ICLR

Geometric Transformers for Protein Interface Contact Prediction

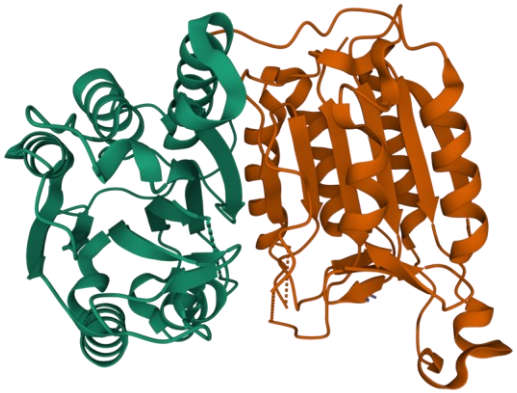
Alex Morehead, Chen Chen, & Jianlin Cheng

College of Engineering
University of Missouri

- 1: Overview
- 2: Preliminaries
- 3: DeepInteract & The Geometric Transformer
- 4: Results & Conclusions

Protein Complexes

Interacting Protein Chains

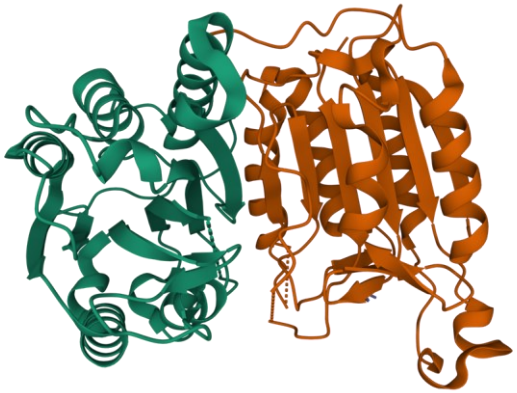


PDB ID: 3H11

- Proteins drive fundamental processes in all known forms of life
- They often interact with one another to form new macromolecules (i.e., protein *complexes*) with new functions

Contact Prediction at Protein Interfaces

Interacting Protein Chains

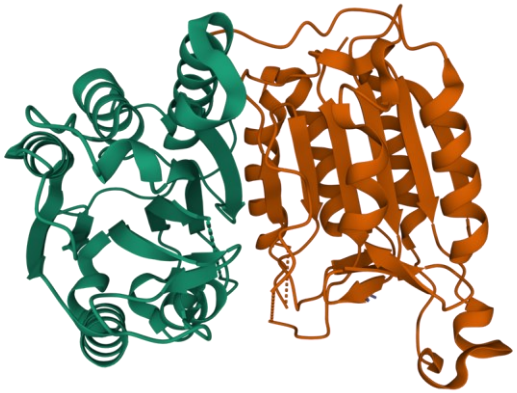


PDB ID: 3H11

- Here, we want to predict which of a protein's atoms will be the *contact* points between two protein *chains*
- The number of *inter-chain* (i.e., *interface*) contact points is often very low

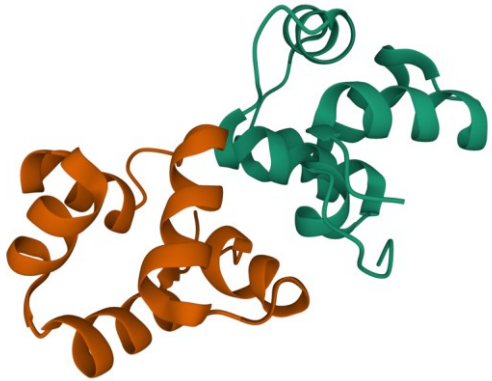
Contact Prediction at Protein Interfaces

Interacting Protein Chains



PDB ID: 3H11

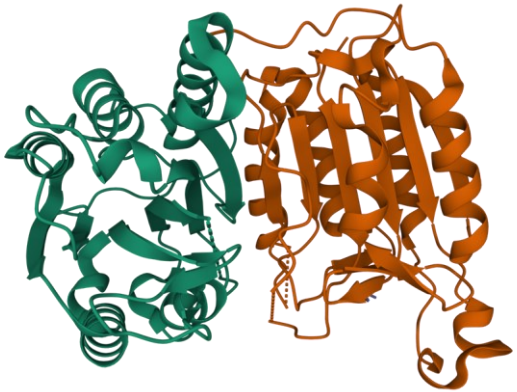
Inter-Chain Contact Points



PDB ID: 6TRI

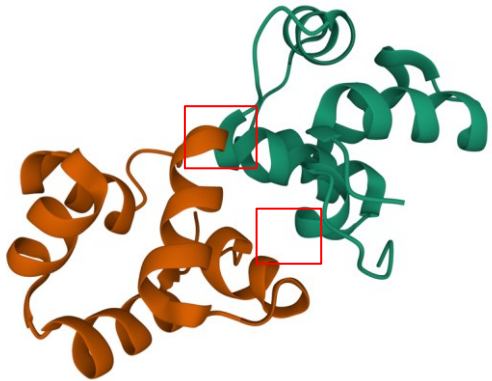
Contact Prediction at Protein Interfaces

Interacting Protein Chains



PDB ID: 3H11

Inter-Chain Contact Points



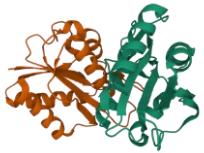
PDB ID: 6TRI

Question

- Knowing a protein chain's 3D structure can provide detailed molecular information about the geometry of its constituent parts and about the intra-chain interactions between its atoms

PDB ID: 4HEQ; Chain A

- Q. Can we utilize graph-based self-attention and the rich geometric information available in 3D protein structures to more precisely predict which atoms serve as contact points between chains?



PDB ID: 4HEQ; Chains A & B

What We Know So Far 💭

- There are many works that have approached interface contact prediction by representing protein chains as flat (i.e., 2D) feature tensors or graphs to serve as inputs for neural networks such as CNNs and geometry-agnostic GNNs

- Nonetheless, no works on interface contact prediction have leveraged two novel ideas:
  - (1) Geometric deep learning to *evolve* protein chains' geometric representations
  - (2) Node-local graph-based self-attention similar to that of [VSPUJGKP 17] and [DB 21]

What We Show

- We introduce the new Geometric Transformer, a graph-based Transformer model trained to evolve representations of 3D protein chains in an SE(3)-invariant manner (e.g., to simplify its learning landscape) 😀

- This model yields new state-of-the-art results for protein interface contact prediction

- The Geometric Transformer also outlines an alternative means of message-passing and information processing on geometric graphs
  - Examples:
    - (1) Treating edges as pseudo-nodes and message passing with them like nodes
    - (2) Enabling the network to learn to gate geometric features much like previous work with RNNs
      - Adapting insights from [LWLLZOJ 22] to model large biomolecules such as proteins

- 1: Overview
- 2: Preliminaries
- 3: DeepInteract & The Geometric Transformer
- 4: Results & Conclusions

Structural Foundations

- As many of you may already be aware, Computational Biology (CB) is focused on the study of biological entities and concepts through the lense of computation

- Geometric Deep Learning (GDL), on the other hand, is a branch of machine learning dedicated to the study, analysis, and representation learning of objects with some underlying structure

Inductive Priors (1)

- One of my goals for this talk is for us all to begin thinking more intentionally about the inductive priors (i.e., the architectural assumptions) we can bring to the sets of problems we are interested in solving

- One such example can be found in the use of convolutional neural networks (CNNs) for image-based deep learning

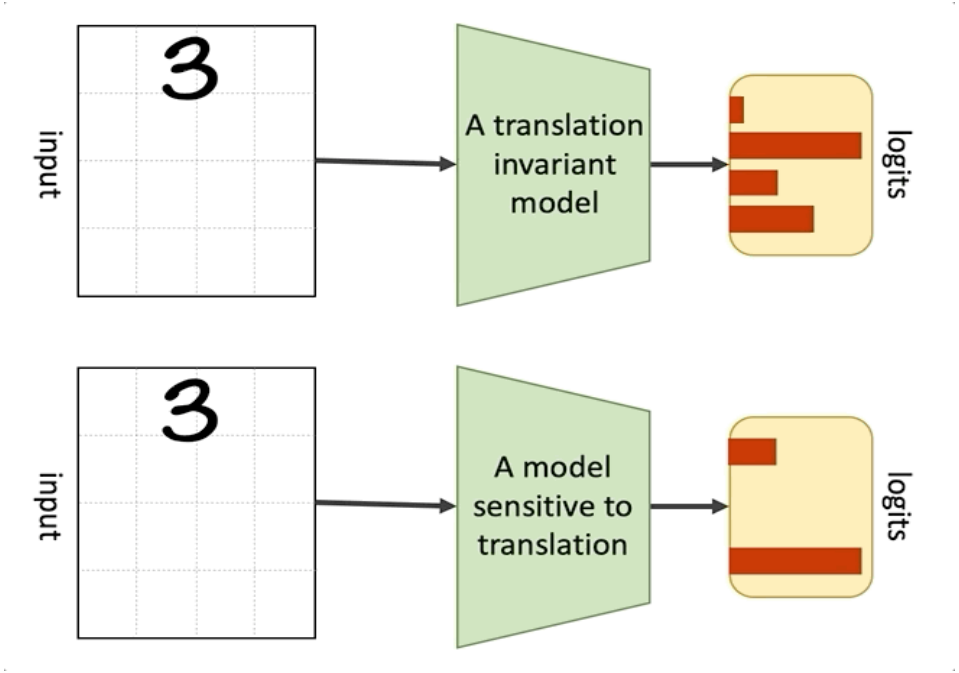
Inductive Priors (2)

- In Geometric Deep Learning, Bronstein et al. (2021) argue that CNNs have become popular for such image-based tasks because of the translation equivariance shared-weight neural networks such as CNNs possess

- The symmetry-preserving properties these neural networks have indicate that their learned feature representations transform in response to equivalent transformations on the domain of their input data

Geometric Deep Learning

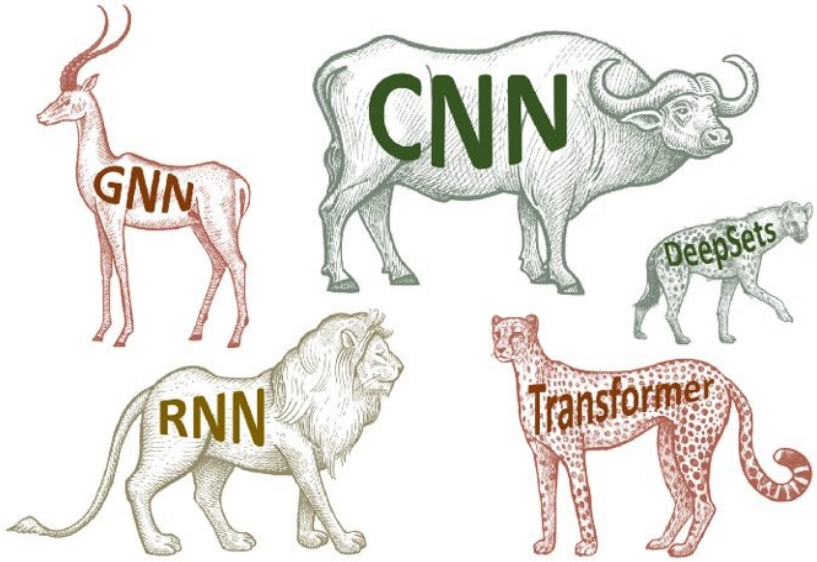
- Translation Equivariance in CNNs



- Credit: Samira Abnar

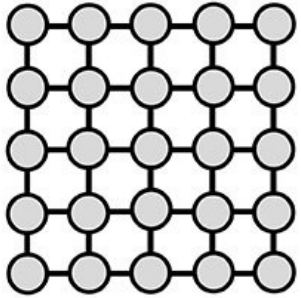
Geometric Deep Learning

- Graph neural networks (GNNs) can be seen as one of many existing geometric deep learning algorithms
- Others, for example, include Recurrent Neural Networks (RNNs) and DeepSets

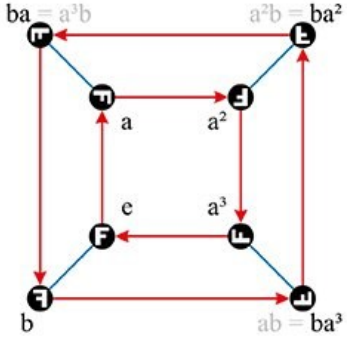


- Credit: Michael Bronstein

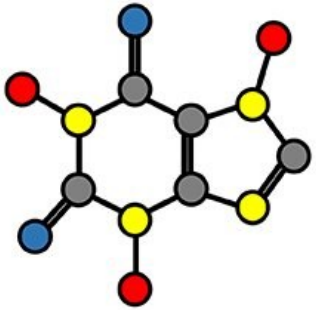
The 4G of Geometric Deep Learning



Grids



Groups

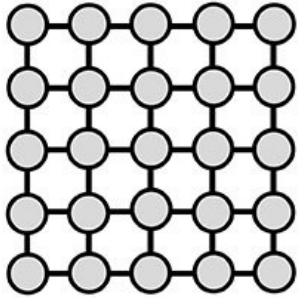


Graphs

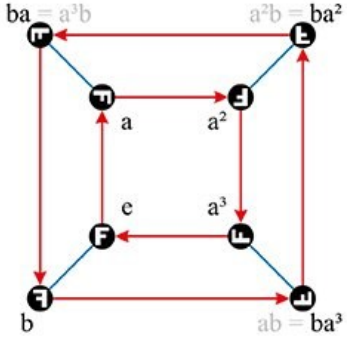
“Gauges”

