SPECTRAL ATTENTION NETWORKS

Meet the authors!

Devin Kreuzer

Dominique Beaini

Mila

Valence

The Drug Design Company

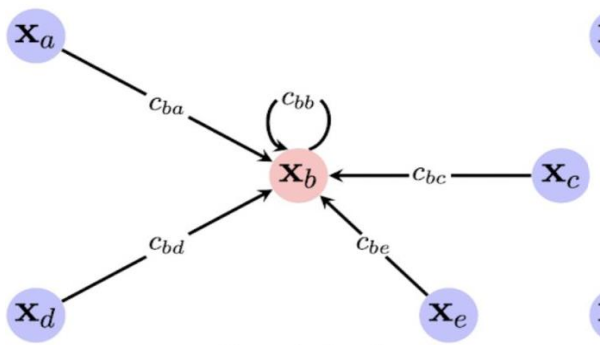
Prof. William L. Hamilton

Vincent Letourneau

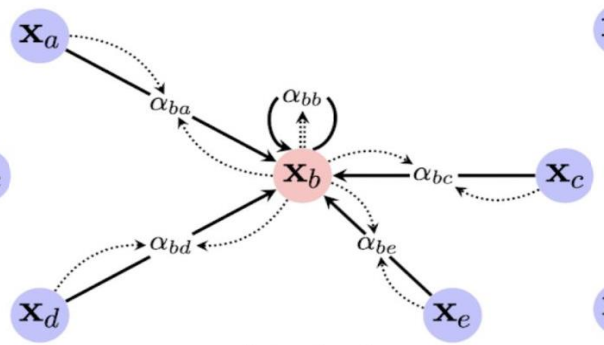
Prudencio Tossou

Preliminaries

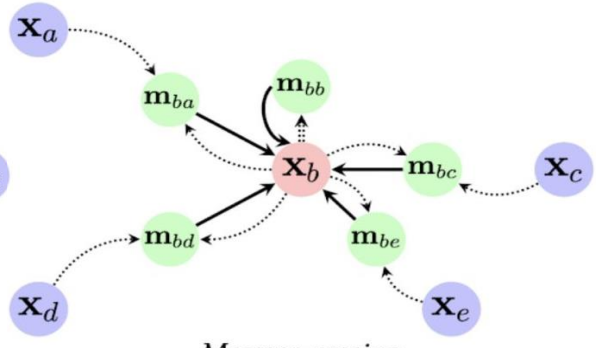
Current GNNs strategies:



Convolutional



Attentional



Message-passing

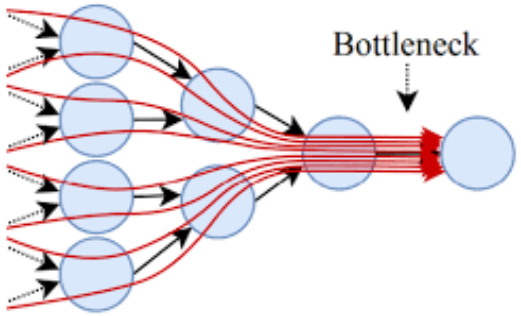
$$\mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} c_{ij}\psi(\mathbf{x}_j)\right)$$

$$\mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} a(\mathbf{x}_i, \mathbf{x}_j)\psi(\mathbf{x}_j)\right)$$

$$\mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$$

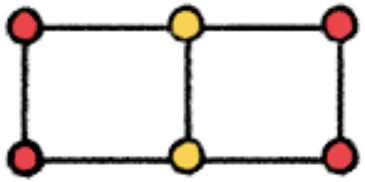
Image from Petar Velickovic, DeepMind

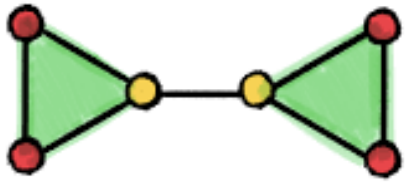
Limitations:



(b) The bottleneck of graph neural networks

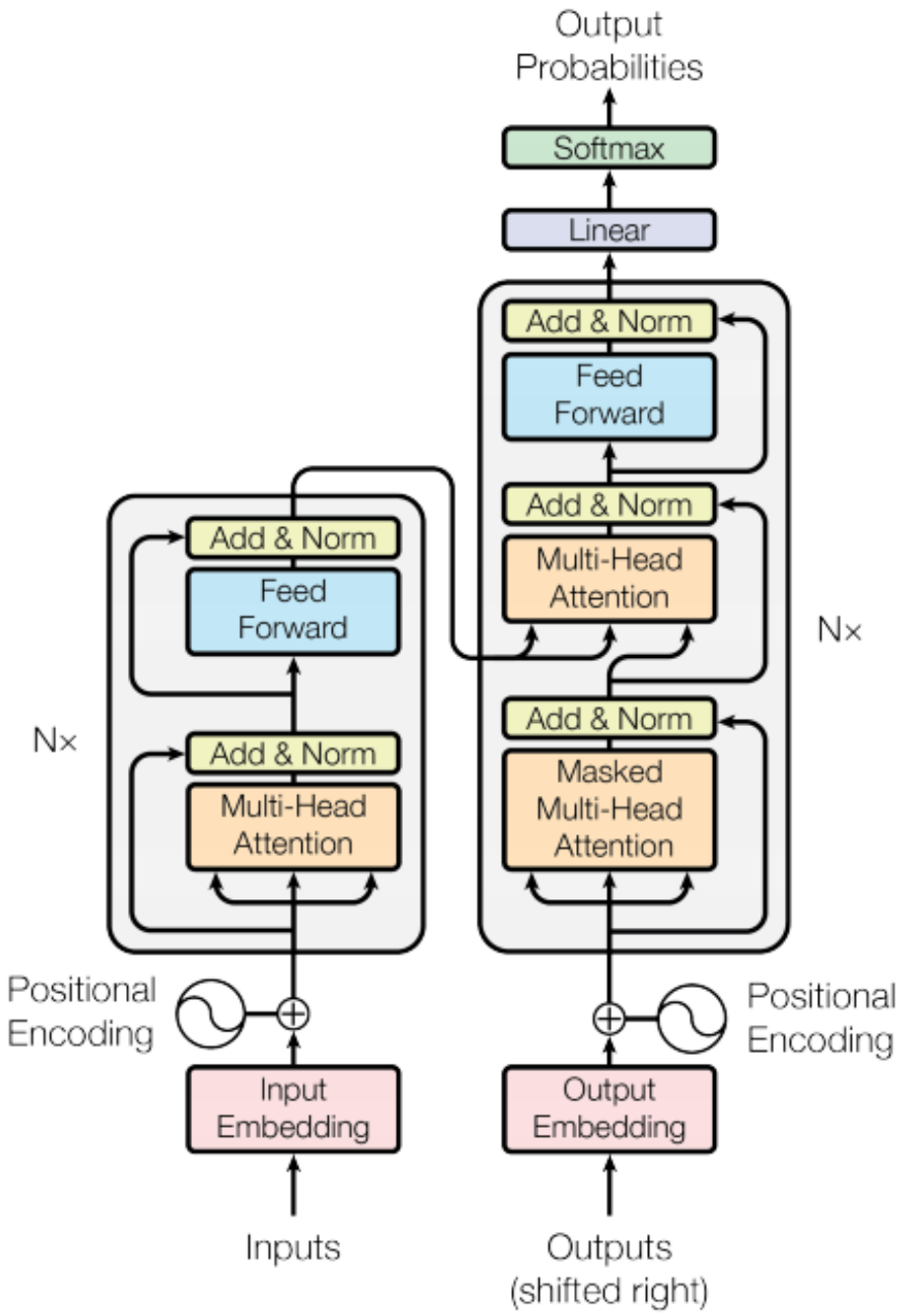
Image from Uri Alon and Eran Yahav, Technion





