

A Generalization of ViT/MLP-Mixer to Graphs with Molecular Application

Xavier Bresson

<https://twitter.com/xbresson>



Department of Computer Science
National University of Singapore (NUS)



Joint work with X. He, B. Hooi (NUS), T. Laurent (LMU)
A. Perold (Element), Y. LeCun (NYU/Meta)



IPAM Workshop on
Learning and Emergence in
Molecular Systems

Jan 25th 2023



Outline

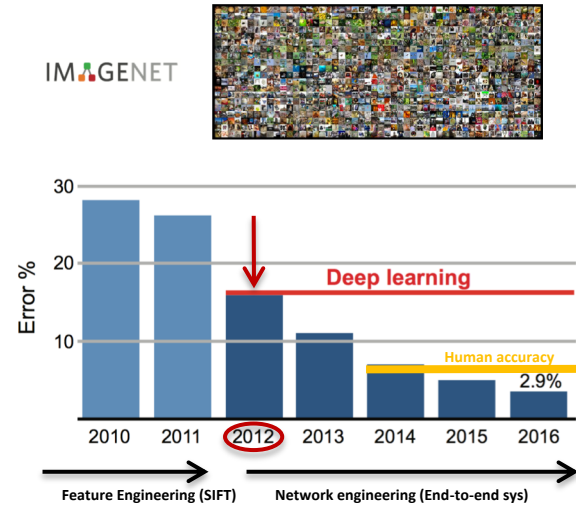
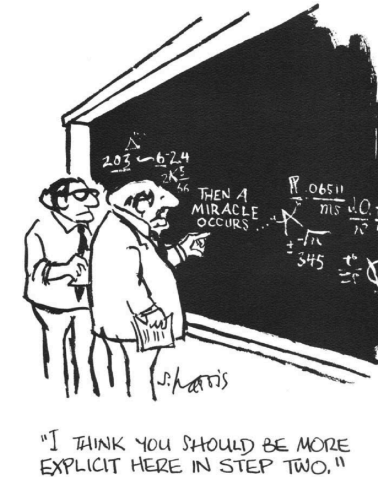
- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- Proposed architecture
- Numerical experiments
- Conclusion

Outline

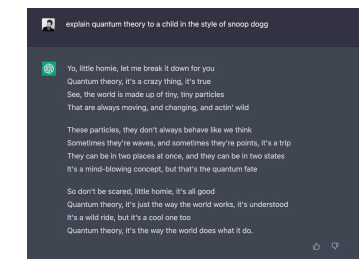
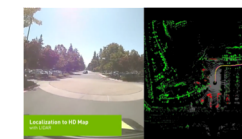
- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- Proposed architecture
- Numerical experiments
- Conclusion

Deep Learning

- The DL revolution – 2012 ImageNet
 - First breakthrough in computer vision (CV)
 - Then speech (SR) and language processing (NLP)
- DL works very well.. in practice but why?
 - High-dimensional data/curse of dimensionality
 - Characterization of energy landscapes/solutions
 - Generalization (feature learning, inductive bias, expressivity)



- DL products
 - Computer Vision : Face recognition, video surveillance, autonomous driving
 - Natural Language Processing : Machine translation, chatbot (ChatGPT)
 - Speech Recognition : Virtual assistants (Alexa/Siri/Google/Cortana)
- DL beyond CV/NLP/SR for scientific discovery.