“Erlangen Programme” of Deep Learning

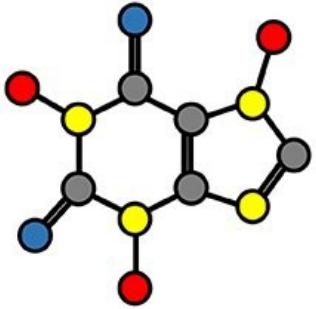
The 4G of Geometric Deep Learning



Grids



Groups



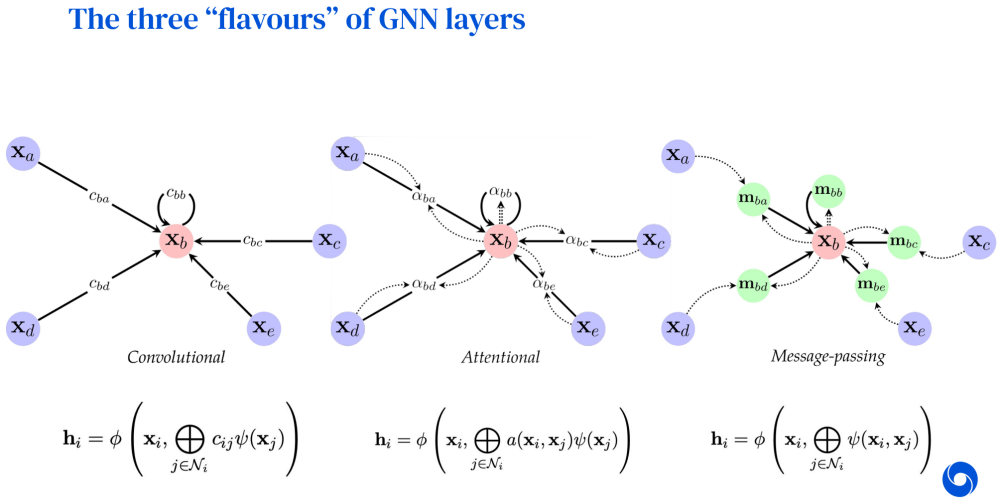
Graphs

“Gauges”

“Erlangen Programme” of Deep Learning

Graph Neural Networks and their Variations

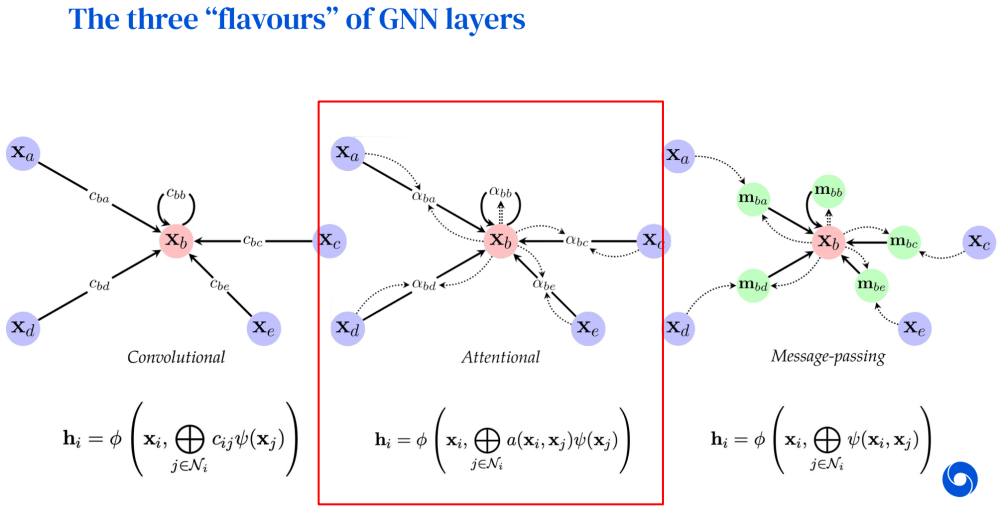
- It can often be helpful to group graph neural networks into three related categories



- Credit: Petar Veličković

Graph Neural Networks and their Variations

- It can often be helpful to group graph neural networks into three related categories

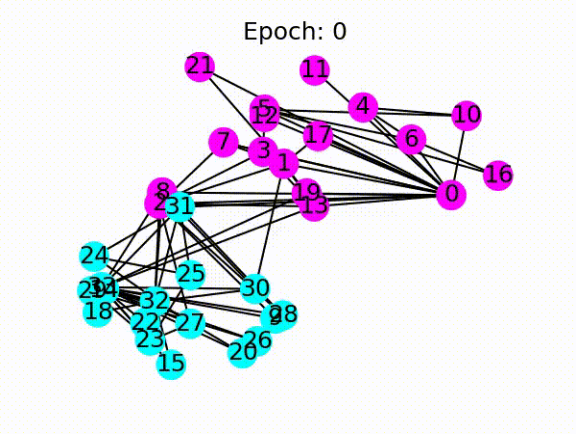


- Credit: Petar Veličković

An Example of Graph Geometric Deep Learning

- Labeled nodes: #0 and #33
- Calculate losses only from the label nodes
- Doesn’t require a lot of labeled samples

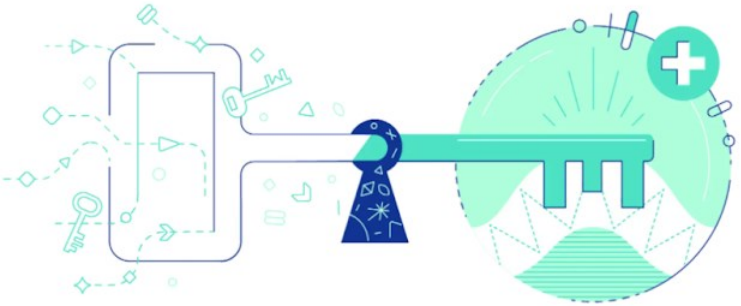
- Feature-less graph
- Use embedding weights to represent nodes
- (i.e., converting node ID to vector of weights)

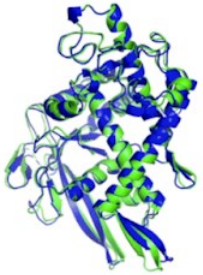


References: [KW 17] & [MCC 22]

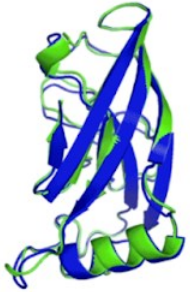
Geometric Deep Learning in Computational Biology

DeepMind



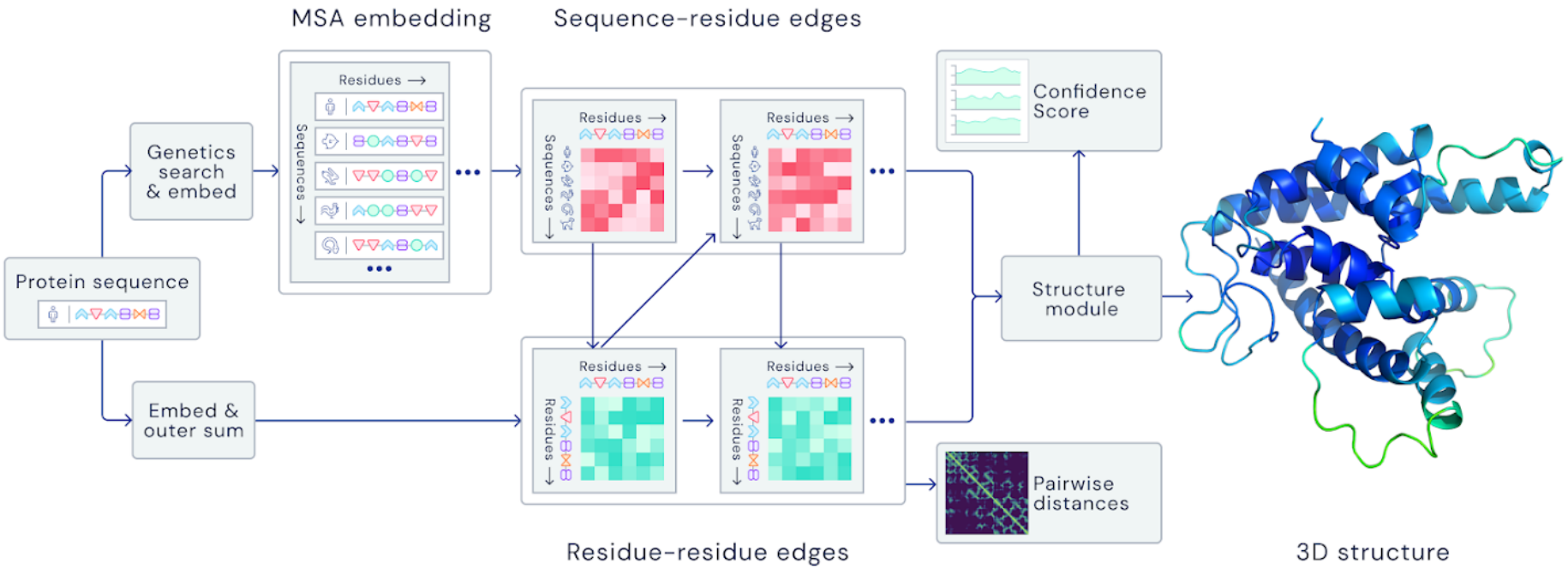


T1037 / 6vr4
90.7 GDT
(RNA polymerase domain)

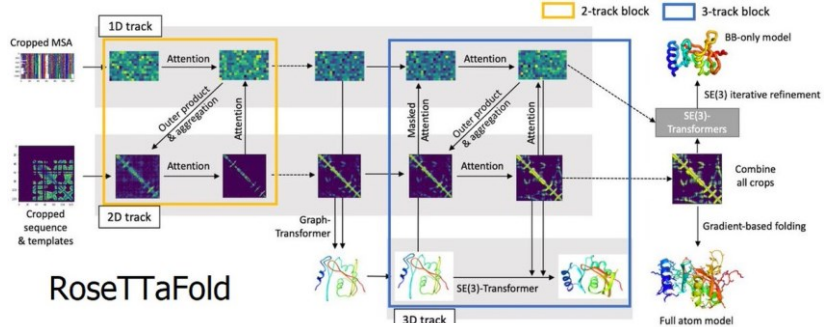


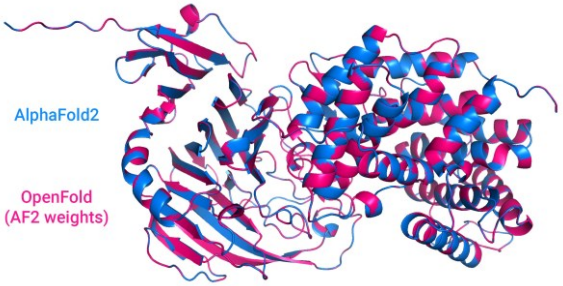
T1049 / 6y4f
93.3 GDT
(adhesin tip)

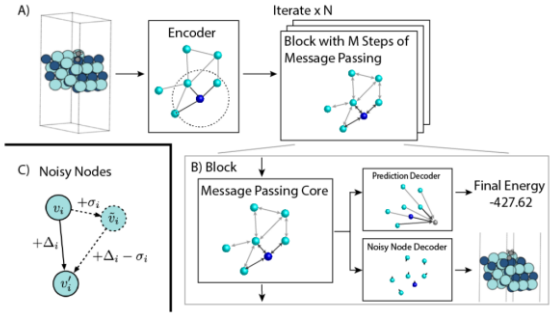
Geometric Deep Learning in Computational Biology



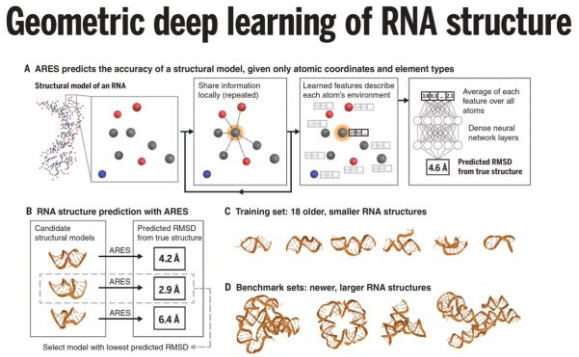
Geometric Deep Learning in Computational Biology

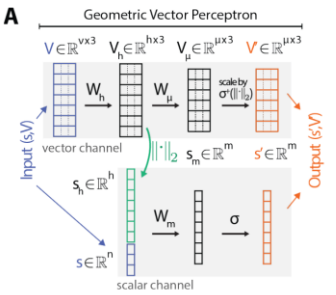


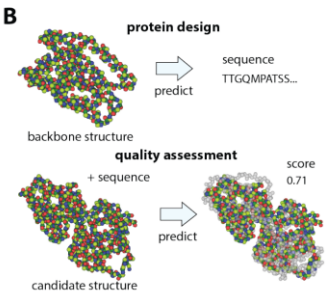




ATOM3D







Graphein

- 1: Overview
- 2: Preliminaries
- 3. DeepInteract & The Geometric Transformer
- 4: Results & Conclusions

Notation & Problem Formulation

- For each protein chain, define a graph $\mathbb{G} :< \mathbb{N}, \mathbb{E}, k >$.
  - $\mathbb{N}$: Atoms in the chain (simplified to the number of Ca atoms/amino acid residues in the chain)
  - $\mathbb{E}$: Edges between Ca atoms
  - k : Number of nearest neighbors to which to connect each Ca atom
    - In this work, we let k = 20 similar to [FBSB 17] and [IGBJ 19]
- Our learning task then is three-fold:
  - 1. Learn new node-level representations $h_{\mathbb{A}} \in \mathbb{R}^{\mathbb{A} \times k}$ and $h_{\mathbb{B}} \in \mathbb{R}^{\mathbb{B} \times k}$ for a chain pair.
  - 2. Channel-wise interleave $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$ into interaction tensor $\mathbb{I} \in \mathbb{R}^{\mathbb{A} \times \mathbb{B} \times 2\mathbb{C}}$, where $\mathbb{A} \in \mathbb{R}$ and $\mathbb{B} \in \mathbb{R}$ are the numbers of amino acid residues in the first and second input protein chains, respectively, and $\mathbb{C} \in \mathbb{R}$ is the number of hidden channels in both $h_{\mathbb{A}}$ and $h_{\mathbb{B}}$.
  - 3. Convolve over $\mathbb{I} \in \mathbb{R}^{\mathbb{A} \times \mathbb{B} \times 2\mathbb{C}}$ to predict interface residue-residue contact probabilities.