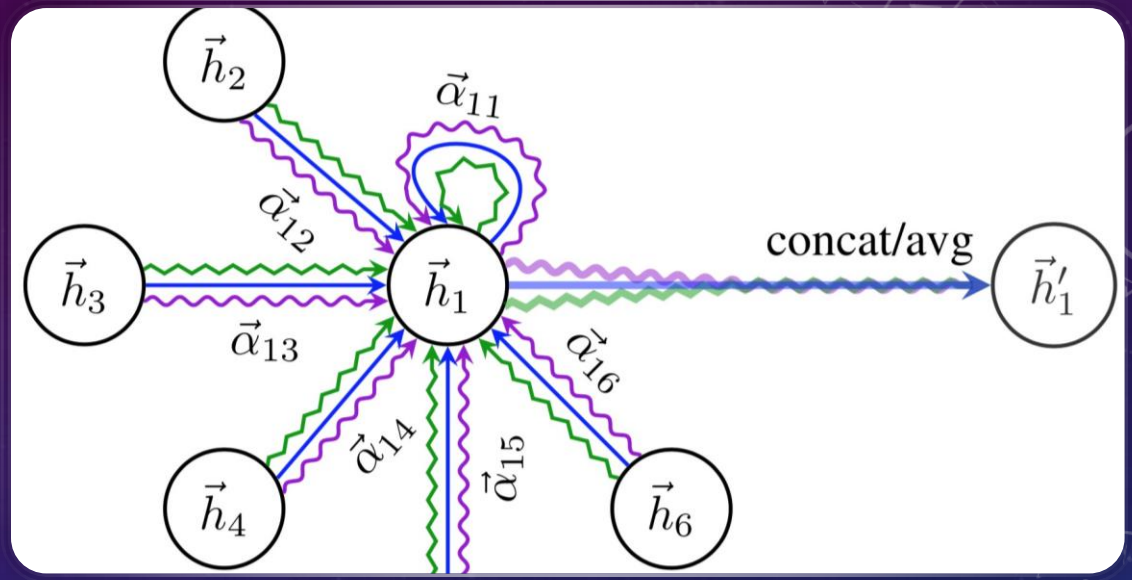
Transformer

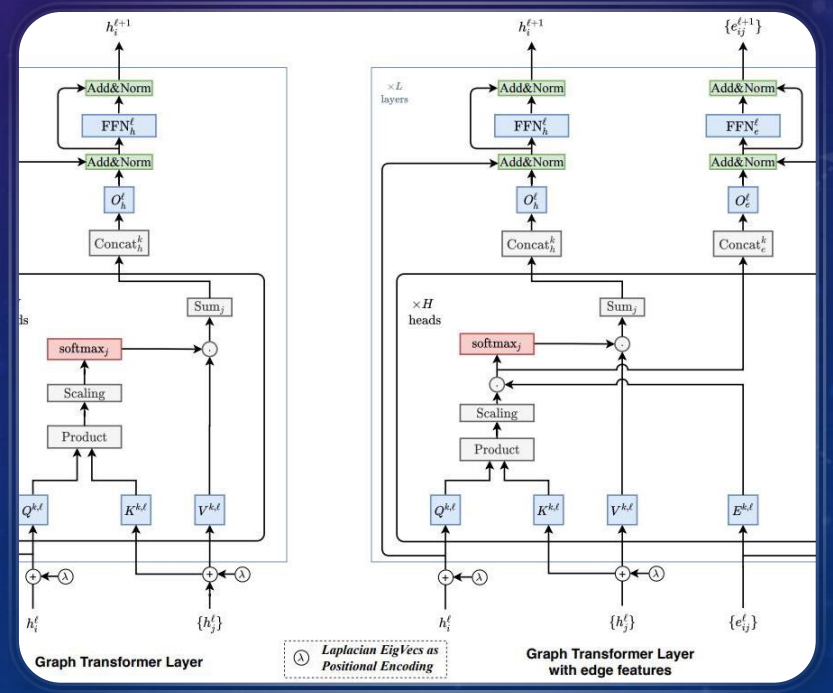
- Revolutionized NLP by removing many inductive biases
- **Main challenge: Positional Encodings for arbitrary graphs**
- Answer: graph Laplacian