AMAZON'S ALEXA



GOOGLE'S ASSISTANT



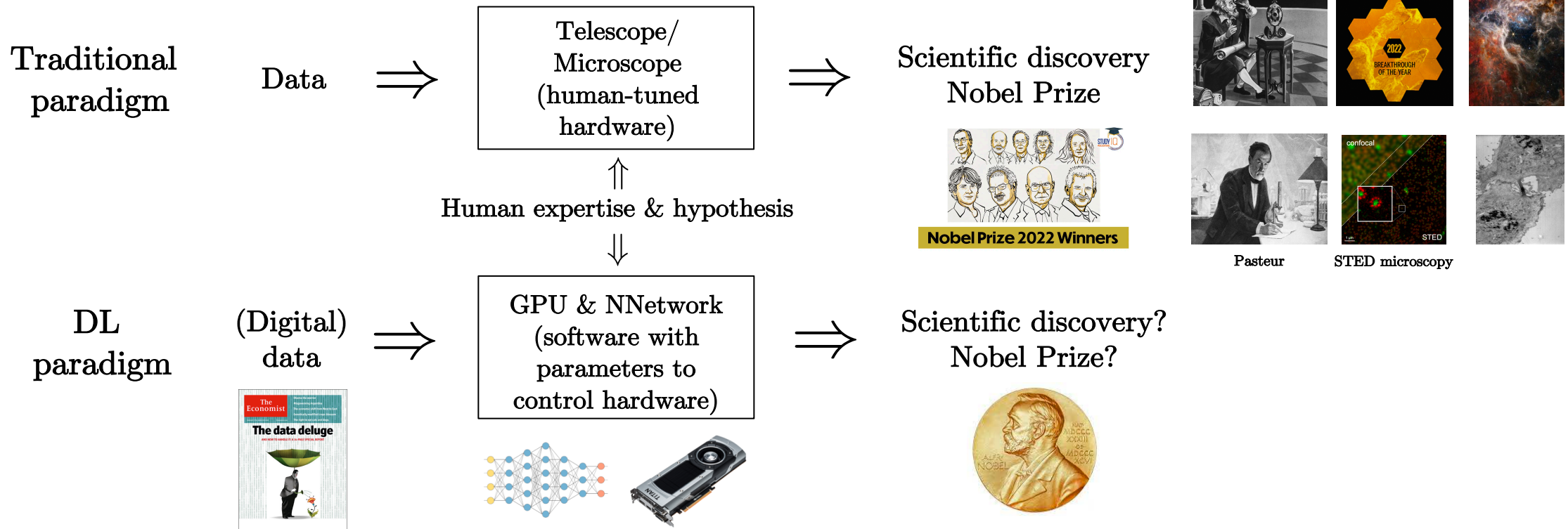
APPLE'S SIRI



MICROSOFT'S CORTANA

DL for science

- DL has been applied to biology, physics, chemistry, material, medicine, engineering, etc
- Scientific tools s.a. telescope/microscope have served physics (Galileo's proof that earth is round and revolves around the sun, study of universe) and biology (Pasteur's discovery of microbes, study of cells, molecules).
- Can DL be the new telescope/microscope for scientific discovery?



DL for molecular science

- A promising application of DL and potentially breakthrough is in molecular science.
- Case studies
 - Drug discovery (halicin antibiotic)
 - De novo drug design (new molecules w/ optimized chemical property)
 - Protein Folding (3D structure)
 - Protein-Drug interaction (drugs with strong binding)
 - Generate protein with text prompt (ChatGPT for biology)

Graph DL for drug discovery



Cell

Volume 180, Issue 4, 20 February 2020, Pages 688-702.e13



Article

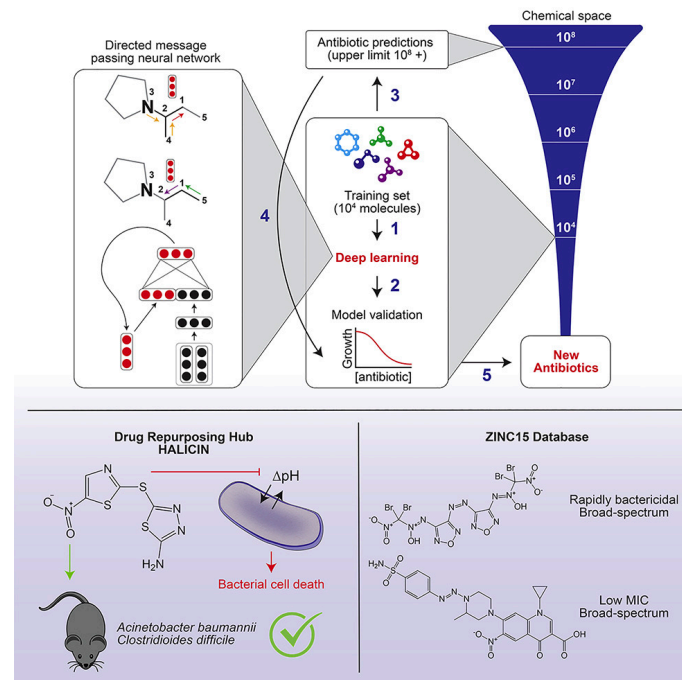
A Deep Learning Approach to Antibiotic Discovery