Datasets

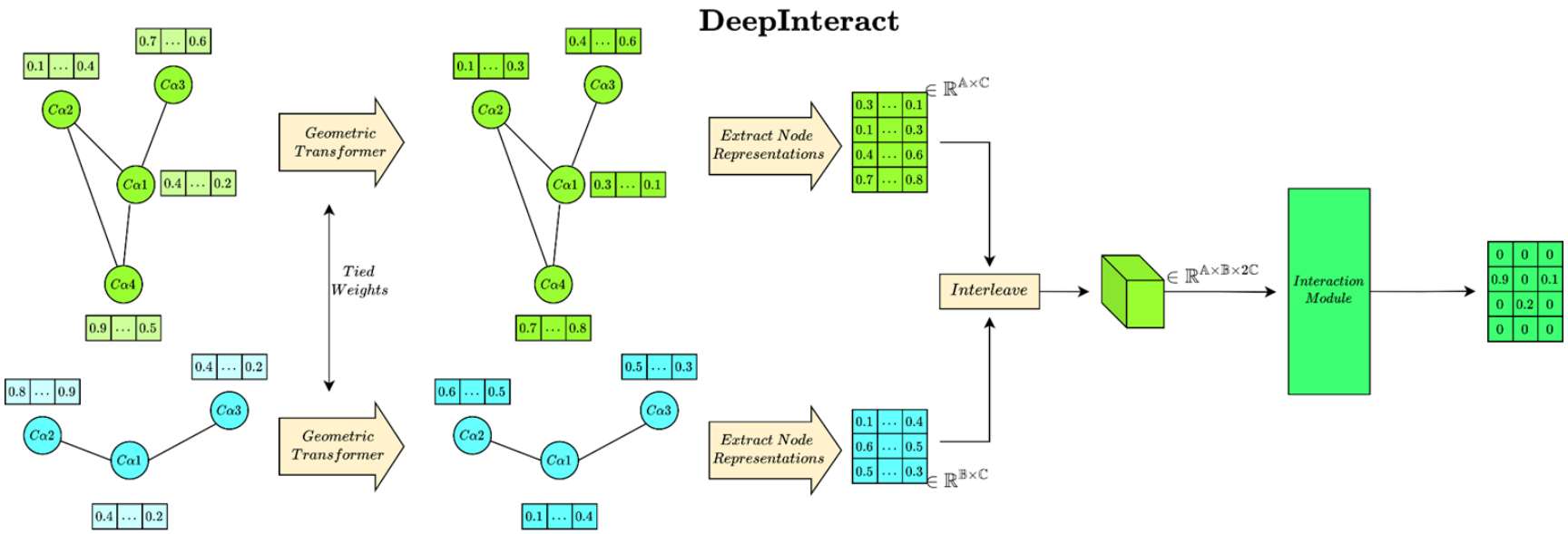
- We derive our training and validation datasets as well as our first test dataset from the enhanced version of the Database of Interacting Protein Structures (DIPS) [TBSD 19], referred to as DIPS-Plus [MCSC 21]
  - After preprocessing all proteins in DIPS-Plus, we are left with 15,618 and 3,548 pairs of protein chains for training and validation, respectively
  - Our first test dataset is comprised of 32 randomly-chosen homodimers and heterodimers from DIPS-Plus (16 of each type, respectively)
- Our second test dataset consists of 14 homodimers and 5 heterodimers from CASP-CAPRI 13-14 [LBNV 19] [LBMN 21]
- Finally, our third test dataset includes the traditional 55 chain pairs for interface contact prediction from Docking Benchmark 5 (DB5) [VMVP 15]

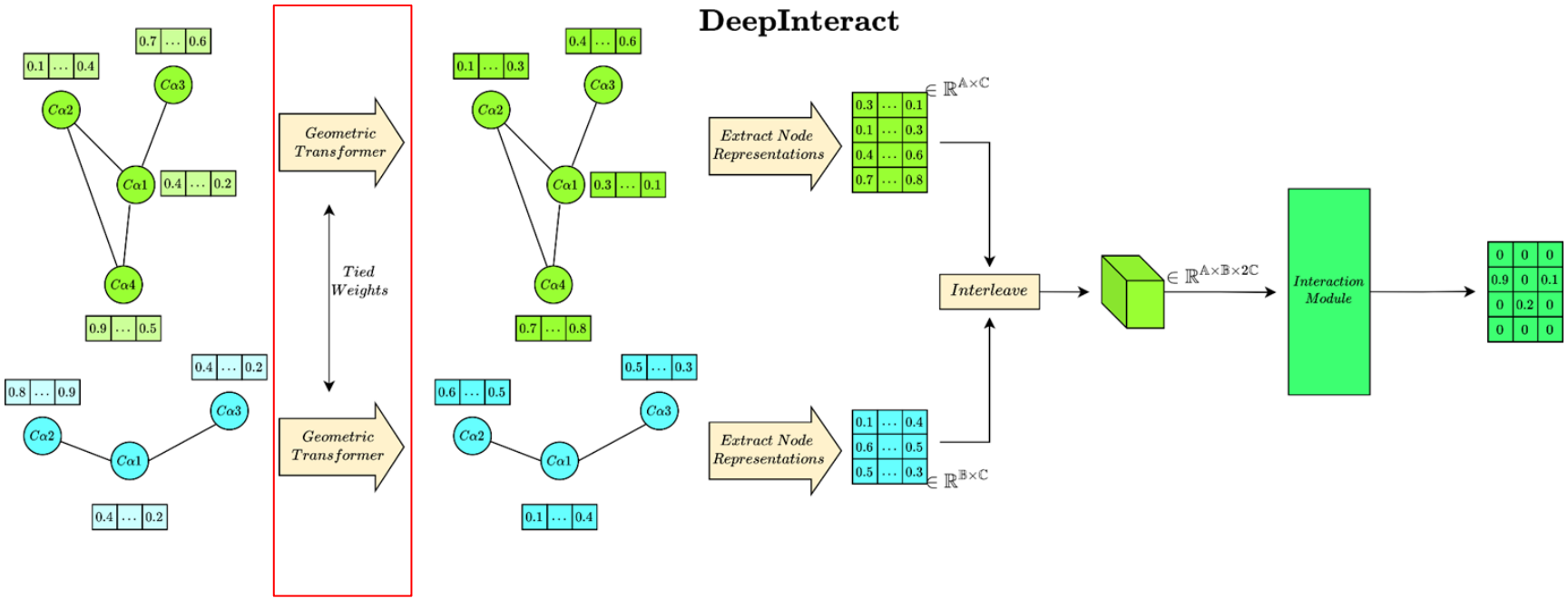
Key Ideas behind DeepInteract

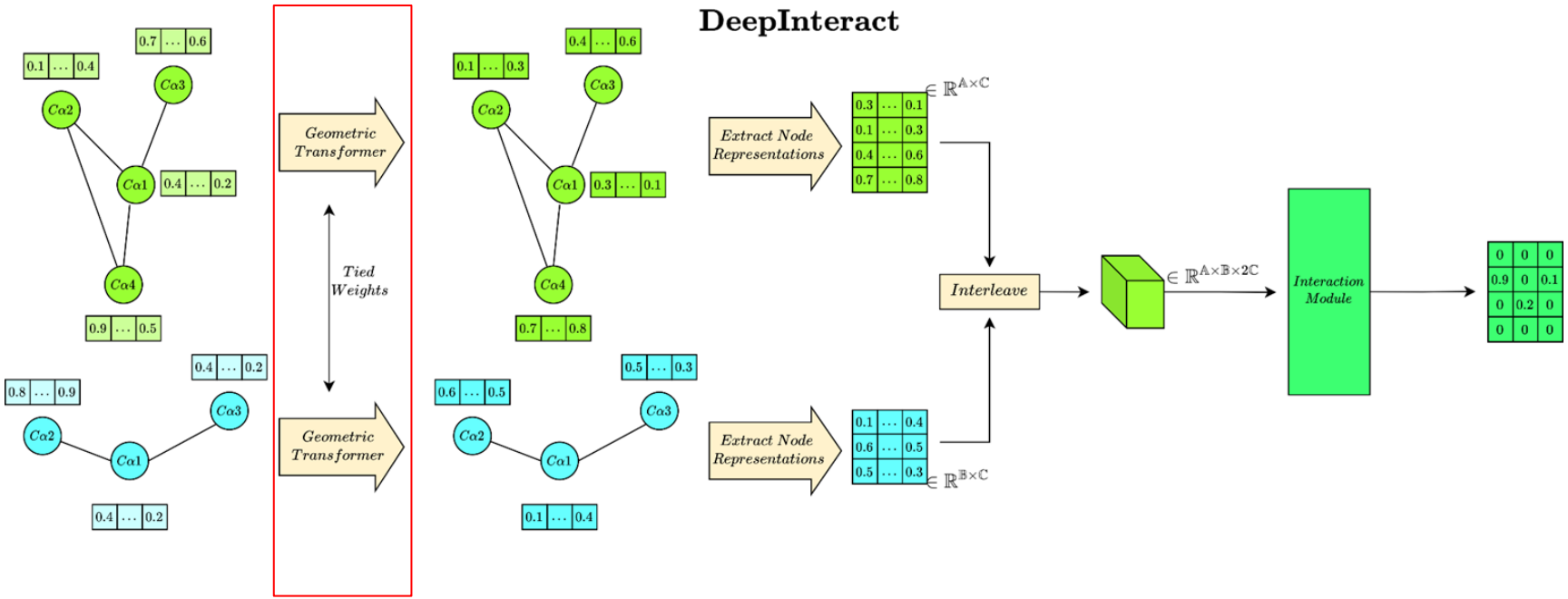
- Can we use graph self-attention and geometric feature gating (GFG) to evolve an initial set of geometric protein descriptors?

- Are these new residue representations useful for a deep CNN to more easily distinguish between interacting and non-interacting inter-chain pairs of residues?

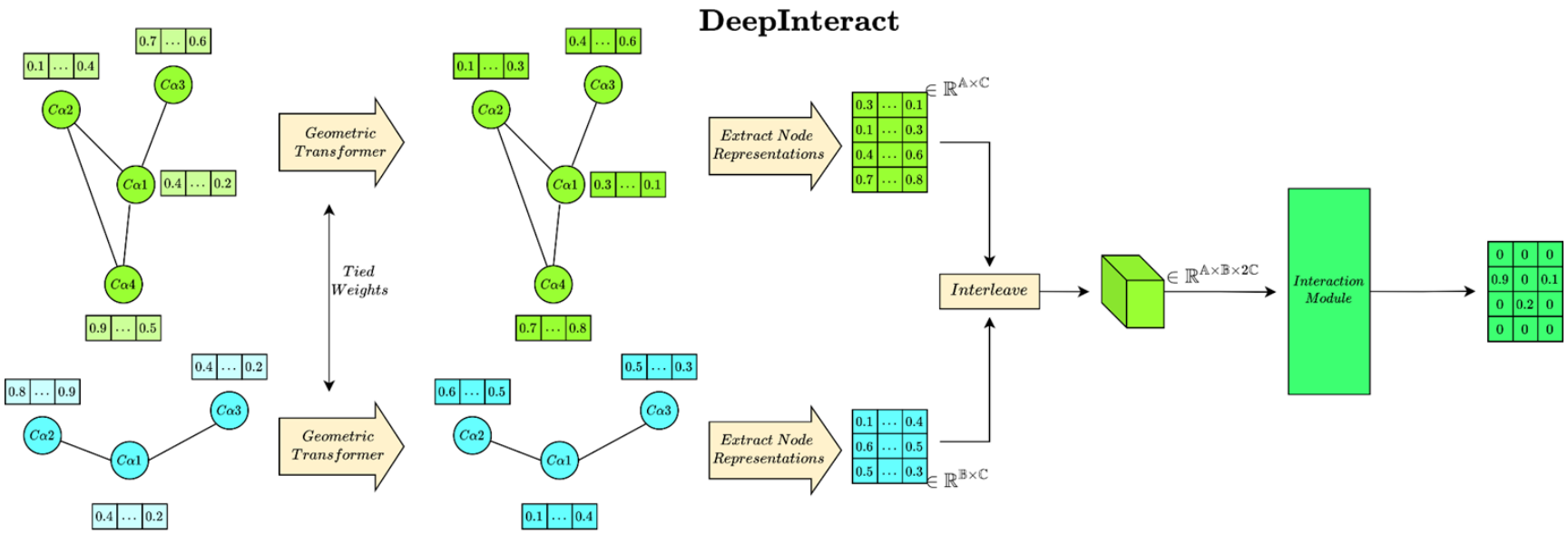
- Can alternative forms of message passing enable better learning over 3D molecular graphs?