PREVIOUS WORK

- We expand primarily on:
  1. GAT [1]
  2. GT [2]

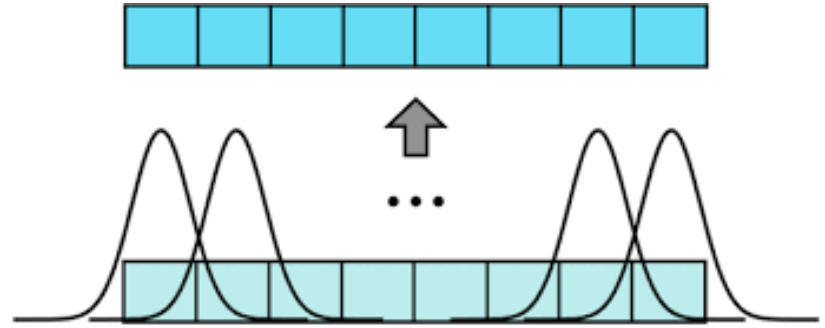




[1] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks.arXiv preprint arXiv:1710.10903, 2017

[2] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs, 2020

Positional Encodings



- Original Transformer PEs
  - Frequencies are pre-determined

- In arbitrary graphs, this process isn't feasible
  - No natural 'pos' value
  - Equivalent given by eigenvectors ϕ of the Laplacian L
  - $L = D - A, \quad L_{sym} = D^{-1/2} L D^{-1/2}$

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Some basic Mathematical intuition

- ϕ : function over domain of nodes
- N_i: neighborhood of node i
- L: operator on functions over nodes

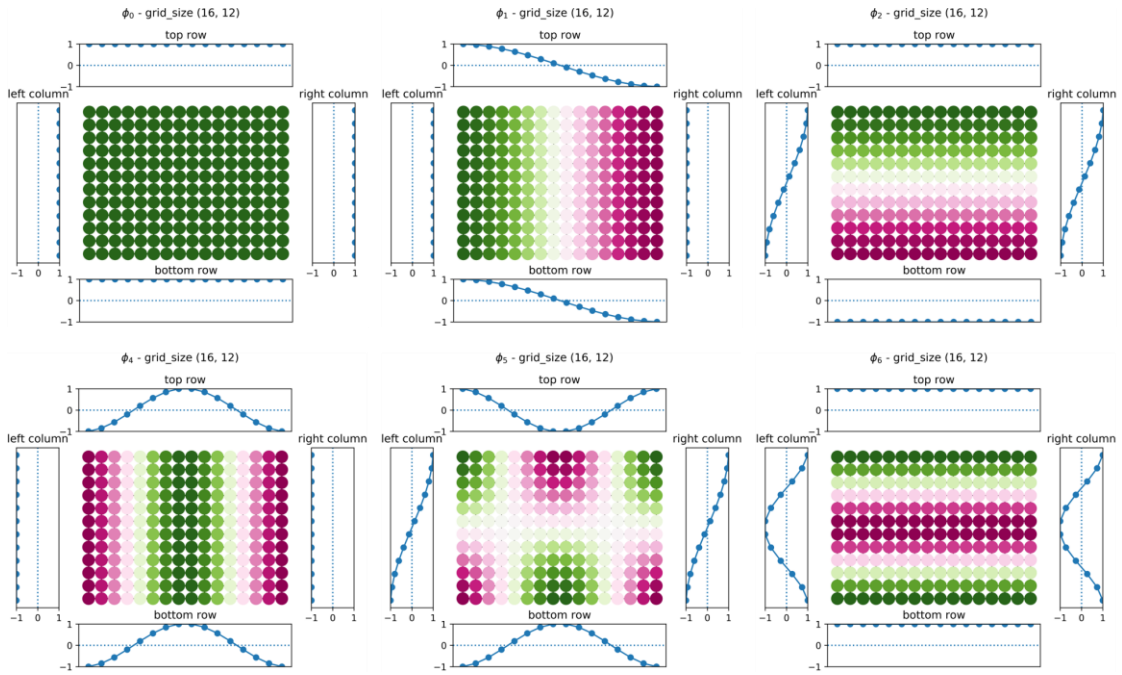
$$(L\phi)_i \;=\; |\,N_i\,|\,\phi_i \;-\sum_{j\epsilon N_i}\phi_j$$

$$\phi^T L\phi \;=\; \left[|\,N_1\,|\,\phi_1^2 - \sum_{j\epsilon N_1}\phi_j^2\right] - \left[|\,N_2\,|\,\phi_2^2 - \sum_{j\epsilon N_2}\phi_j^2\right] - \ldots = \lambda$$

- The eigenvectors ϕ assign values to each node which minimize the quantity λ
  - ϕ are sine/cosine functions traversing the graph with frequency λ
  - Can as positional encodings!