Jonathan M. Stokes^{1,2,3}, Kevin Yang^{3,4,10}, Kyle Swanson^{3,4,10}, Wengong Jin^{3,4}, Andres Cubillos-Ruiz^{1,2,5}, Nina M. Donghia^{1,5}, Craig R. MacNair⁶, Shawn French⁶, Lindsey A. Carfrae⁶, Zohar Bloom-Ackermann^{2,7}, Victoria M. Tran², Anush Chiappino-Pepe^{5,7}, Ahmed H. Badran², Ian W. Andrews^{1,2,5}, Emma J. Chory^{1,2}, George M. Church^{5,7,8}, Eric D. Brown⁶, Tommi S. Jaakkola^{3,4} ... James J. Collins^{1,2,5,8,9,11}  

Highlights

- A deep learning model is trained to predict antibiotics based on structure
- Halicin is predicted as an antibacterial molecule from the Drug Repurposing Hub
- Halicin shows broad-spectrum antibiotic activities in mice
- More antibiotics with distinct structures are predicted from the ZINC15 database

<https://www.sciencedirect.com/science/article/pii/S0092867420301021>



Halicin was previously developed for anti-diabetic treatment.

MIT
Technology
Review

77 Mass Ave

AI vs. bacteria

nature

Explore content Journal information Publish with us Subscribe

View all journals

nature > news > article

NEWS • 20 FEBRUARY 2020

Powerful antibiotics discovered using AI

Machine learning spots molecules that work even against 'untreatable' strains of bacteria.

FINANCIAL TIMES

Artificial intelligence + Add to myFT

AI discovers antibiotics to treat drug-resistant diseases

Machine learning uncovers potent new drug able to kill 35 powerful bacteria

Quantamagazine

Physics Mathematics Biology Computer Science All Articles

ARTIFICIAL INTELLIGENCE

Machine Learning Takes On Antibiotic Resistance

🔍 📄 📊

To combat resistant bacteria and refill the trickling antibiotic pipeline, scientists are getting help from deep learning networks.



Graph DL for *de novo* drug design

Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules

Rafael Gómez-Bombarelli,^{†,‡,§,¶,||} Jennifer N. Wei,^{‡,§,¶} David Duvenaud,^{‡,§,¶} José Miguel Hernández-Lobato,^{‡,§,¶} Benjamín Sánchez-Lengeling,[‡] Dennis Sheberla,[‡] Jorge Aguilera-Iparraguirre,[‡] Timothy D. Hirzel,[‡] Ryan P. Adams,^{‡,||} and Alán Aspuru-Guzik^{*,†,‡,§,¶,||}

[†]Kyulux North America Inc., 10 Post Office Square, Suite 800, Boston, Massachusetts 02109, United States

[‡]Department of Chemistry and Chemical Biology, Harvard University, Cambridge, Massachusetts 02138, United States

[§]Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, Ontario M5S 3H5, Canada

[¶]Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, U.K.

^{||}Google Brain, Mountain View, California, United States

^{||}Princeton University, Princeton, New Jersey, United States

^{*}Biologically-Inspired Solar Energy Program, Canadian Institute for Advanced Research (CIFAR), Toronto, Ontario M5S 1M1, Canada

[Supporting Information](#)

ABSTRACT: We report a method to convert discrete representations of molecules to and from a multidimensional continuous representation. This model allows us to generate new molecules for efficient exploration and optimization through open-ended spaces of chemical compounds. A deep neural network was trained on hundreds of thousands of existing chemical structures to construct three coupled functions: an encoder, a decoder, and a predictor. The encoder converts the discrete representation of a molecule into a real-valued continuous vector, and the decoder converts these continuous vectors back to discrete molecular representations. The predictor estimates chemical properties from the latent continuous vector representation of the molecule. Continuous representations of molecules allow us to automatically generate novel chemical structures by performing simple operations in the latent space, such as decoding random vectors, perturbing known chemical structures, or interpolating between molecules. Continuous representations also allow the use of powerful gradient-based optimization to efficiently guide the search for optimized functional compounds. We demonstrate our method in the domain of drug-like molecules and also in a set of molecules with fewer than nine heavy atoms.