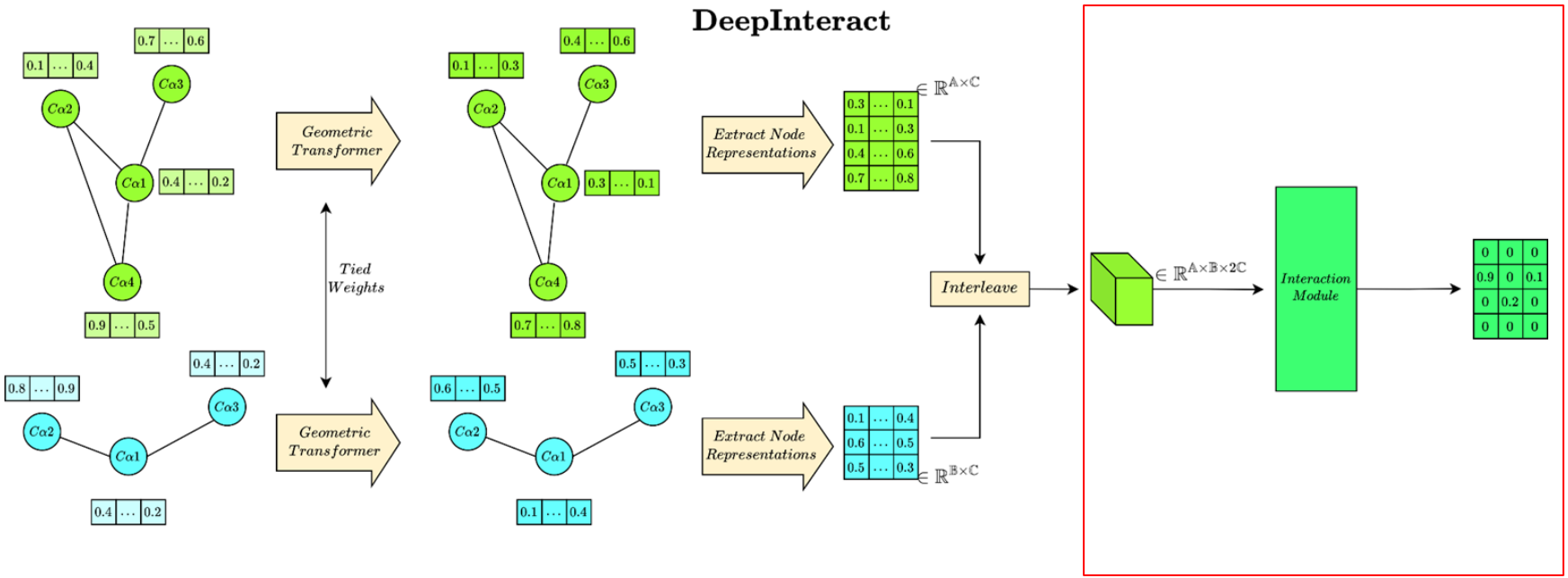

DeepInteract
Cα1
Cα2
Cα3
Cα4
Geometric Transformer
Tied Weights
Geometric Transformer
Extract Node Representations
Extract Node Representations
Interleave
Interaction Module

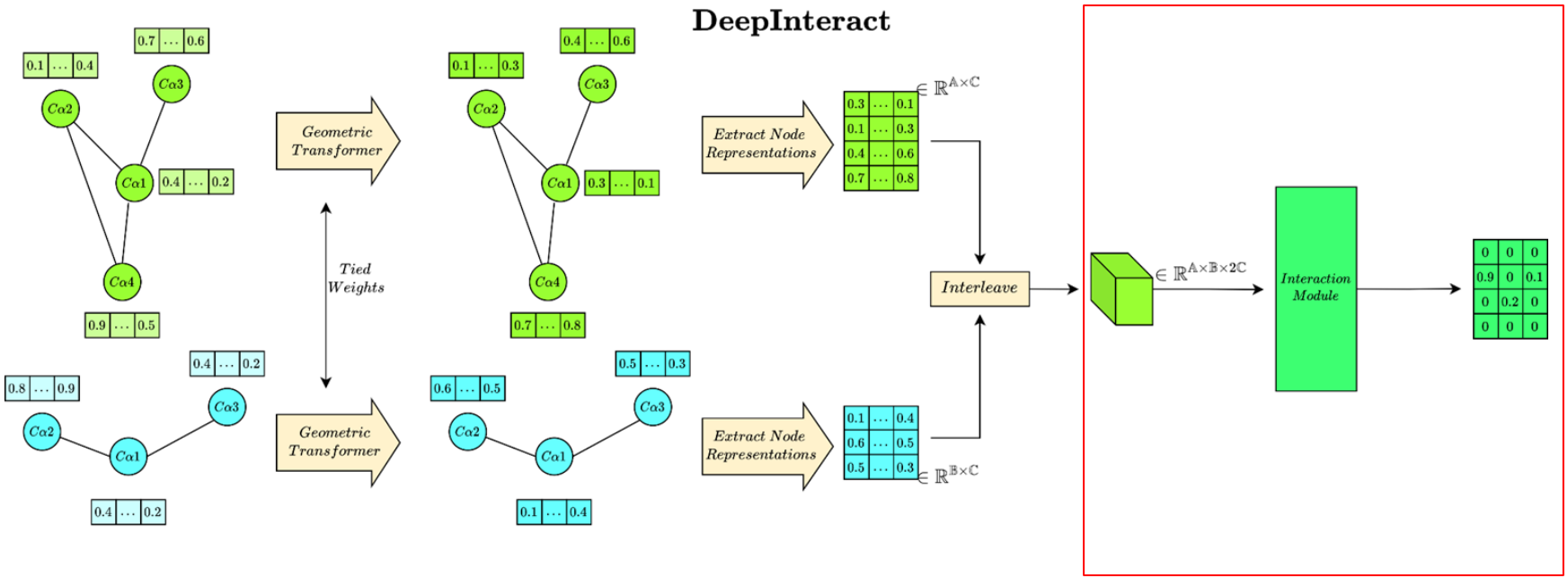

DeepInteract
Cα1
Cα2
Cα3
Cα4
Geometric Transformer
Tied Weights
Geometric Transformer
Extract Node Representations
Extract Node Representations
Interleave
Interaction Module



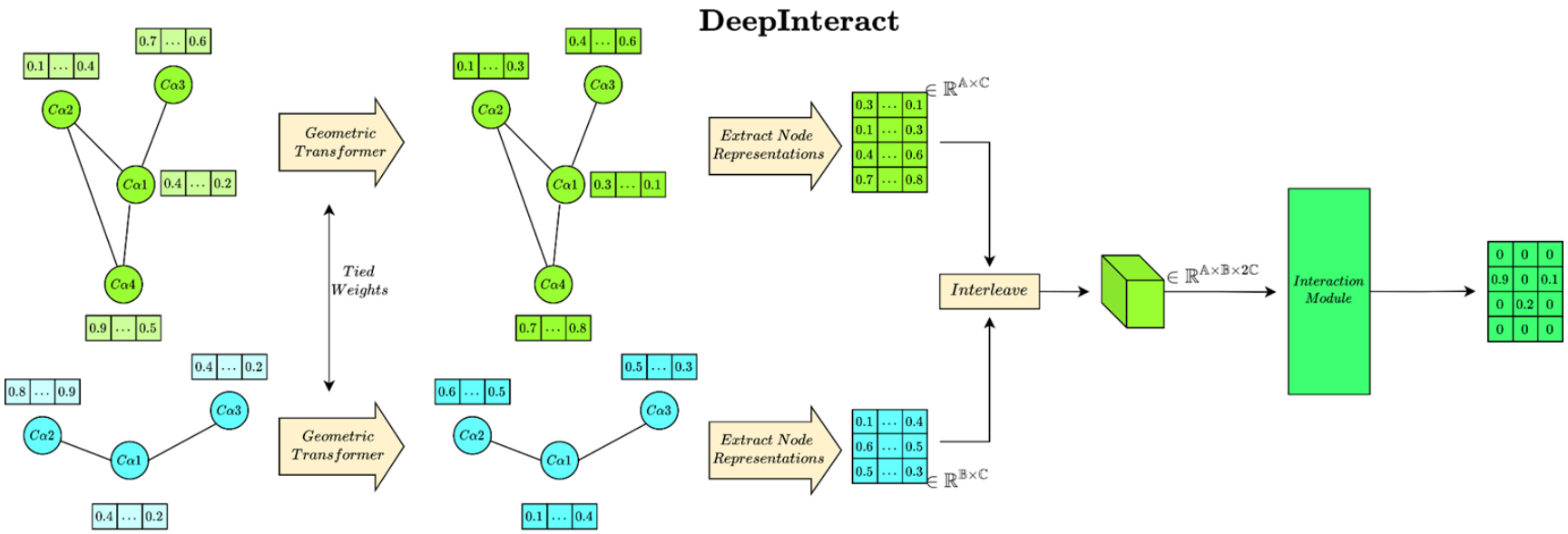
The Geometric Transformer here
learns new atom (i.e., node) representations


DeepInteract
Cα1
Cα2
Cα3
Cα4
Geometric Transformer
Tied Weights
Geometric Transformer
Extract Node Representations
Extract Node Representations
Interleave
Interaction Module

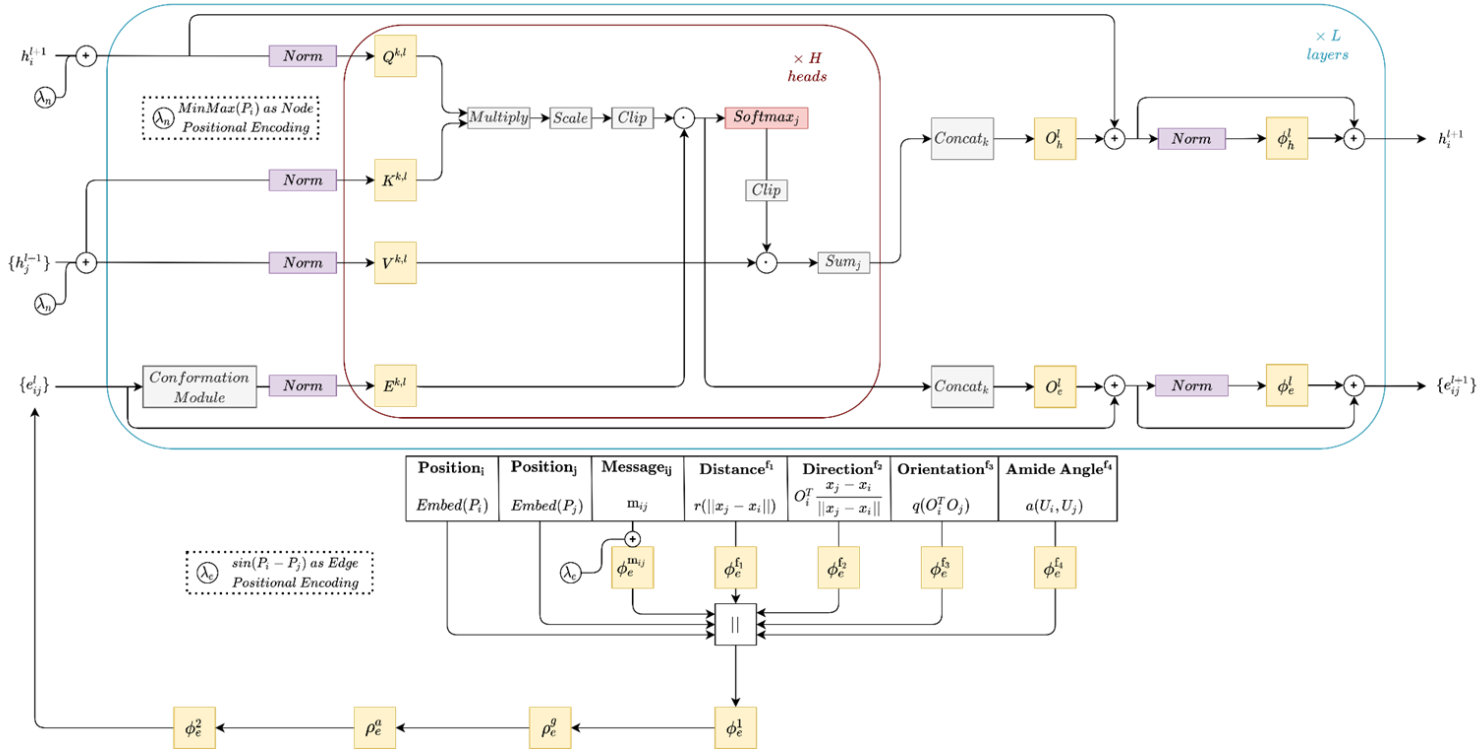

DeepInteract
Cα1
Cα2
Cα3
Cα4
Geometric Transformer
Tied Weights
Geometric Transformer
Extract Node Representations
Extract Node Representations
Interleave
Interaction Module



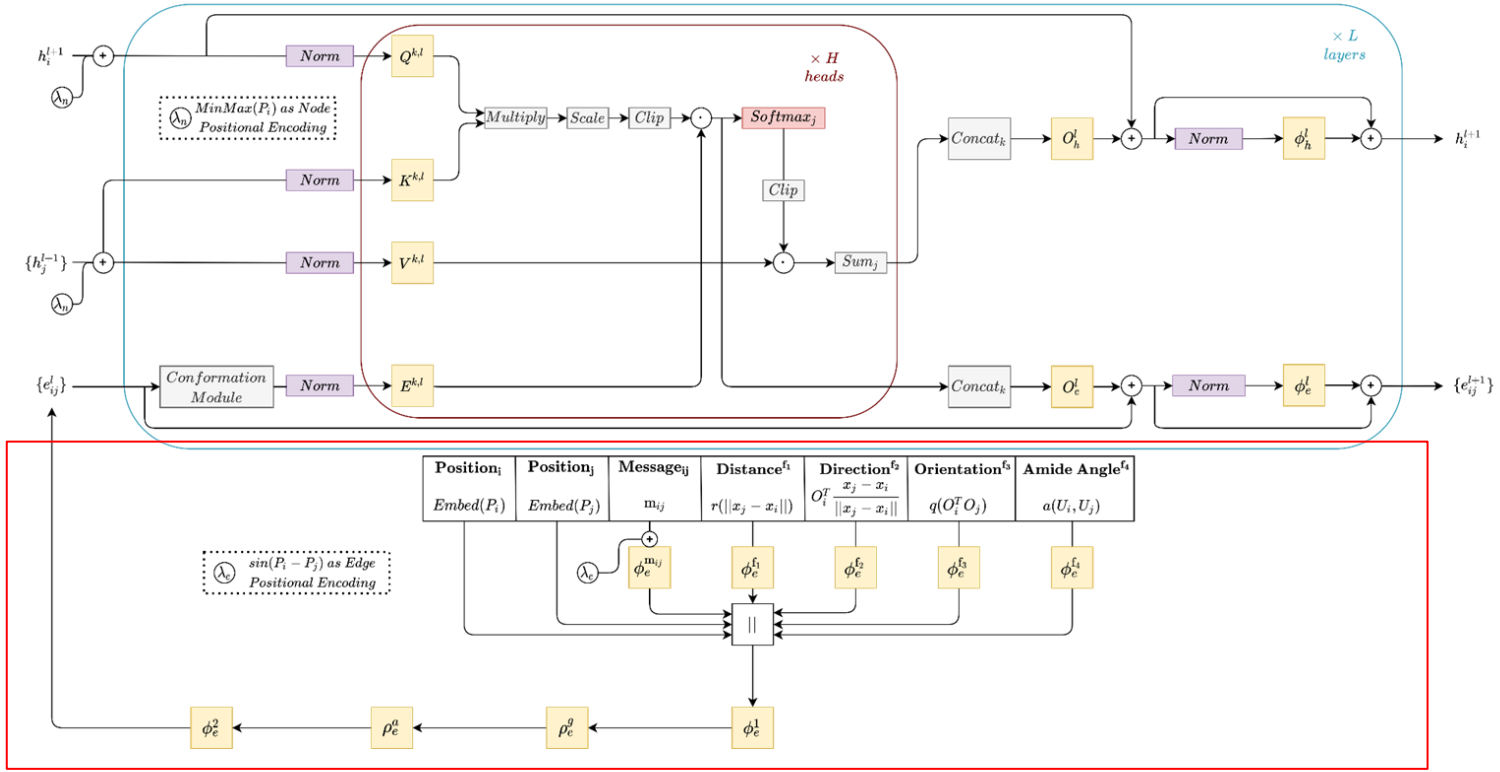
We then feed the new atom representations into a deep CNN to get contact probabilities as output

