sum_{\ell=1}^{k} \alpha \left\langle \tilde{\mathbf{e}}_{F+1}, \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \right\rangle \cdots \left\langle \tilde{\mathbf{e}}_{F+1}, \begin{bmatrix} \mathbf{x}_{\ell-1} \\ 1 \end{bmatrix} \right\rangle \left\langle \tilde{\mathbf{e}}_j, \begin{bmatrix} \mathbf{x}_\ell \\ 1 \end{bmatrix} \right\rangle \left\langle \tilde{\mathbf{e}}_{F+1}, \begin{bmatrix} \mathbf{x}_{\ell+1} \\ 1 \end{bmatrix} \right\rangle \cdots \left\langle \tilde{\mathbf{e}}_{F+1}, \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \right\rangle \mathbf{e}_j \\ &= \sum_{j=1}^{d} \sum_{\ell=1}^{k} (\alpha \cdot 1 \cdots 1 \cdot \langle \mathbf{x}_\ell, \mathbf{e}_j \rangle \cdot 1 \cdots 1)\, \mathbf{e}_j \\ &= \sum_{j=1}^{d} \left\langle \sum_{\ell=1}^{k} \alpha \mathbf{x}_\ell, \mathbf{e}_j \right\rangle \mathbf{e}_j \\ &= \sum_{\ell=1}^{k} \alpha \mathbf{x}_\ell = f(\mathbf{x}_1, \cdots, \mathbf{x}_k). \end{aligned}$$

To show that f can be computed by a CP layer of rank Fk, it thus remains to show that $\boldsymbol{\mathcal{T}}$ admits a partially symmetric CP decomposition of rank Fk. This follows from Corollary 4.3 in [45], which

states that any kth order tensor $\boldsymbol{\mathcal{A}}$ of CP rank less than k has symmetric CP rank bounded by k. Indeed, consider the tensors

$$\boldsymbol{\mathcal{A}}^{(j)} = \sum_{\ell=1}^{k} \alpha \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{\ell-1 \text{ times}} \circ \tilde{\mathbf{e}}_j \circ \underbrace{\tilde{\mathbf{e}}_{F+1} \circ \cdots \circ \tilde{\mathbf{e}}_{F+1}}_{k-\ell \text{ times}}$$

for $j \in [d]$. They are all k-th order tensor of CP rank bounded by k. Thus, by Corollary 4.3 in [45], they all admit a symmetric CP decomposition of rank at most k, from which it directly follows that the tensor $\boldsymbol{\mathcal{T}} = \sum_{j=1}^{F} \boldsymbol{\mathcal{A}}^{(j)} \circ \mathbf{e}_j$ admits a partially symmetric CP decomposition of rank at most Fk. □

C Proof of Theorem 3

Theorem. *With probability one, any function $f_{CP} : (\mathbb{R}^F)^k \to \mathbb{R}^d$ computed by a CP layer (of any rank) whose parameters are drawn randomly (from a continuous distribution) cannot be computed by a function of the form*

$$g_{sum} : \mathbf{x}_1, \cdots, \mathbf{x}_k \mapsto \sigma' \left(\mathbf{M} \left(\sigma \left(\sum_{i=1}^{k} \mathbf{W}^\top \mathbf{x}_i \right) \right) \right)$$

where $\mathbf{M} \in \mathbb{R}^{d \times R}$, $\mathbf{W} \in \mathbb{R}^{F \times R}$ and σ, σ' are component-wise activation function.