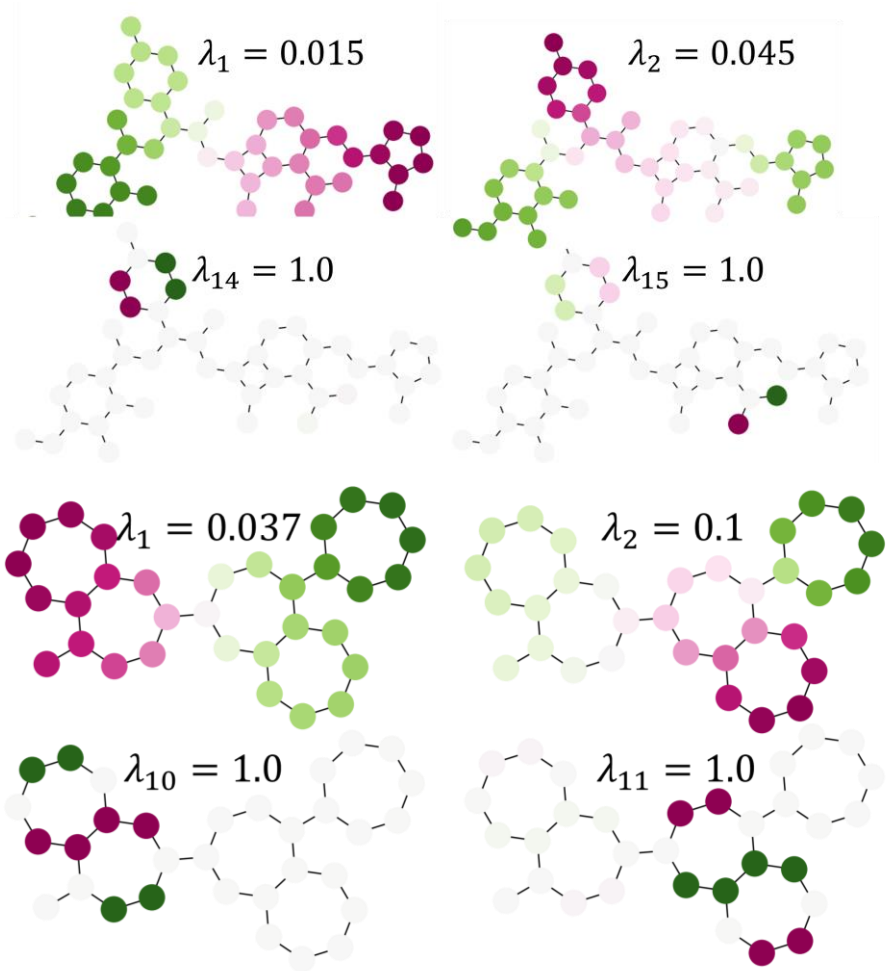
Laplace eigenvectors are sine-wave equivalents

Eigenvectors of a grid



λ: Eigenvalue, represents the frequency of the wave
$\boldsymbol{\phi}$: Eigenvector, represents the sine/cosine function