JCIM
JOURNAL OF
CHEMICAL INFORMATION
AND MODELING

RetroGNN: Fast Estimation of Synthesizability for Virtual Screening and De Novo Design by Learning from Slow Retrosynthesis Software

Cheng-Hao Liu*, Maksym Korablyov, Stanisław Jastrzębski, Paweł Włodarczyk-Pruszyński, Yoshua Bengio, and Marwin Segler*

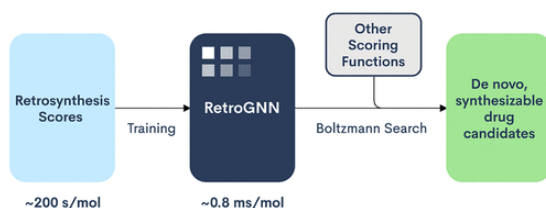
Cite this: *J. Chem. Inf. Model.* 2022, 62, 10, 2293–2300
Publication Date: April 22, 2022
<https://doi.org/10.1021/acs.jcim.1c01476>
Copyright © 2022 American Chemical Society

Article Views
1798

Altmetric
8

Citations
1

[LEARN ABOUT THESE METRICS](#)



scientific reports

[Explore content](#) [About the journal](#) [Publish with us](#)

[nature](#) > [scientific reports](#) > [articles](#) > [article](#)

Article | [Open Access](#) | [Published: 24 July 2019](#)

Optimization of Molecules via Deep Reinforcement Learning

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N. Zare & Patrick Riley

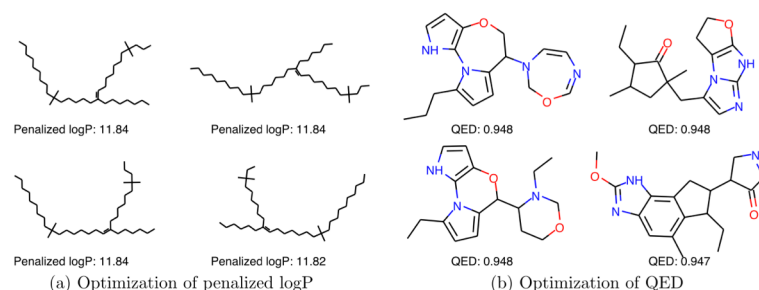


Figure 2. Sample molecules in the property optimization task. (a) Optimization of penalized logP from MolDQN-bootstrap; note that the generated molecules are obviously not drug-like due to the use of a single-objective reward. (b) Optimization of QED from MolDQN-twosteps.

logP: molecular solubility

QED: Quantitative Estimate of Druglikeness

Sundin et al. *Journal of Cheminformatics* (2022) 14:86
<https://doi.org/10.1186/s13321-022-00667-8>

RESEARCH

Open Access

Human-in-the-loop assisted de novo molecular design

Iris Sundin^{1*}, Alexey Voronov^{2*}, Haoping Xiao¹, Kostas Papadopoulos^{2,5}, Esben Jannik Bjerrum^{2,5}, Markus Heinonen¹, Atanas Patronov^{2,5}, Samuel Kaski^{1,3} and Ola Engkvist^{2,4}



Jiacheng Xiong^{1,2}, Zhaoping Xiong^{1,3}, Kaixian Chen^{1,2}, Hualiang Jiang^{1,2} and Mingyue Zheng^{1,2}

¹Drug Discovery and Design Center, State Key Laboratory of Drug Research, Shanghai Institute of Materia Medica, Chinese Academy of Sciences, 555 Zuchongzhi Road, Shanghai 201203, China

²University of Chinese Academy of Sciences, No. 19A Yuquan Road, Beijing 100049, China

³Shanghai Institute for Advanced Immunochemical Studies, and School of Life Science and Technology, ShanghaiTech University, Shanghai 200031, China

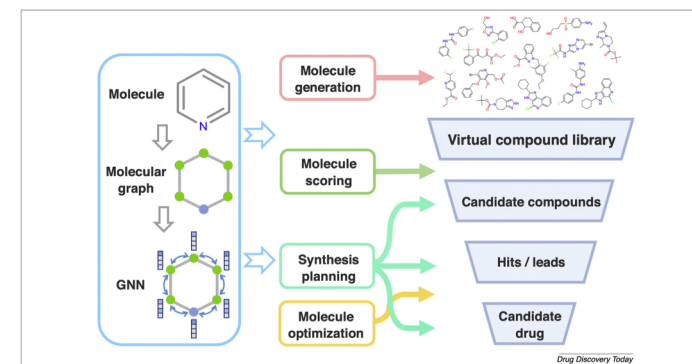
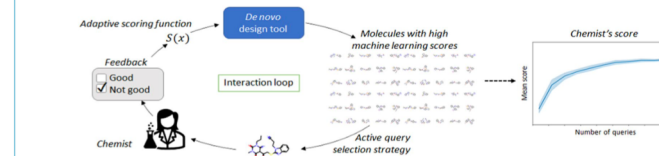


FIGURE 1
The applications of graph neural networks (GNNs) in all stages of automated de novo drug design.