Proof. Let $f_{CP} : (\mathbb{R}^F)^k \to \mathbb{R}^d$ be the function computed by a random CP layer. I.e.,

$$f_{CP}(\mathbf{x}_1, \cdots, \mathbf{x}_k) = \mathbf{A} \left(\left(\mathbf{B}^\top \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} \right) \odot \cdots \odot \left(\mathbf{B}^\top \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} \right) \right)$$

where the entries of the matrices $\mathbf{B} \in \mathbb{R}^{(N+1) \times R}$ and $\mathbf{A} \in \mathbb{R}^{M \times R}$ are identically and independently drawn from a distribution which is continuous w.r.t. the Lebesgue measure. It is well know that since the entries of the two parameter matrices are drawn from a continuous distribution, all the entries of $\mathbf{A}$ and $\mathbf{B}$ are non-zero and distinct with probability one. It follows that all the entries of the vector $f_{CP}(\mathbf{x}_1, \cdots, \mathbf{x}_k)$ are k-th order multilinear polynomials of the entries of the input vectors $\mathbf{x}_1, \cdots, \mathbf{x}_k$, which, with probability one, have non trivial high-order interactions that cannot be computed by a linear map. In particular, with probability one, the map f_{CP} cannot be computed by any map of the form $g_{sum} : \mathbf{x}_1, \cdots, \mathbf{x}_k \mapsto \sigma' \left(\mathbf{M} \left(\sigma \left(\sum_{i=1}^{k} \mathbf{W}^\top \mathbf{x}_i \right) \right) \right)$ since the sum pooling aggregates the inputs in a way that prevents modeling independent higher order multiplicative interactions, despite the non-linear activation functions. □

D Efficiency Study

We study efficiency by comparing tGNN with models on Cora on a CPU over 10 runtimes, and compare the number of model paramterts, number of training epochs per second, and accuracy. Sampling means we sample '3' neighbors for each node or we use 'Full' neighborhood.

Table 4: Study of Efficiency vs. Performance on Cora. tGNN in comparison with GNN architectures.

	Dropout	LR	Weight Decay	Hidden	Rank	Head	Sampling	#Params	Time(s)	Epoch	Epoch/s	Acc	Std
tGNN	0	0.005	5.00E-05	32	8	_	3	58128	290.9774	1389	4.7736	85.55	1.33
tGNN	0	0.005	5.00E-05	32	32	_	3	94272	790.0373	1321	1.6721	86.25	0.58
tGNN	0	0.005	5.00E-05	32	64	_	3	142464	383.2255	1343	3.5045	86.06	1.08
tGNN	0	0.005	5.00E-05	32	128	_	3	238848	999.1514	1247	1.2481	86.76	1.19
tGNN	0	0.005	5.00E-05	32	256	_	3	431616	1193.7131	1272	1.0656	86.97	1.24
tGNN	0	0.005	5.00E-05	32	512	_	3	817152	1621.9083	1332	0.8213	87.33	1.83
tGNN	0	0.005	5.00E-05	32	1024	_	3	1588224	2377.5139	1265	0.5321	87.62	1.63
GCN	0	0.005	5.00E-05	32	_	_	3	46080	212.3576	1509	7.1059	84.29	1.02
GCN	0	0.005	5.00E-05	32	_	_	Full	46080	205.0601	1276	6.2226	85.24	1.69
GCN	0	0.005	5.00E-05	64	_	_	3	92160	316.6461	1240	3.916	85.12	2.11
GCN	0	0.005	5.00E-05	64	_	_	Full	92160	318.3962	1161	3.6464	85.59	2.03
GAT	0	0.005	5.00E-05	32	_	1	3	92238	605.3269	1998	3.3007	83.66	1.54
GAT	0	0.005	5.00E-05	32	_	1	Full	92238	548.7319	1638	2.9851	84.79	2.26
GAT	0	0.005	5.00E-05	32	_	8	3	762992	1491.5643	1594	1.069	86.26	1.35
GAT	0	0.005	5.00E-05	32	_	8	Full	762992	1524.4348	1276	0.837	87.07	1.64
GAT	0	0.005	5.00E-05	64	_	1	3	184462	626.4011	1740	2.7778	84.15	1.29
GAT	0	0.005	5.00E-05	64	_	1	Full	184462	682.7776	1465	2.1456	86.07	2.55
GAT	0	0.005	5.00E-05	64	_	8	3	1525872	2164.689	1348	0.6227	85.32	1.31
GAT	0	0.005	5.00E-05	64	_	8	Full	1525872	2143.036	1105	0.5156	87.01	0.96
GCN2	0	0.005	5.00E-05	32	_	_	3	48128	256.9575	1708	6.647	83.17	1.5
GCN2	0	0.005	5.00E-05	32	_	_	Full	48128	210.7702	1302	6.1773	84.38	2.03
GCN2	0	0.005	5.00E-05	64	_	_	3	100352	353.0055	1581	4.4787	84.7	1.13
GCN2	0	0.005	5.00E-05	64	_	_	Full	100352	307.7913	1219	3.9605	84.79	1.64
GCN2	0	0.005	5.00E-05	Input Dim	_	_	3	4117009	3013.6082	1051	0.3488	86.72	1.82
GCN2	0	0.005	5.00E-05	Input Dim	_	_	Full	4117009	3091.4392	1013	0.3277	87.54	1.66