Eigenvectors of a molecule

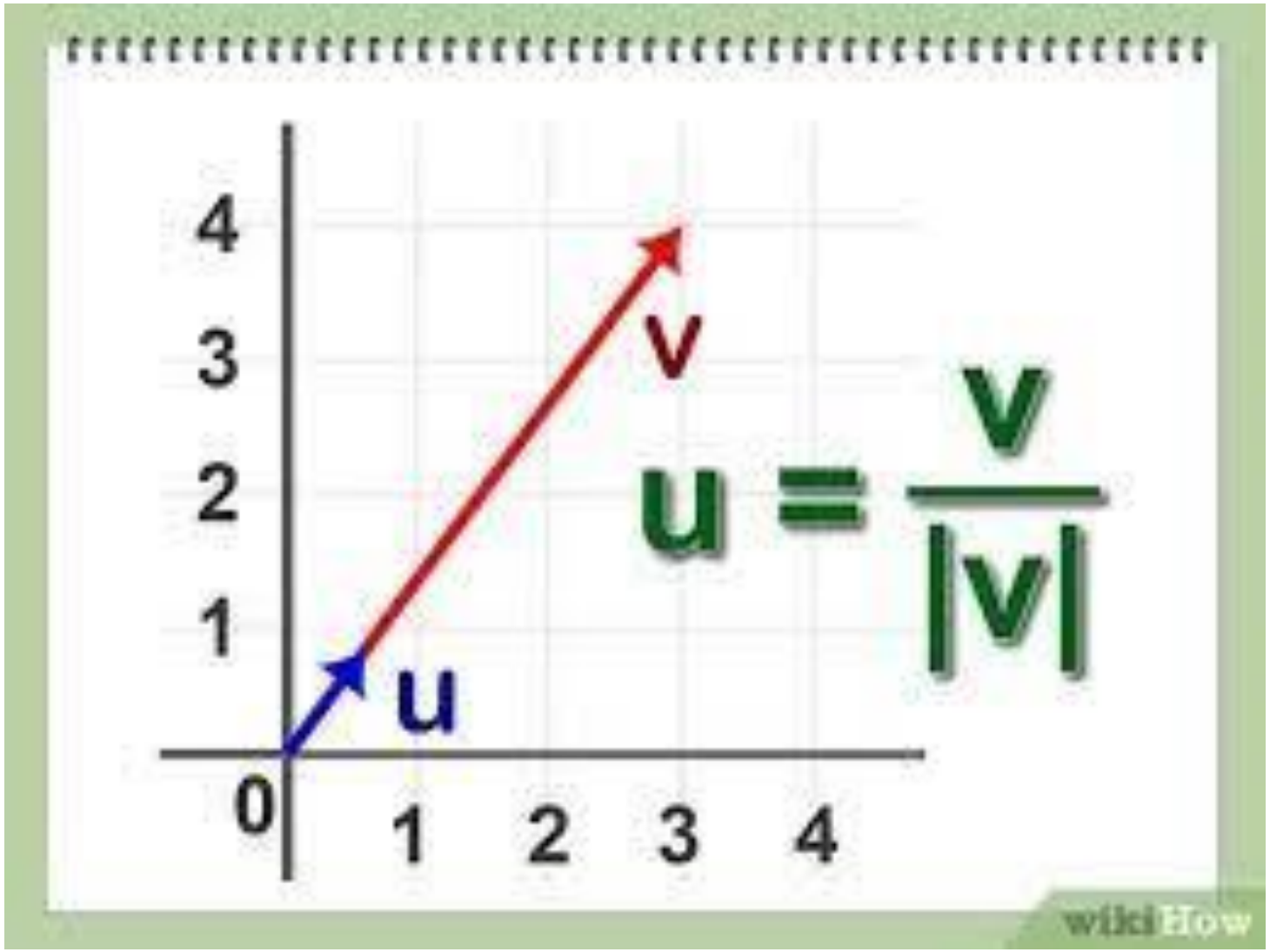


LAPLACE EIGENFUNCTIONS *ETIQUETTE*

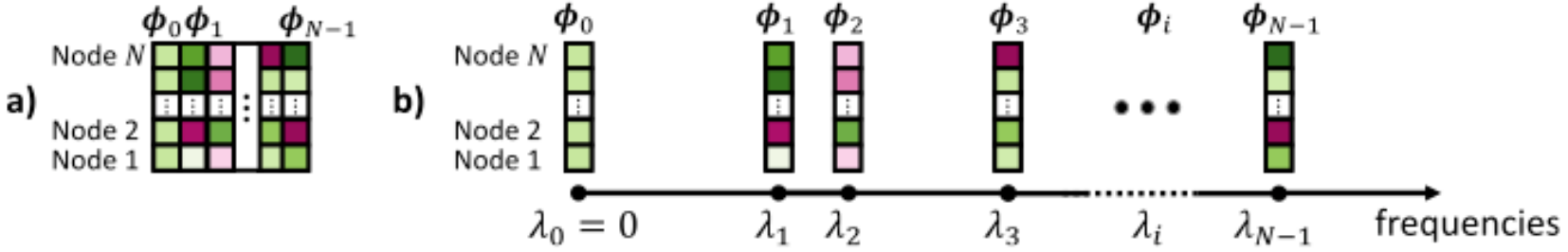
KEY CHALLENGES IN DIRECTLY CONCATENATING/ADDING ϕ AS PE

Etiquette #1: Normalization

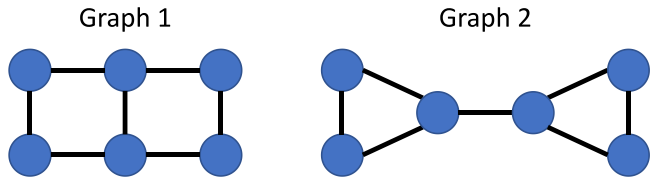
- Given ϕ, $c\phi$ is also a valid option
- Must pick one vector in the span of ϕ
- Select normalized eigenvectors: $\langle \phi, \phi \rangle = 1$



Etiquette #2: Eigenvalues



- An eigenvalue supplies valuable information about its eigenvector
- Eigenvalue spectrum provides important information about graph structure
- Important physical properties depend on the eigenvalues:

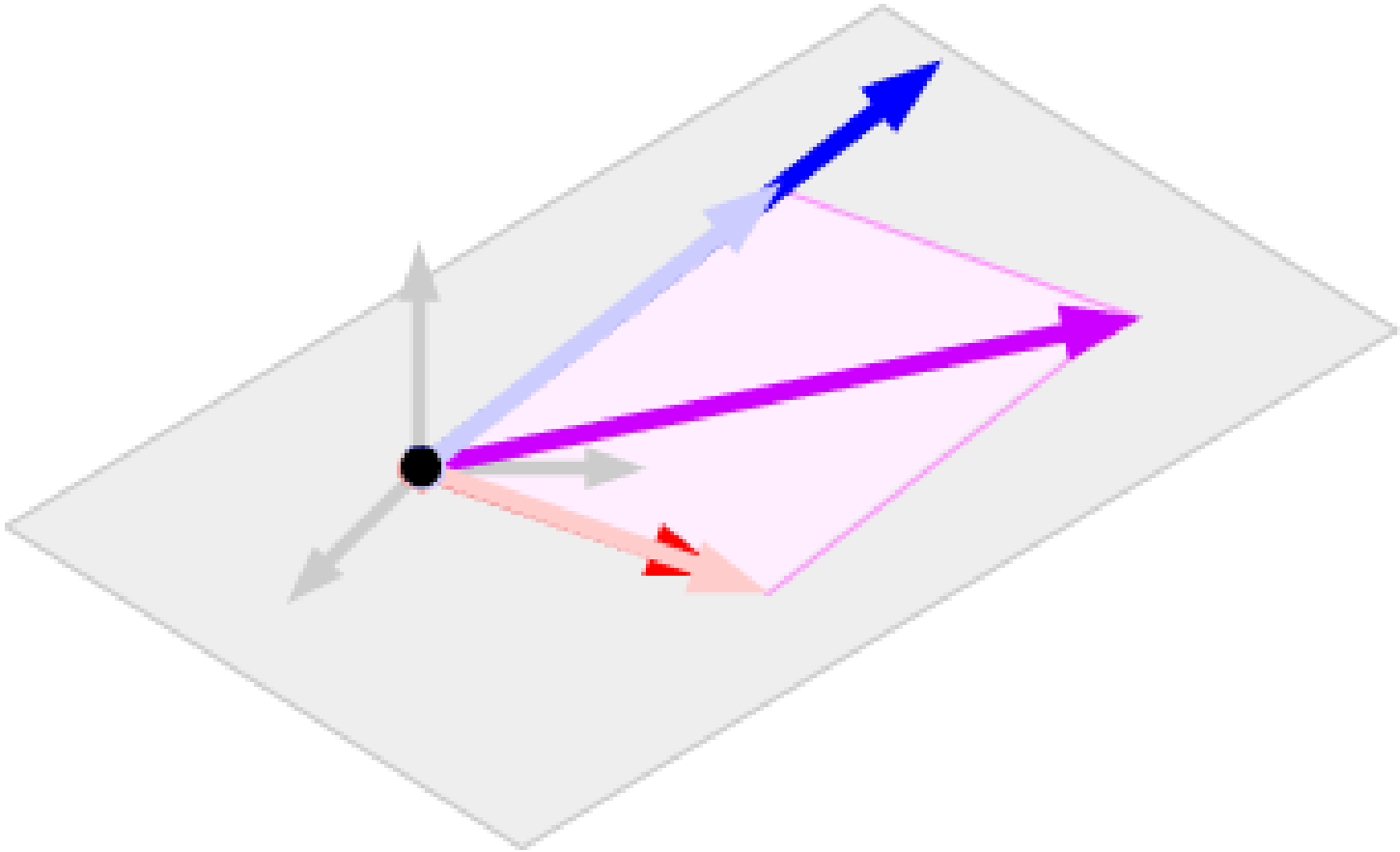


Eigenvalues of Graph 1	Eigenvalues of Graph 2
0	0
1	0.438
2	3
3	3
3	3
5	4.562

$$d_D^2(j_1, j_2) = \sum_{k>0} e^{-2t\lambda_i} (\phi_{i,j_1} - \phi_{i,j_2})^2 \quad , \quad d_B^2(j_1, j_2) = \sum_{i>0} \frac{(\phi_{i,j_1} - \phi_{i,j_2})^2}{\lambda_i^2}$$