Journal of Cheminformatics

Graphical Abstract



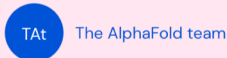
Graph DL for protein folding/structure

AlphaFold: a solution to a 50-year-old grand challenge in biology

SHARE



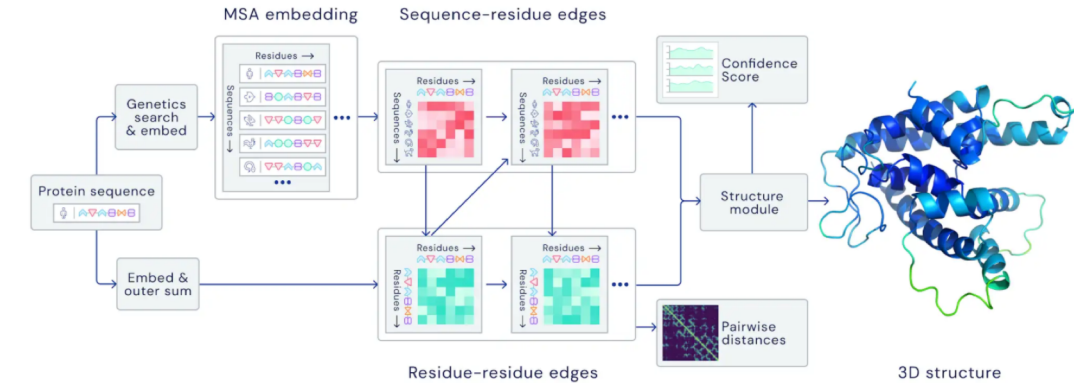
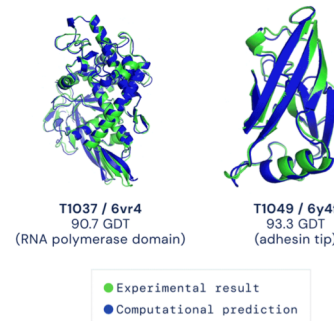
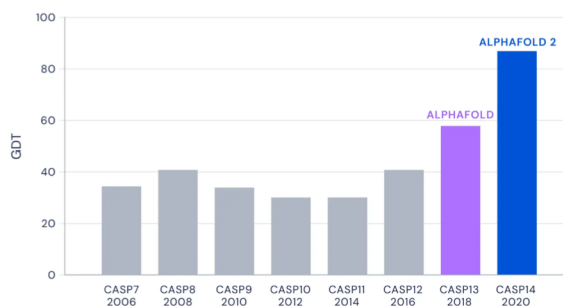
AUTHORS



Proteins are essential to life, supporting practically all its functions. They are large complex molecules, made up of chains of amino acids, and what a protein does largely depends on its unique 3D structure. Figuring out what shapes proteins fold into is known as the “protein folding problem”, and has stood as a grand challenge in biology for the past 50 years. In a major scientific advance, the latest version of our AI system AlphaFold has been recognised as a solution to this grand challenge by the organisers of the biennial Critical Assessment of protein Structure Prediction (CASP). This breakthrough demonstrates the impact AI can have on scientific discovery and its potential to dramatically accelerate progress in some of the most fundamental fields that explain and shape our world.

<https://deepmind.com/blog/article/alphafold-a-solution-to-a-50-year-old-grand-challenge-in-biology>

Median Free-Modelling Accuracy



Graph Transformer

Science

RESEARCH ARTICLES

Cite as: M. Baek et al., *Science* 10.1126/science.abj8754 (2021).

Accurate prediction of protein structures and interactions using a three-track neural network

Minkyung Baek^{1,2}, Frank DiMaio^{1,2}, Ivan Anishchenko^{1,2}, Justas Dauparas^{1,2}, Sergey Ovchinnikov^{3,4}, Gyu Rie Lee^{1,2}, Jue Wang^{1,2}, Qian Cong^{5,6}, Lisa N. Kinch⁷, R. Dustin Schaeffer⁸, Claudia Millán⁹, Hahnbeom Park^{1,2}, Carson Adams^{1,2}, Caleb R. Glassman^{5,10}, Andy DeGiovanni¹², Jose H. Pereira¹², Andria V. Rodrigues¹², Alberdina A. van Dijk¹³, Ana C. Ebrecht¹³, Diederik J. Opperman¹⁴, Theo Sagmeister¹⁵, Christoph Buhllheller^{15,16}, Tea Pavkov-Keller^{15,17}, Manoj K. Rathinaswamy¹⁸, Udit Dalwadi¹⁹, Calvin K. Yip¹⁹, John E. Burke¹⁸, K. Christopher Garcia^{9,10,11,20}, Nick V. Grishin^{6,21,7}, Paul D. Adams^{12,22}, Randy J. Read⁸, David Baker^{1,2,23*}