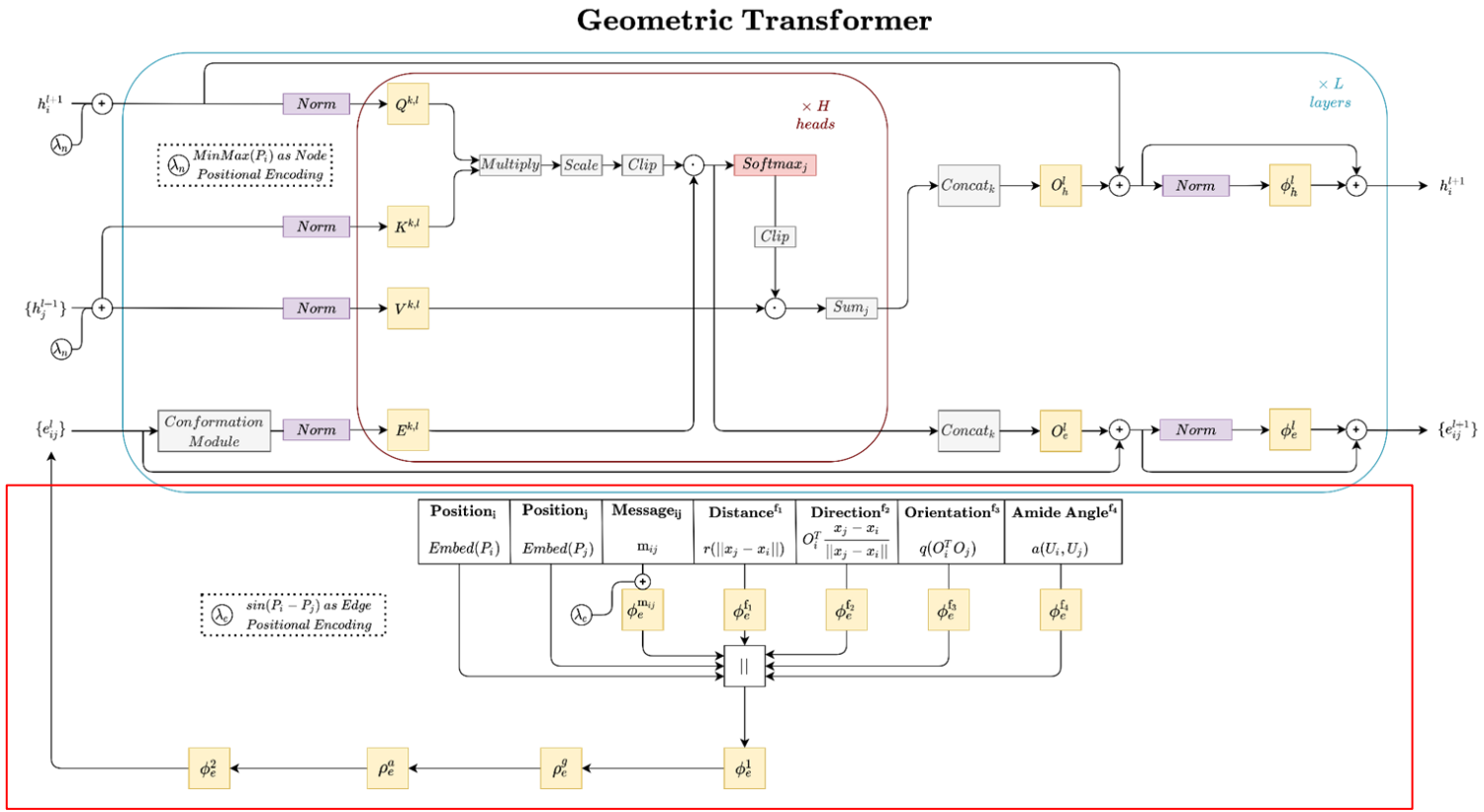

DeepInteract
Cα1
Cα2
Cα3
Cα4
Geometric Transformer
Tied Weights
Geometric Transformer
Extract Node Representations
Extract Node Representations
Interleave
Interaction Module

Geometric Transformer



Geometric Transformer





We introduce an Edge Initialization Module to accelerate the Geometric Transformer's training

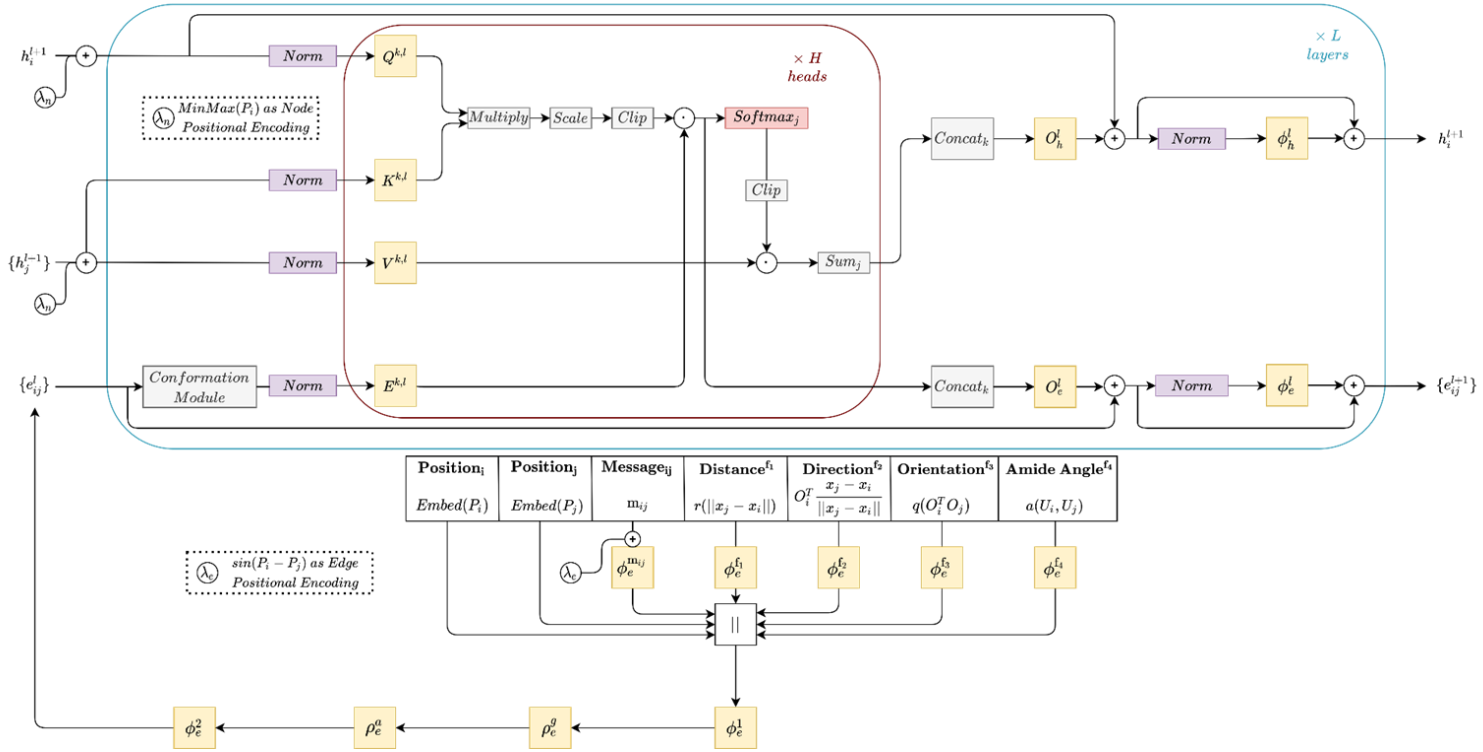
Edge Initialization Module - Definition

$$c_{ij} = \phi_e^1([\mathrm{p}_1 \,||\, \mathrm{p}_2 \,||\, \phi_e^{\mathrm{m}_{ij}}(\mathrm{m}_{ij} \,||\, \lambda_e) \,||\, \phi_e^{\mathrm{f}_1}(\mathrm{f}_1) \,||\, \phi_e^{\mathrm{f}_2}(\mathrm{f}_2) \,||\, \phi_e^{\mathrm{f}_3}(\mathrm{f}_3) \,||\, \phi_e^{\mathrm{f}_4}(\mathrm{f}_4)]) \quad (1)$$

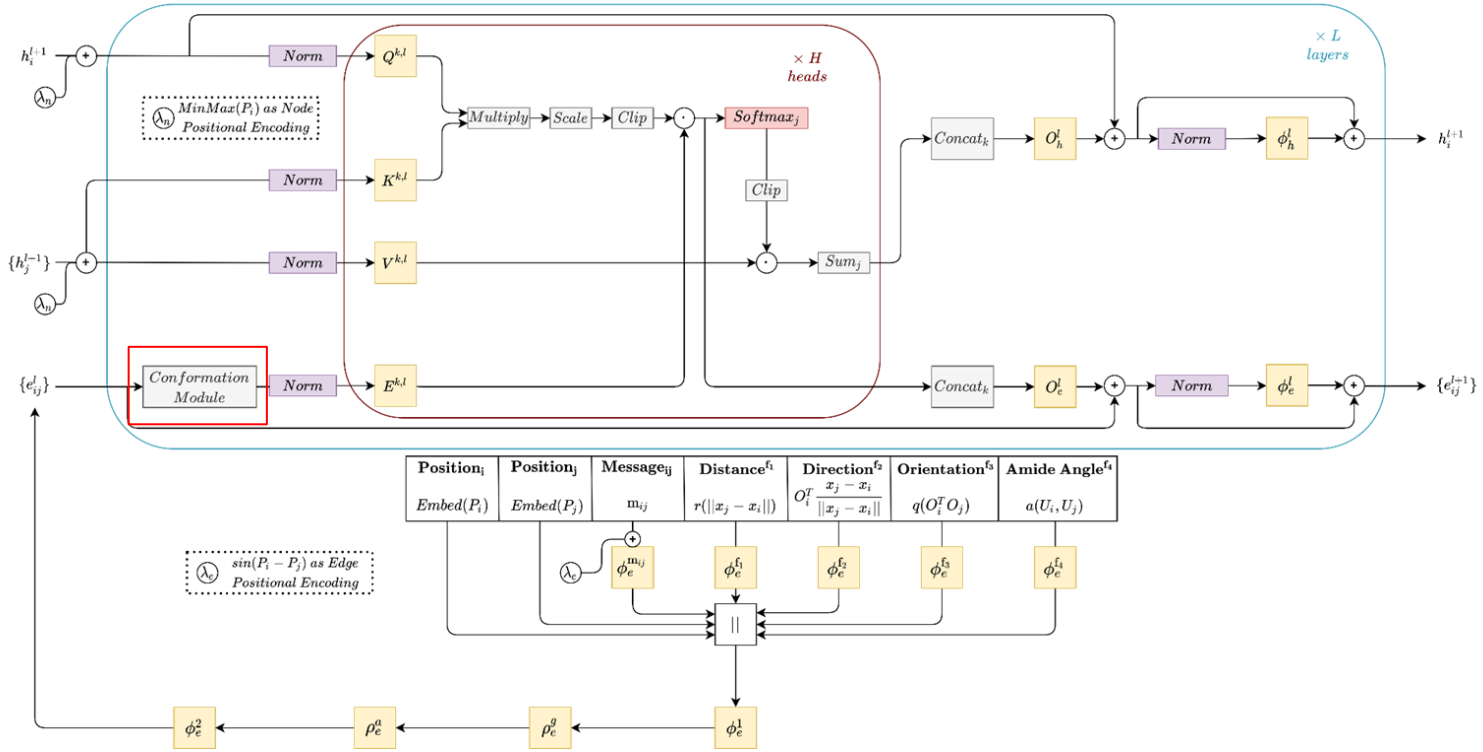
$$e_{ij} = \phi_e^2(\rho_e^a(\rho_e^g(c_{ij}))) \quad (2)$$