Table 5: Study of Efficiency vs. Performance on Cora. CP pooling in comparison with classical pooling techniques.

	Dropout	LR	Weight Decay	Hidden	Rank	Sampling	#Params	Time(s)	Epoch	Epoch/s	Acc	Std
tGNN	0	0.005	5.00E-05	32	8	3	58128	290.9774	1389	4.7736	85.55	1.33
Mean	0	0.005	5.00E-05	32	_	Full	46080	449.2177	2352	5.2358	83.26	1.06
Mean	0	0.005	5.00E-05	64	_	Full	92160	398.3229	1496	3.7747	83.88	1.77
Max	0	0.005	5.00E-05	32	_	Full	46080	464.0722	2371	5.0655	83.33	1.96
Max	0	0.005	5.00E-05	64	_	Full	92160	394.1546	1496	3.7955	83.68	2.13

E Dataset

A more detailed statistics of real-world datasets.

Table 6: Statistics of node graphs.

Dataset	#Nodes	#Edges	#Node Features	#Edge Features	#Classes
Cora	2,708	5,429	1,433	/	7
Citeseer	3,327	4,732	3,703	/	6
Pubmed	19,717	44,338	500	/	3
PRODUCTS	2,449,029	61,859,140	100	/	47
ARXIV	169,343	1,166,243	128	/	40
PROTEINS	132,534	9,561,252	8	8	112

F Hyperparameter

For experiments on real-world datasets, we use NVIDIA P100 Pascal as our GPU computation resource.

Table 7: Statistics of graph datasets

Dataset	#Graphs	#Node Features	#Classes
ZINC	12,000	28	/
CIFAR10	60,000	5	10
MNIST	70,000	3	10
MolHIV	41,127	9	2

And the searching hyperparameter includes the learning rate, weight decay, dropout, decomposition rank R.

Table 8: Hyperparameter searching range corresponding to Section 5.

Hyperparameter	Searing Range
learning rate	{0.01, 0.001, 0.0001, 0.05, 0.005, 0.0005, 0.003}
weight decay	{5e-5 ,5e-4, 5e-3, 1e-5, 1e-4, 1e-3, 0}
dropout	{0, 0.1, 0.3, 0.5, 0.7, 0.8, 0.9}
R	{32, 64, 128, 256, 512, 25, 50, 75, 100, 200}

Table 9: Hyperparameters for tGNN corresponding to Section 5.

Dataset	learning rate	weight decay	dropout	R
Cora	0.001	5.00E-05	0.9	512
Citeseer	0.001	1.00E-04	0	512
Pubmed	0.005	5.00E-04	0.1	512
PRODUCTS	0.001	5.00E-05	0.3	128
ARXIV	0.003	5.00E-05	0	512
PROTEINS	0.0005	5.00E-04	0.9	50
ZINC	0.005	5.00E-04	0	100
CIFAR10	0.005	1.00E-04	0	100
MNIST	0.005	5.00E-05	0	75
MolHIV	0.001	5.00E-05	0.8	100