Etiquette #3: Multiplicities

- Two eigenvectors can have the same eigenvalue
- Need to pick a vector in the space spanned by both!
  - Even more challenging!



Etiquette #4: Variable number of eigenvectors

- A graph G_i can have at most N_i linearly independent eigenvectors
- Prior elected to select a fixed number k eigenvectors for each graph, where $\mathrm{k} \leq N_i, \forall\ i$
- Major bottleneck when the smallest graphs << largest graphs in the dataset s
- Need for model that uses fixed dimension PE that captures entire spectrum

Etiquette #5: Sign ambiguity

- Given ϕ, $-\phi$ is also a valid
- Left with 2^k possible combination of valid eigenvectors
- Previous work proposed data augmentation: random sign flipping

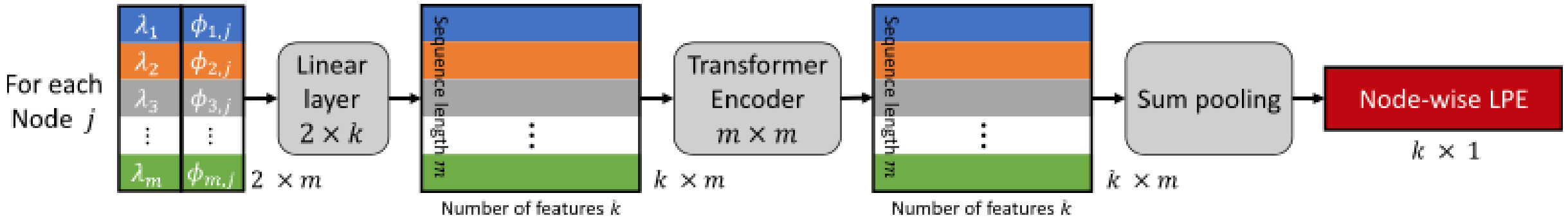
MODEL ARCHITECTURE

OVERVIEW

1. Pre-compute first *m* eigenvectors and fully-connect each graph (noting the added edges)
2. Learn positional encodings that address the *etiquettes* previously described
3. Concatenate the learned positional encodings (LPE) to the input node features
4. Apply the Main Transformer

LPE over Nodes

- Treat Laplace spectrum as a variable-length sequence where λ is the position of ϕ
- A Transformer is naturally suited to learn PEs

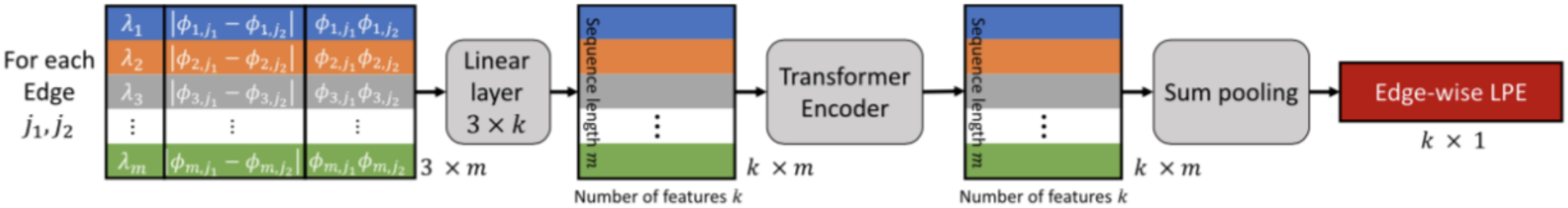


LPE over Nodes

- This technique addresses *etiquettes #1-4*
  - L2 normalized eigenvectors
  - Eigenvalues paired with eigenvectors
  - Number of eigenvectors treated as a variable
  - Aware of multiplicities
- *Etiquette #5* is unaddressed
  - Randomly flip sign of eigenvectors during training
  - Resolved in the LPE Over Edges

LPE Transformer over edges

- Very important notion:
  - ϕ on its own reveals no information at all
  - $\Delta\phi$ tells us everything!
  - Just like potential energy
- Laplace PEs are more naturally represented as edge features



- Downside: Huge computational burden
  - Extra factor of N if considering fully connected graphs

Main Graph Transformer

- Based on previous work
  - Multi-head attention:

$$\hat{\boldsymbol{h}}_i^{l+1} = \boldsymbol{O}_h^l \Big\|_{k=1}^{H} \Big(\sum_{j \in V} w_{ij}^{k,l} \boldsymbol{V}^{k,l} \boldsymbol{h}_j^l\Big)$$