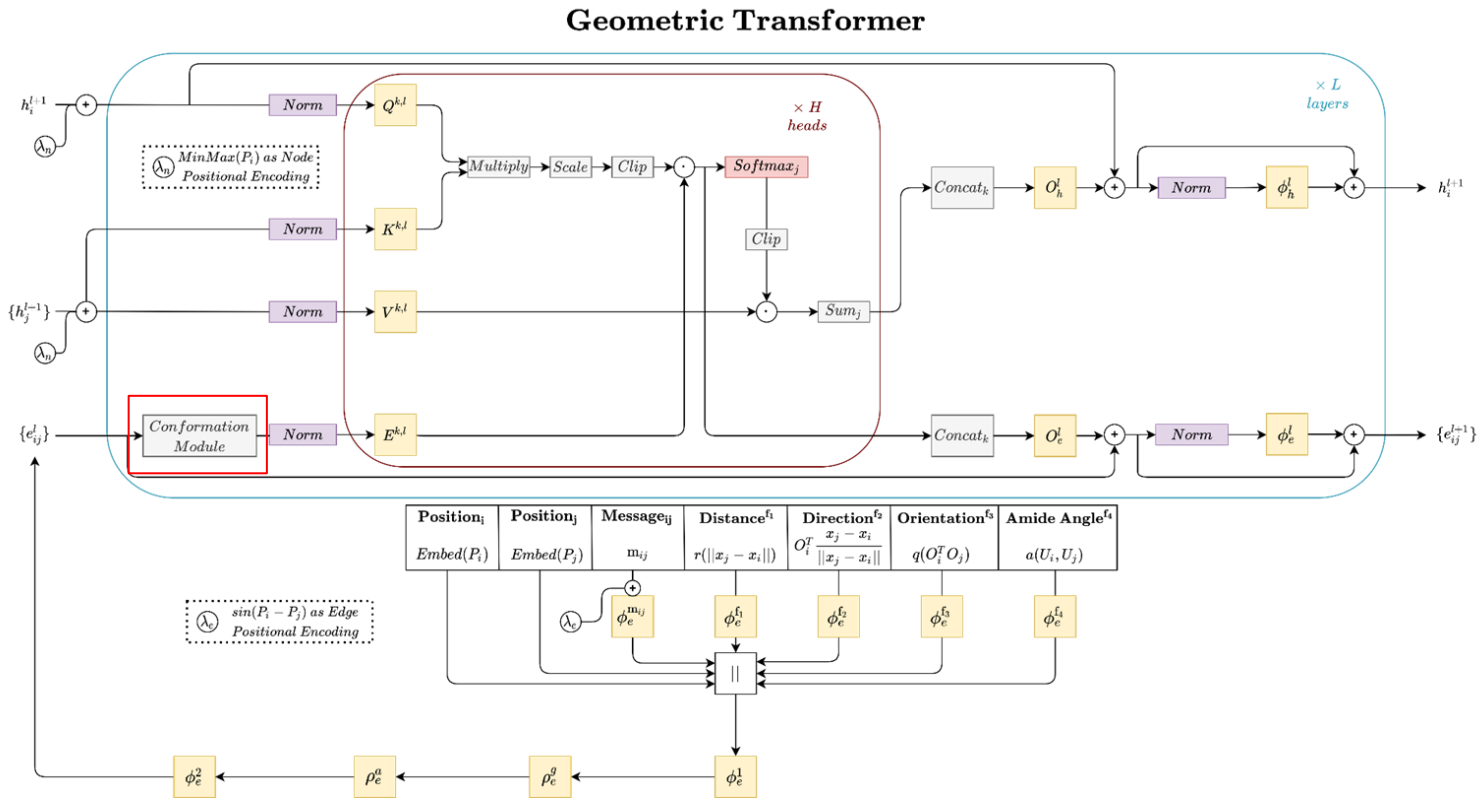
1. ϕ_e^i : The $i'th$ edge information update function such as a multi-layer perceptron
2. $||$: Channel-wise concatenation
3. p_1 and p_2 : Trainable one-hot vectors indexed, respectively, by P_i and P_j, the positions of nodes i and nodes j in the chain's underlying amino acid sequence
4. m_{ij} : Any user-predefined features for e (e.g., Euclidean distances between nodes i and nodes j)
5. λ_e : *Edge-wise* sinusoidal positional encodings $sin(P_i - P_j)$ for e
6. f_1,f_2, f_3 , and f_4: Our four protein-specific geometric features, in order
7. ρ_e^a and ρ_e^g : Feature addition and channel-wise gating functions, respectively

Geometric Transformer

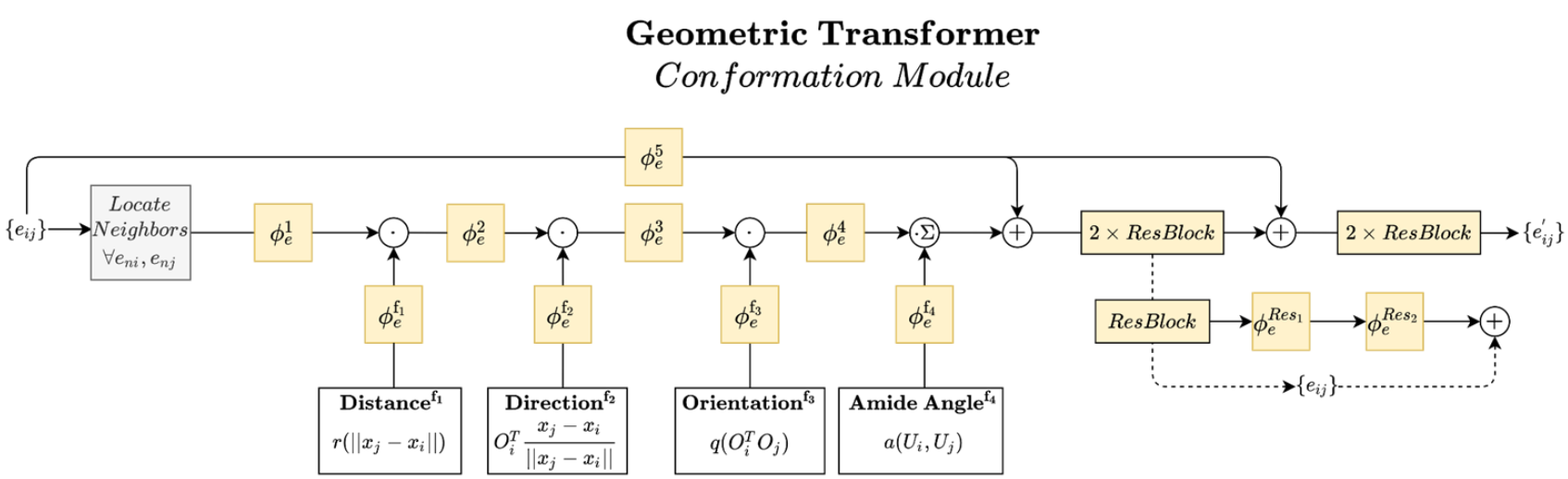


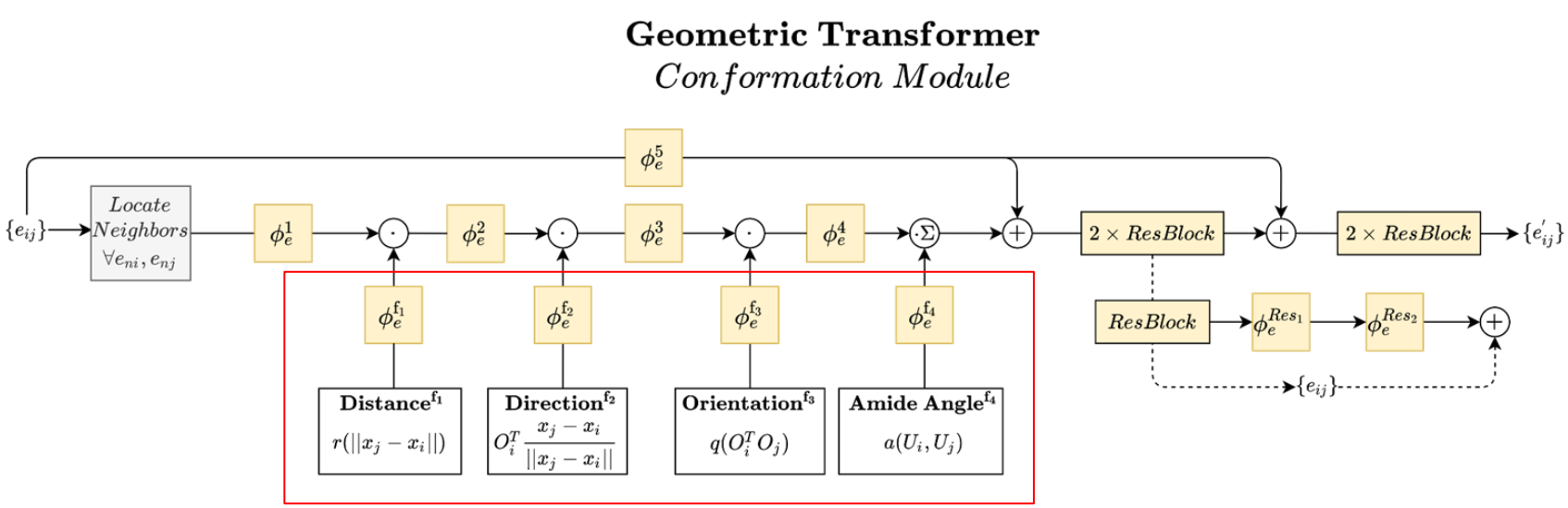
Geometric Transformer

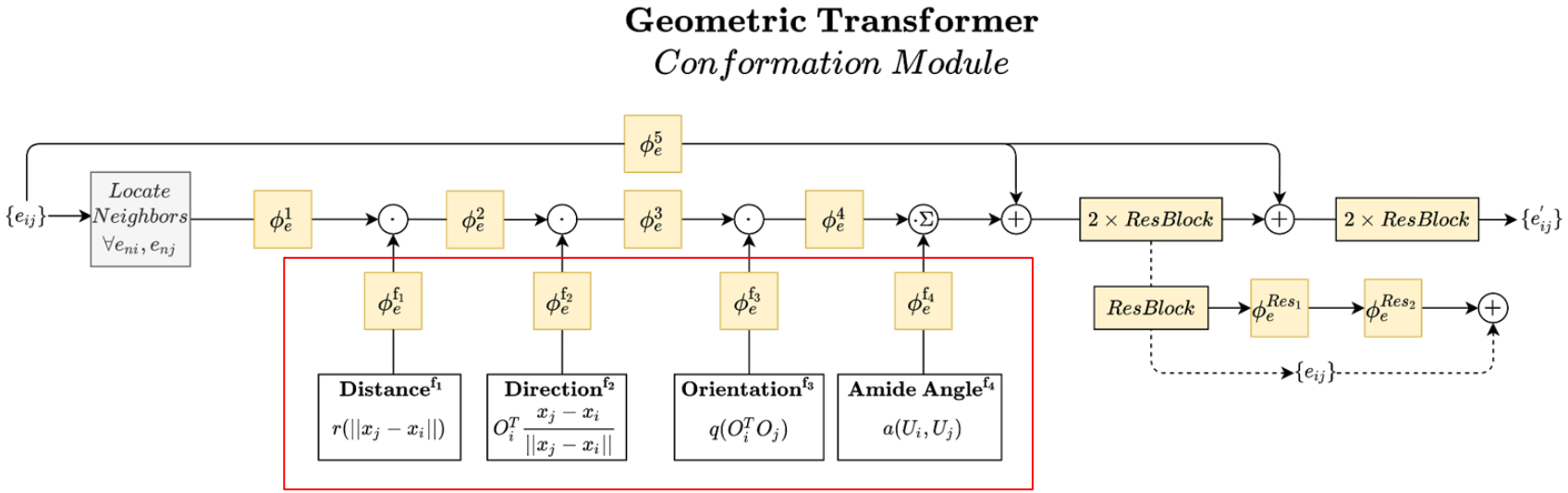




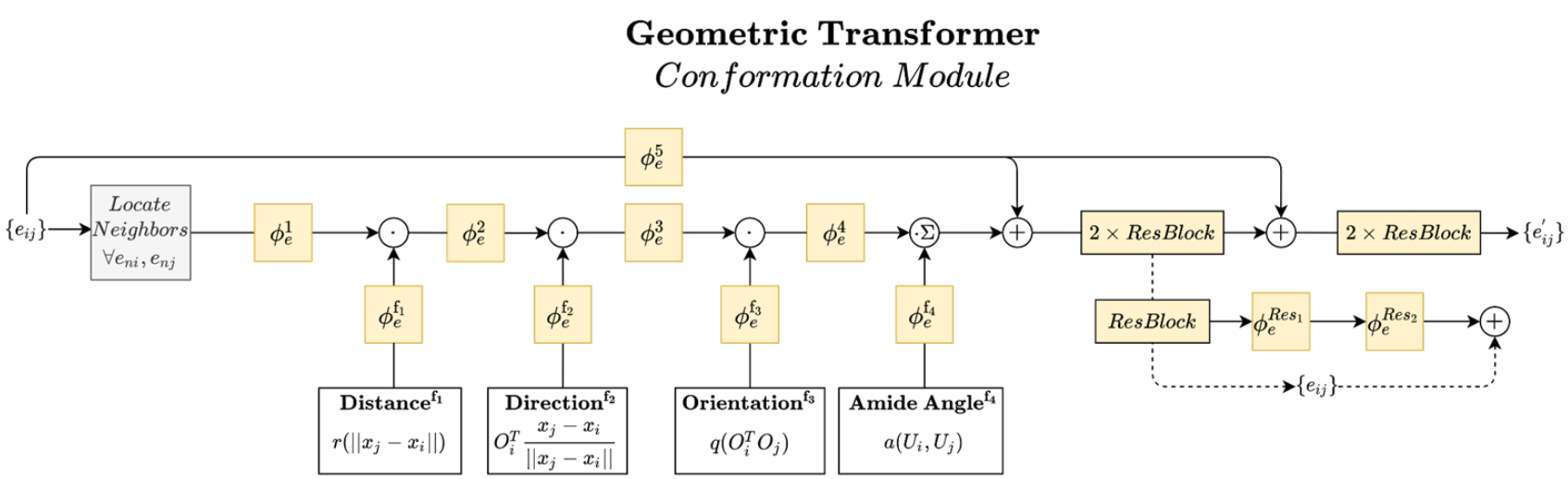
We also introduce a Conformation Module that learns to *evolve* protein representations using GFG


Geometric Transformer
Conformation Module
$\{e_{ij}\}$
Locate Neighbors $\forall e_{ni}, e_{nj}$
ϕ_e^1
ϕ_e^2
ϕ_e^3
ϕ_e^4
ϕ_e^5
$\phi_e^{f_1}$
$\phi_e^{f_2}$
$\phi_e^{f_3}$
$\phi_e^{f_4}$
Distance^{f_1} $r(||x_j - x_i||)$
Direction^{f_2} $O_i^T \frac{x_j - x_i}{||x_j - x_i||}$
Orientation^{f_3} $q(O_i^T O_j)$
Amide Angle^{f_4} $a(U_i, U_j)$
$2 \times ResBlock$
$2 \times ResBlock$
$\{e'_{ij}\}$
$ResBlock$
$\phi_e^{Res_1}$
$\phi_e^{Res_2}$
$\{e_{ij}\}$