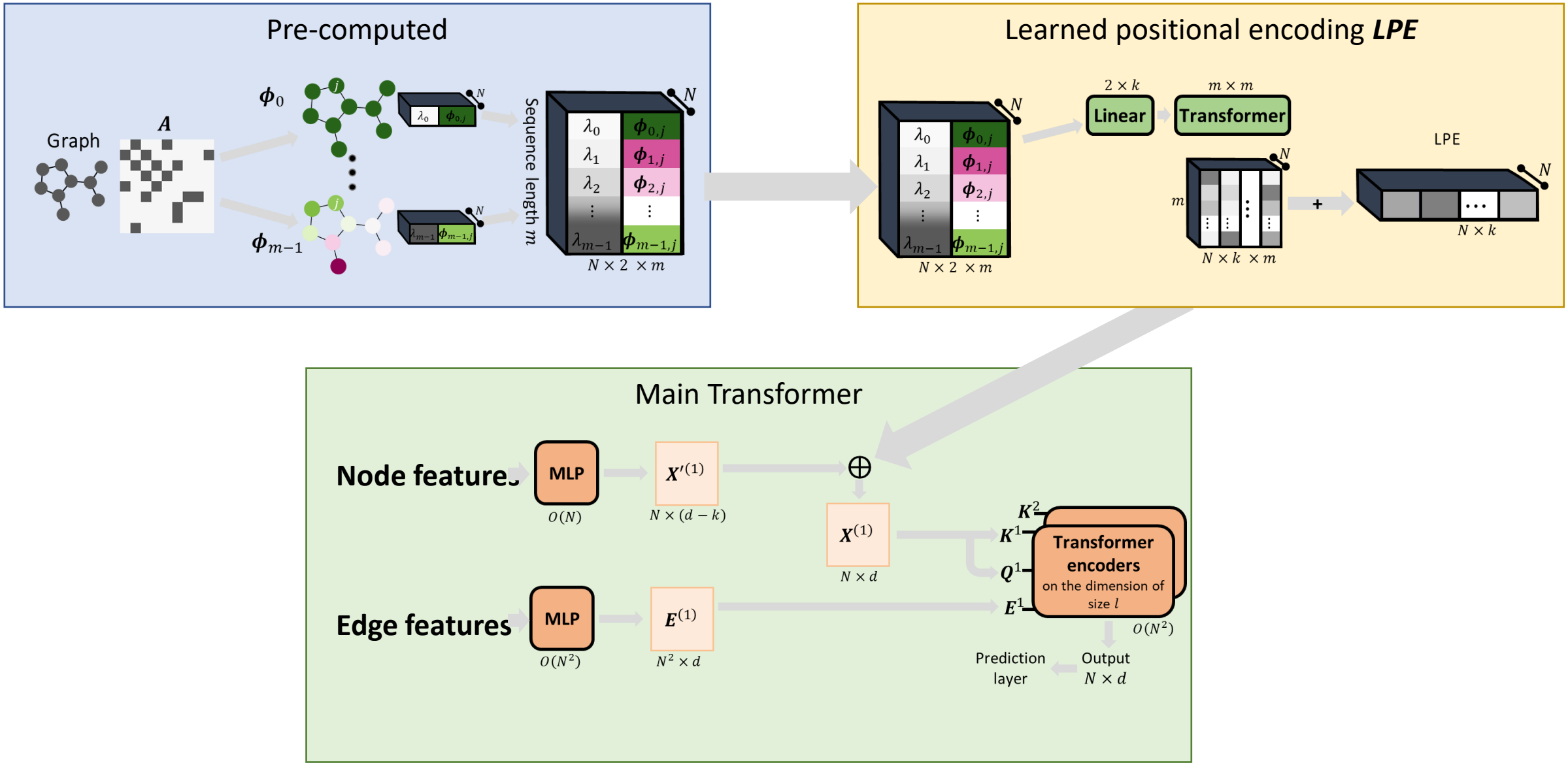
$$\hat{\hat{\boldsymbol{h}}}^{l+1} = \mathrm{Norm}(\boldsymbol{h}_i^l + \hat{\boldsymbol{h}}_i^{l+1}), \quad \hat{\hat{\boldsymbol{h}}}_i^{l+1} = \boldsymbol{W}_2^l \mathrm{ReLU}(\boldsymbol{W}_1^l \hat{\hat{\boldsymbol{h}}}_i^{l+1}), \quad \boldsymbol{h}_i^{l+1} = \mathrm{Norm}(\hat{\hat{\boldsymbol{h}}}^{l+1} + \hat{\hat{\boldsymbol{h}}}_i^{l+1})$$

- Two key additions
  1. **Separate** weights for real and added edges
  2. Hyperparameter that tunes the weight between local and global attention

$$\hat{\boldsymbol{w}}_{ij}^{k,l} = \begin{cases} \dfrac{\boldsymbol{Q}^{1,k,l}\boldsymbol{h}_i^l \circ \boldsymbol{K}^{1,k,l}\boldsymbol{h}_j^l \circ \boldsymbol{E}^{1,k,l}\boldsymbol{e}_{ij}}{\sqrt{d_k}} & \text{if } i \text{ and } j \text{ are connected in sparse graph} \\ \dfrac{\boldsymbol{Q}^{2,k,l}\boldsymbol{h}_i^l \circ \boldsymbol{K}^{2,k,l}\boldsymbol{h}_j^l \circ \boldsymbol{E}^{2,k,l}\boldsymbol{e}_{ij}}{\sqrt{d_k}} & \text{otherwise} \end{cases}$$

$$w_{ij}^{k,l} = \begin{cases} \frac{1}{1+\gamma} \cdot \mathrm{softmax}(\sum_{d_k} \hat{\boldsymbol{w}}_{ij}^{k,l}) & \text{if } i \text{ and } j \text{ are connected in sparse graph} \\ \frac{\gamma}{1+\gamma} \cdot \mathrm{softmax}(\sum_{d_k} \hat{\boldsymbol{w}}_{ij}^{k,l}) & \text{otherwise} \end{cases}$$

Overview


Pre-computed
Graph
A
ϕ_0
ϕ_{m-1}
Sequence length m
$N \times 2 \times m$
Learned positional encoding ***LPE***
$2 \times k$
Linear
$m \times m$
Transformer
LPE
$N \times k \times m$
$N \times k$
Main Transformer
Node features
MLP
$O(N)$
$X'^{(1)}$
$N \times (d-k)$
$X^{(1)}$
$N \times d$
Edge features
MLP
$O(N^2)$
$E^{(1)}$
$N^2 \times d$
Transformer encoders
on the dimension of size l
$O(N^2)$
Prediction layer
Output
$N \times d$