Geometric Transformer
Conformation Module
$\{e_{ij}\}$
Locate Neighbors $\forall e_{ni}, e_{nj}$
ϕ_e^5
ϕ_e^1
ϕ_e^2
ϕ_e^3
ϕ_e^4
$\cdot$
$\cdot\Sigma$
$+$
$2 \times ResBlock$
$2 \times ResBlock$
$\{e'_{ij}\}$
$\phi_e^{f_1}$
$\phi_e^{f_2}$
$\phi_e^{f_3}$
$\phi_e^{f_4}$
Distance^{f_1}
$r(||x_j - x_i||)$
Direction^{f_2}
$O_i^T \frac{x_j - x_i}{||x_j - x_i||}$
Orientation^{f_3}
$q(O_i^T O_j)$
Amide Angle^{f_4}
$a(U_i, U_j)$
$ResBlock$
$\phi_e^{Res_1}$
$\phi_e^{Res_2}$
$\{e_{ij}\}$

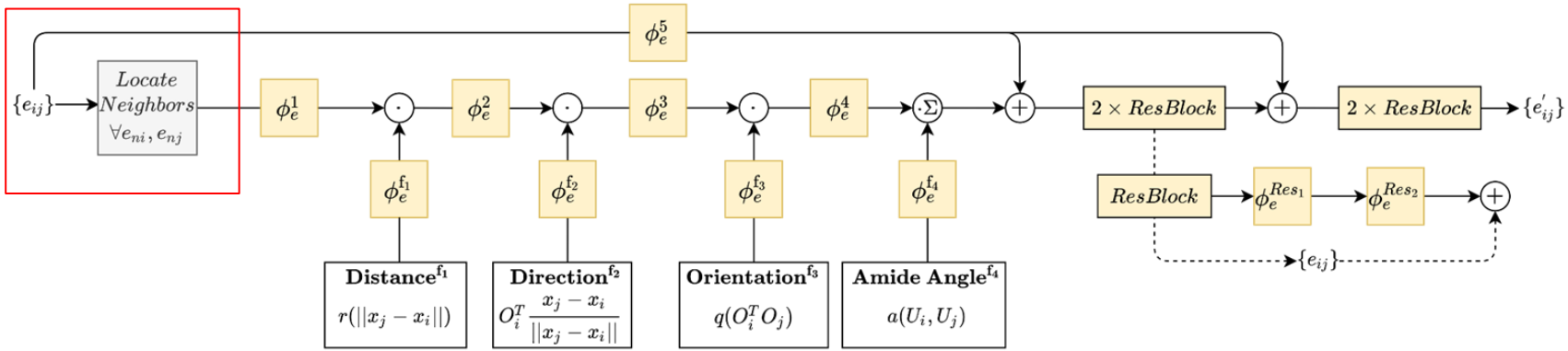


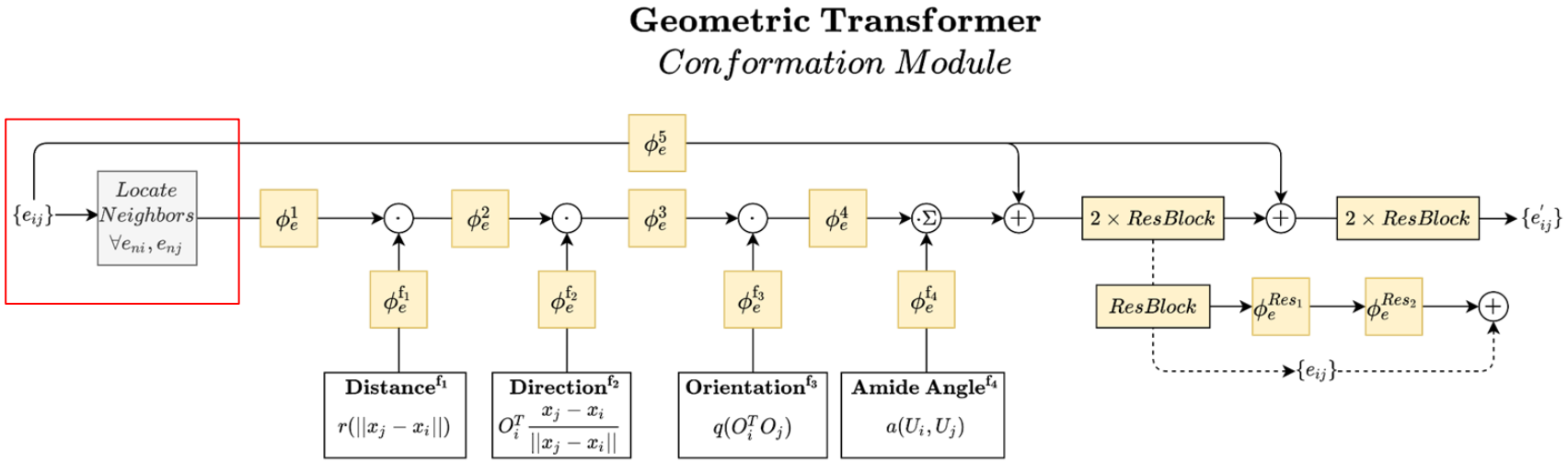
Here, the Conformation Module learns to selectively gate each type of geometric information, similar to work overcoming vanishing gradients with RNNs


Geometric Transformer
Conformation Module
$\{e_{ij}\}$
Locate Neighbors $\forall e_{ni}, e_{nj}$
ϕ_e^1
ϕ_e^2
ϕ_e^3
ϕ_e^4
ϕ_e^5
$\phi_e^{f_1}$
$\phi_e^{f_2}$
$\phi_e^{f_3}$
$\phi_e^{f_4}$
Distance^{f_1} $r(||x_j - x_i||)$
Direction^{f_2} $O_i^T \frac{x_j - x_i}{||x_j - x_i||}$
Orientation^{f_3} $q(O_i^T O_j)$
Amide Angle^{f_4} $a(U_i, U_j)$
$2 \times ResBlock$
$2 \times ResBlock$
$\{e'_{ij}\}$
$ResBlock$
$\phi_e^{Res_1}$
$\phi_e^{Res_2}$
$\{e_{ij}\}$

Geometric Transformer

Conformation Module





Moreover, it does so by treating an edge's "neighboring" edges as pseudo-nodes, offering the edge a wider receptive field with which to update its representations

Conformation Module - Definition (1)

$$\mathbb{E}_k = \{e_{n_1 i}, e_{n_2 j} \mid (n_1, n_2 \in \mathbb{N}_k) \text{ and } (n_1, n_2 \neq i, j)\} \tag{3}$$

1. $\mathbb{E}_k$: The edge geometric neighborhood of edge (i.e., pseudo-node) e, defined as the $2n$ edges above in (3)
2. $\mathbb{N}_k \subset \mathbb{N}$: The source nodes for incoming edges on $e's$ source and destination nodes

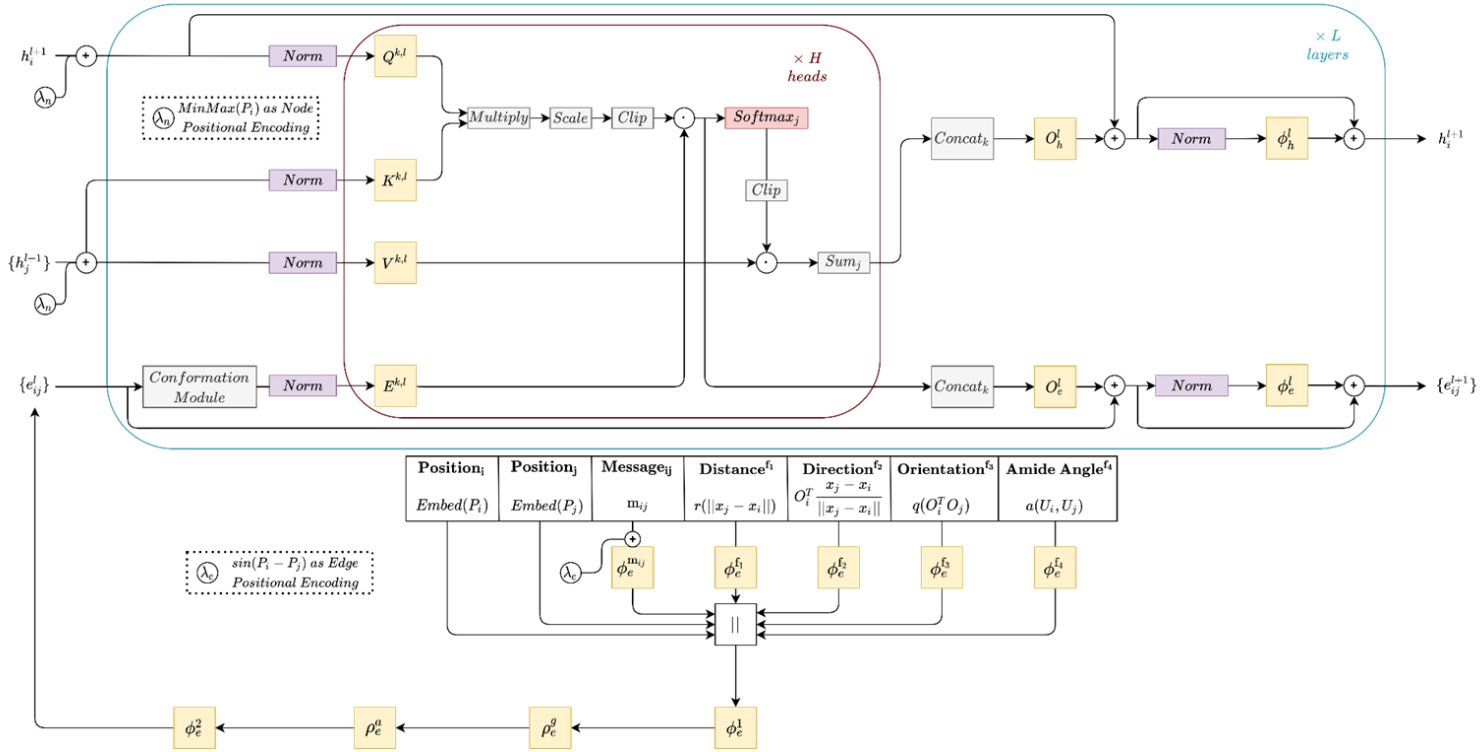
Conformation Module - Definition (2)

$$O_{ij} = \sum_{k \in \mathbb{E}_k} [(\phi_e^n(e_{ij,k}^n) \odot \phi_e^{\mathrm{f}_n}(\mathrm{f}_n)), \forall n \in \mathbb{F}] \tag{4}$$

$$e_{ij} = 2 \times ResBlock_2(\phi_e^5(e_{ij}) + 2 \times ResBlock_1(\phi_e^5(e_{ij}) + O_{ij})) \tag{5}$$

1. $\mathbb{F}$: The set of our protein geometric features
2. $\odot$: Element-wise multiplication
3. $e_{ij,k}^n$: Neighboring edge $e_k's$ representation after being gated with $\mathrm{f}_{\mathrm{n}-1}$
4. $2 \times ResBlock_i$: The $i'th$ application of two unique, successive residual blocks, each defined as $ResBlock(x) = \phi_e^{Res_2}(\phi_e^{Res_1}(x)) + x$

Geometric Transformer



- 1: Overview
- 2: Preliminaries
- 3: DeepInteract & The Geometric Transformer
- 4. Results & Conclusion

Test Results on DIPS-Plus Targets

Table 1: The average top-k precision on two types of DIPS-Plus test targets.

	16 (Homo)			16 (Hetero)		
Method	10	$L/10$	$L/5$	10	$L/10$	$L/5$
BI	0	0	0	0.02	0.02	0.02
DH	0.13	0.12	0.09			
CC				**0.17**	**0.16**	**0.15**
DI (GCN)	0.22 (0.06)	0.20 (0.07)	0.18 (0.04)	0.08 (0.01)	0.08 (0.01)	0.07 (0.02)
DI (GT)	0.27 (0.06)	0.24 (0.04)	0.21 (0.04)	0.10 (0.04)	0.09 (0.04)	0.08 (0.04)
DI (GeoT w/o EPE)	**0.28 (0.05)**	0.24 (0.01)	0.23 (0.03)	0.11 (0.05)	0.10 (0.04)	0.09 (0.03)
DI (GeoT w/o GFG)	0.27 (0.08)	0.24 (0.08)	0.21 (0.08)	0.10 (0.02)	0.09 (0.02)	0.09 (0.01)
DI (GeoT)	0.25 (0.03)	**0.25 (0.03)**	**0.23 (0.02)**	0.15 (0.04)	0.14 (0.05)	0.11 (0.04)

Table 2: The average top-k precision and recall on DIPS-Plus test targets of both types.

	32 (Both Types)					
Method	P@10	P@$L/10$	P@$L/5$	R@L	R@$L/2$	R@$L/5$
BI	0.01	0.01	0.01	0.01	0.004	0.003
DI (GCN)	0.15 (0.03)	0.16 (0.01)	0.12 (0.02)	0.10 (0.02)	0.06 (0.01)	0.03 (0.003)
DI (GT)	0.18 (0.05)	0.16 (0.04)	0.15 (0.04)	0.13 (0.02)	0.07 (0.01)	0.04 (0.01)
DI (GeoT w/o EPE)	0.19 (0.04)	0.18 (0.03)	0.16 (0.03)	0.14 (0.02)	0.08 (0.02)	0.04 (0.02)
DI (GeoT w/o GFG)	0.18 (0.05)	0.16 (0.04)	0.15 (0.04)	0.14 (0.02)	0.08 (0.02)	0.04 (0.01)
DI (GeoT)	**0.20 (0.01)**	**0.19 (0.01)**	**0.17 (0.02)**	**0.15 (0.003)**	**0.09 (0.004)**	**0.04 (0.002)**

Test Results on CASP-CAPRI 13-14 Targets

Table 3: The average top-k precision on dimers from CASP-CAPRI 13 & 14.