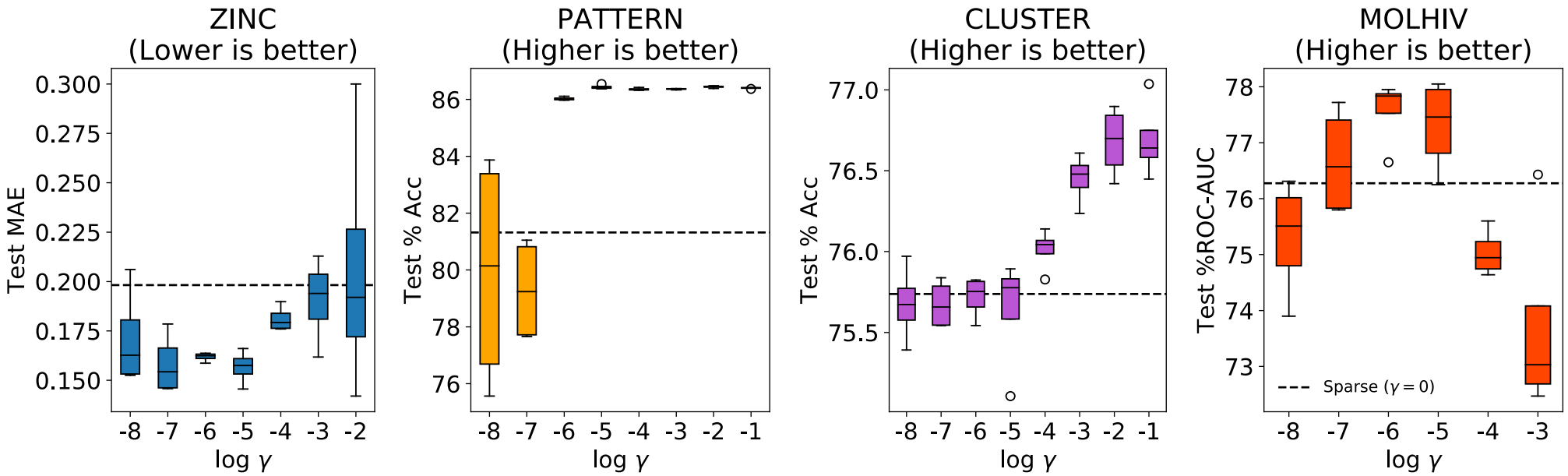
LIMITATIONS

- Node LPE does not address sign invariance issue
- Computational complexity
  - Node LPE: $O(N^3)$ if considering all eigenfunctions
  - Edge LPE: $O(N^4)$ if considering all eigenfunctions and fully–connected graphs
  - Main Graph Transformer: $O(N^2)$

EXPERIMENTS

Ablation studies

- Study the effect of hyperparameter γ



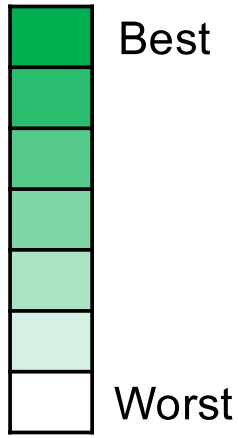
Ablation studies

- Compare sparse to full attention

Model details		ZINC	PATTERN	CLUSTER	MOLHIV
Attention	**LPE**	**MAE**	**% ACC**	**% ACC**	**% ROC-AUC**
Sparse	-	0.267 ± 0.032	83.613 ± 0.663	75.683 ± 0.098	73.46 ± 0.71
Sparse	Node	0.198 ± 0.004	81.329 ± 2.150	75.738 ± 0.106	76.61 ± 0.62
Full	-	0.392 ± 0.055	86.322 ± 0.049	76.447 ± 0.177	73.84 ± 1.80
Full	Node	$\mathbf{0.157 \pm 0.006}$	$\mathbf{86.441 \pm 0.040}$	$\mathbf{76.691 \pm 0.247}$	$\mathbf{77.57 \pm 0.61}$

Comparison to SOTA

	ZINC	PATTERN	CLUSTER	MOLHIV
Model	**MAE**	**% ACC**	**% ACC**	**% ROC-AUC**
GCN	0.367 ± 0.011	71.892 ± 0.334	68.498 ± 0.976	76.06 ± 0.97
GraphSage	0.398 ± 0.002	50.492 ± 0.001	63.844 ± 0.110	-
GatedGCN	0.282 ± 0.015	85.568 ± 0.088	73.840 ± 0.326	-
GatedGCN-PE	0.214 ± 0.013	86.508 ± 0.085	76.082 ± 0.196	
GIN	0.526 ± 0.051	85.387 ± 0.136	64.716 ± 1.553	75.58 ± 1.40
PNA	0.142 ± 0.010	-	-	79.05 ± 1.32
DGN	-	-	-	$\mathbf{79.70 \pm 0.97}$
Attention-based				
GAT	0.384 ± 0.007	78.271 ± 0.186	70.587 ± 0.447	-
GT (sparse)	0.226 ± 0.014	84.808 ± 0.068	73.169 ± 0.662	-
GT (full)	0.598 ± 0.049	56.482 ± 3.549	27.121 ± 8.471	-
SAN (ours)	$\mathbf{0.139 \pm 0.006}$	$\mathbf{86.581 \pm 0.037}$	$\mathbf{76.691 \pm 0.247}$	77.85 ± 0.65



CONCLUSION

- First successful fully-connected GNN model to perform well
- Outperforms other Attention-based models by wide margin
- Large computational bottleneck
  - Need to implement variations that scale linearly or logarithmically to enable Edge LPE

THANK YOU!!!

Follow us on Twitter!
@dom_beaini
@KreuzerDevin
@valence_ai