	14 (Homo)			5 (Hetero)		
Method	10	$L/10$	$L/5$	10	$L/10$	$L/5$
BI	0	0	0	0.04	0	0.03
DH	0.02	0.02	0.02			
CC				0.06	0.08	0.05
DI (GCN)	0.12 (0.04)	0.11 (0.03)	**0.13 (0.02)**	0.10 (0.07)	0.11 (0.08)	0.09 (0.04)
DI (GT)	0.08 (0.03)	0.09 (0.05)	0.08 (0.03)	0.14 (0.02)	0.14 (0.02)	0.12 (0.03)
DI (GeoT w/o EPE)	0.11 (0.01)	0.12 (0.02)	0.11 (0.01)	0.18 (0.07)	0.20 (0.09)	0.18 (0.04)
DI (GeoT w/o GFG)	0.10 (0.02)	0.10 (0.02)	0.09 (0.02)	0.14 (0.03)	0.17 (0.03)	0.14 (0.02)
DI (GeoT)	**0.18 (0.05)**	**0.13 (0.03)**	0.11 (0.02)	**0.30 (0.09)**	**0.31 (0.07)**	**0.24 (0.04)**

Table 4: The average top-k precision and recall across all targets from CASP-CAPRI 13 & 14.

	19 (Both Types)					
Method	P@10	P@$L/10$	P@$L/5$	R@L	R@$L/2$	R@$L/5$
BI	0.01	0	0.01	0.02	0.01	0.001
DI (GCN)	0.12 (0.04)	0.10 (0.05)	0.09 (0.04)	0.11 (0.001)	0.06 (0.01)	0.02 (0.01)
DI (GT)	0.10 (0.03)	0.09 (0.03)	0.08 (0.02)	0.11 (0.02)	0.06 (0.01)	0.02 (0.01)
DI (GeoT w/o EPE)	0.13 (0.02)	0.14 (0.03)	0.13 (0.02)	0.12 (0.01)	0.07 (0.01)	0.03 (0.01)
DI (GeoT w/o GFG)	0.11 (0.01)	0.12 (0.02)	0.10 (0.02)	0.11 (0.01)	0.06 (0.01)	0.03 (0.01)
DI (GeoT)	**0.21 (0.01)**	**0.19 (0.01)**	**0.14 (0.01)**	**0.13 (0.02)**	**0.08 (0.01)**	**0.04 (0.003)**

Test Results on Docking Benchmark 5 Targets

Table 5: The average top-k precision and recall on DB5 test targets.

	55 (Hetero)					
Method	P@10	P@$L/10$	P@$L/5$	R@L	R@$L/2$	R@$L/5$
BI	0	0.002	0.001	0.003	0.001	0.0004
CC	0.002	0.003	0.003	0.007	0.003	0.001
DI (GCN)	0.005 (0.002)	0.006 (0.001)	0.007 (0.001)	0.013 (0.002)	0.008 (0.001)	0.003 (0.001)
DI (GT)	0.008 (0.004)	0.008 (0.005)	0.008 (0.004)	0.010 (0.005)	0.006 (0.003)	0.003 (0.002)
DI (GeoT w/o EPE)	0.011 (0.004)	0.009 (0.004)	0.011 (0.002)	0.018 (0.01)	0.010 (0.004)	0.0034 (0.002)
DI (GeoT w/o GFG)	0.008 (0.001)	0.008 (0.001)	0.009 (0.002)	0.014 (0.01)	0.006 (0.002)	0.003 (0.001)
DI (GeoT)	**0.013 (0.001)**	**0.009 (0.003)**	**0.011 (0.001)**	**0.018 (0.001)**	**0.010 (0.001)**	**0.0034 (0.001)**

Disclaimer

- New standardized means of evaluating interface contact prediction methods are needed

- One way to do this may be to use structures generated by AlphaFold to curate a new benchmark dataset of homomeric and heteromeric targets

Conclusion

- We introduce the Geometric Transformer for predicting contact points in protein interfaces

- The Geometric Transformer lays the foundation for conducting alternative forms of message-passing on 3D graphs (e.g., treating edges as pseudo-nodes) and exploiting geometric feature gating to enhance the expressiveness of graph-like Transformers architectures

Extensions (1)

- We have already extended some ideas introduced with the Geometric Transformer (e.g., edge message-passing) to other problems in protein bioinformatics (i.e., EnQA for protein model quality assessment)

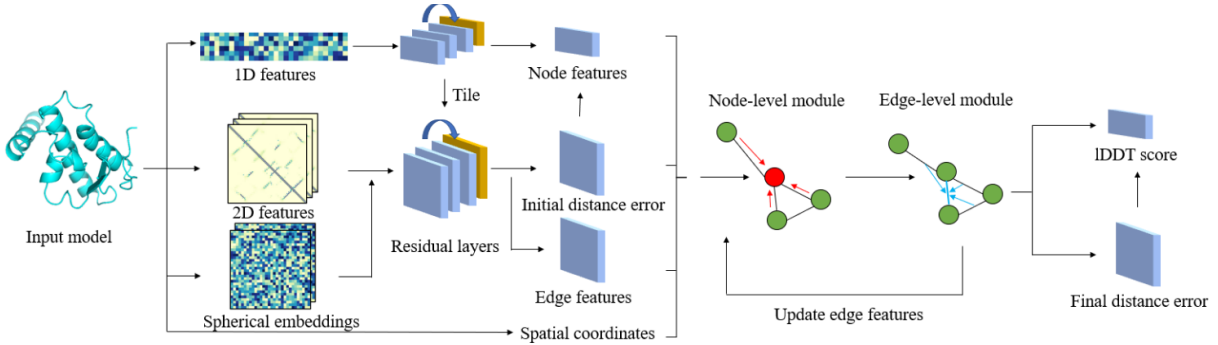


Figure 2. The illustration of the overall architecture of EnQA. The 1D/2D features from the input model are first converted into hidden node and edge features for the 3D-equivarant graph module. The spatial coordinates of Ca atoms of the residues are also used as an extra feature. The node and edge network modules update the graph features iteratively. In the end, the final per-residue lDDT score and distance errors of residue pairs are predicted from the updated node/edge features and spatial coordinates by the 3D-equivariant network.

Extensions (1)

- We have already extended some ideas introduced with the Geometric Transformer (e.g., edge message-passing) to other problems in protein bioinformatics (i.e., EnQA for protein model quality assessment)

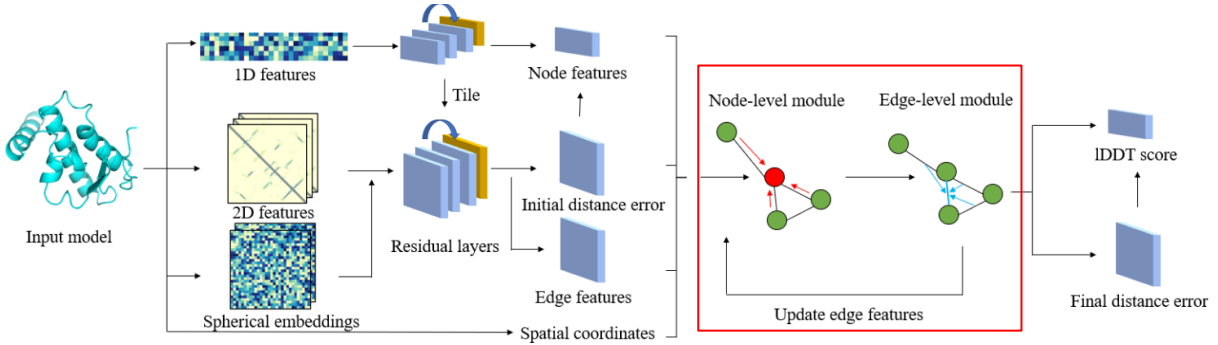


Figure 2. The illustration of the overall architecture of EnQA. The 1D/2D features from the input model are first converted into hidden node and edge features for the 3D-equivarant graph module. The spatial coordinates of Ca atoms of the residues are also used as an extra feature. The node and edge network modules update the graph features iteratively. In the end, the final per-residue lDDT score and distance errors of residue pairs are predicted from the updated node/edge features and spatial coordinates by the 3D-equivariant network.

Extensions (1)

- We have already extended some ideas introduced with the Geometric Transformer (e.g., edge message-passing) to other problems in protein bioinformatics (i.e., EnQA for protein model quality assessment)

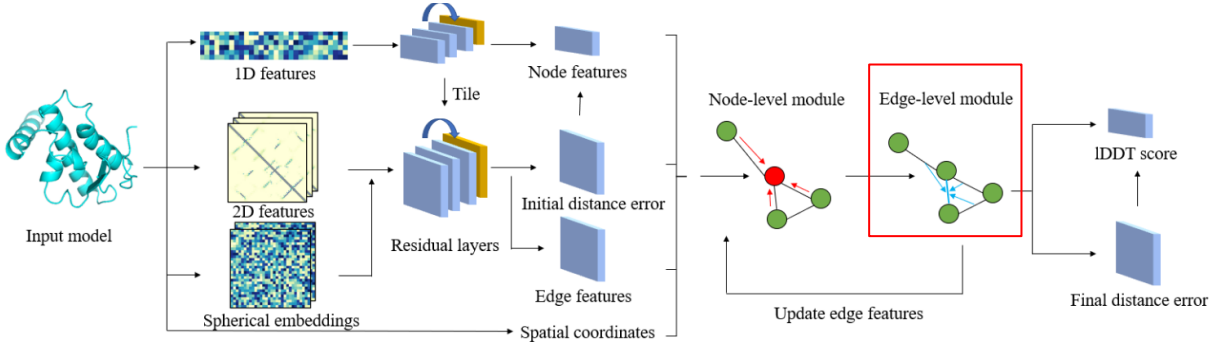
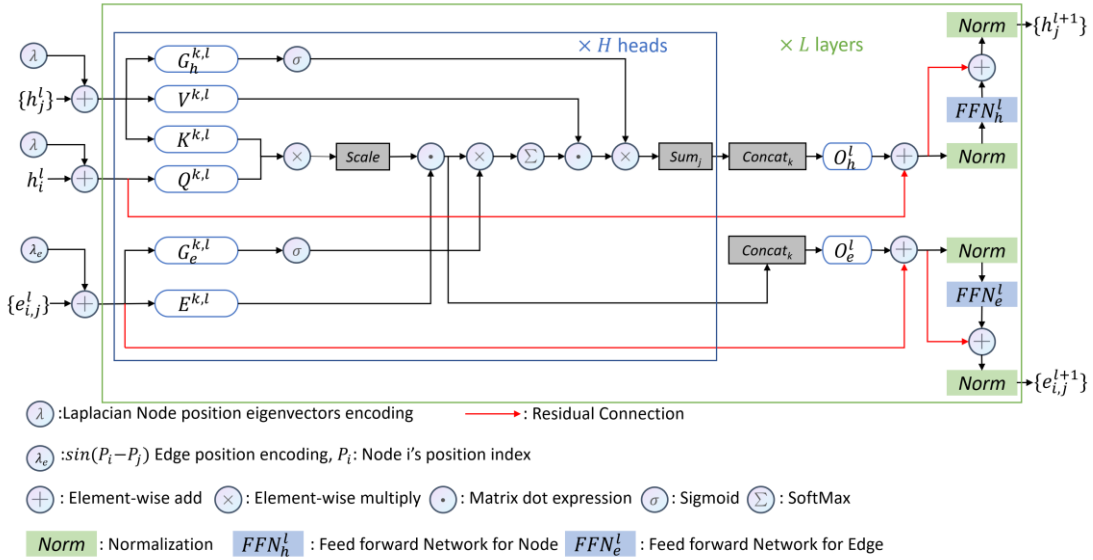


Figure 2. The illustration of the overall architecture of EnQA. The 1D/2D features from the input model are first converted into hidden node and edge features for the 3D-equivarant graph module. The spatial coordinates of Ca atoms of the residues are also used as an extra feature. The node and edge network modules update the graph features iteratively. In the end, the final per-residue lDDT score and distance errors of residue pairs are predicted from the updated node/edge features and spatial coordinates by the 3D-equivariant network.

Extensions (2)

- Likewise, we have also adapted the idea of graph-based feature gating in the Geometric Transformer to other problems in protein bioinformatics (i.e., DPROQ for quaternary protein model quality assessment)



Future Directions

- Ablating the CNN component of our DeepInteract pipeline and instead treating all chains as a single heterogeneous geometric graph

- Replacing the node-local self-attention mechanism in the Geometric Transformer with a node-global multi-head attention module - long-range interactions may play an important role in determining interface contact points, especially if modeling the complex as a single graph

- Investigating the impact of protein-protein interface interaction information on models designed for protein-protein, protein-ligand, or protein-DNA docking, just to name a few examples

Reproducibility

- Source code, data, and pre-trained models are available on GitHub at

https://github.com/BioinfoMachineLearning/DeepInteract

References

- [VSPUJGKP 17] Attention is All you Need. In Advances in Neural Information Processing Systems, volume 30, 2017.
- [DB 21] A Generalization of Transformer Networks to Graphs. In AAAI Workshop on Deep Learning on Graphs: Methods and Applications (DLG-AAAI), 2021.
- [LWLLZOJ 22] Spherical Message Passing for 3D Molecular Graphs. In International Conference on Learning Representations, 2022.
- [FBSB 17] Protein Interface Prediction using Graph Convolutional Networks. In Advances in Neural Information Processing Systems, volume 30, 2017.
- [IGBJ 19] Generative models for graph-based protein design. In Advances in Neural Information Processing Systems, volume 32, 2019.
- [TBSD 19] End-to-End Learning on 3D Protein Structure for Interface Prediction. In Advances in Neural Information Processing Systems, volume 32, 2019.
- [MCSC 21] DIPS-Plus: The Enhanced Database of Interacting Protein Structures for Interface Prediction. In arXiv preprint arXiv:2106.04362, 2021.
- [LBNV 19] Blind prediction of homo-and hetero-protein complexes: The CASP13-CAPRI experiment. In Proteins: Structure, Function, and Bioinformatics, 2019.
- [LBMN 21] Prediction of protein assemblies, the next frontier: The CASP14-CAPRI experiment. In Proteins: Structure, Function, and Bioinformatics, 2021.
- [VMVP 15] Updates to the integrated protein–protein interaction benchmarks: Docking Benchmark version 5 and Affinity Benchmark version 2. In the Journal of Molecular Biology, 2015.

References

- [KW 17] Semi-Supervised Classification with Graph Convolutional Networks. In International Conference on Learning Representations, 2017.
- [MCC 22] Semi-Supervised Graph Learning Meets Dimensionality Reduction. In review. Available on arXiv, 2022.