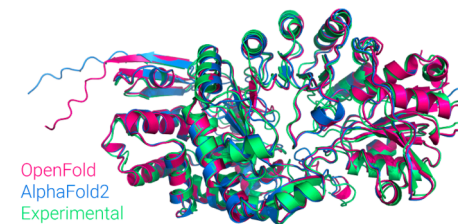
RoseTTaFold

bioRxiv

THE PREPRINT SERVER FOR BIOLOGY

OpenFold: Retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization

Gustaf Ahlritz, Nazim Bouatta, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, Arkadiusz Nowaczynski, Bei Wang, Marta M Stepniowska-Dziubinska, Shang Zhang, Adegoke Ojewole, Murat Efe Guney, Stella Biderman, Andrew M Watkins, Stephen Ra, Pablo Ribalta Lorenzo, Lucas Nivon, Brian Weitzner, Yih-En Andrew Ban, Peter K Sorger, Emad Mostaque, Zhao Zhang, Richard Bonneau, Mohammed AlQuraishi



OpenFold

Graph DL for protein-drug discovery & design

Structure-based drug discovery with deep learning

Rıza Özçelik^{1,2+}, Derek van Tilborg^{1,2+}, José Jiménez-Luna³, and Francesca Grisoni^{1,2*}

¹Eindhoven University of Technology, Institute for Complex Molecular Systems and Dept. Biomedical Engineering, Eindhoven, Netherlands.

²Centre for Living Technologies, Alliance TU/e, WUR, UU, UMC Utrecht, Netherlands.

³Microsoft Research Cambridge, Cambridge, United Kingdom.

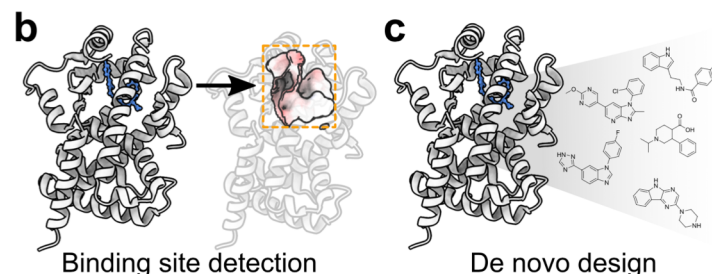
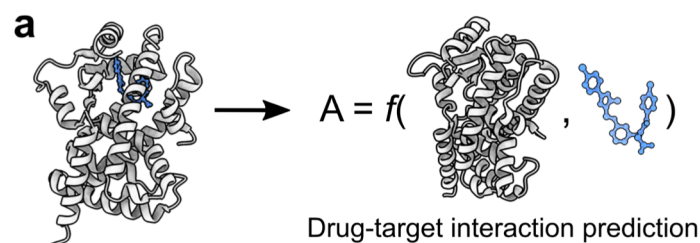


Figure 1. Structure-based drug discovery tasks discussed in this review: (a) *drug-target interaction prediction*, which aims to predict the affinity between a protein and a ligand, using the structural information of both molecular entities; (b) *binding site detection*, which aims to identify ‘druggable’ cavities in the protein structure, (c) *de novo design*, aiming to design bioactive molecules from scratch using the information of a protein target.

arXiv > q-bio > arXiv:2202.05146

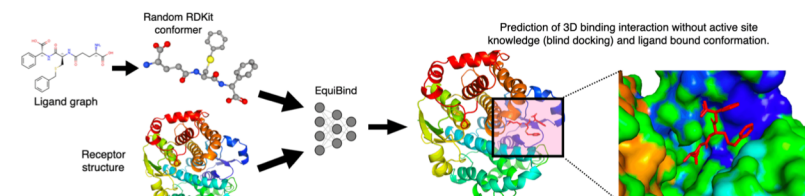
Quantitative Biology > Biomolecules

[Submitted on 7 Feb 2022 (v1), last revised 4 Jun 2022 (this version, v4)]

EquiBind: Geometric Deep Learning for Drug Binding Structure Prediction

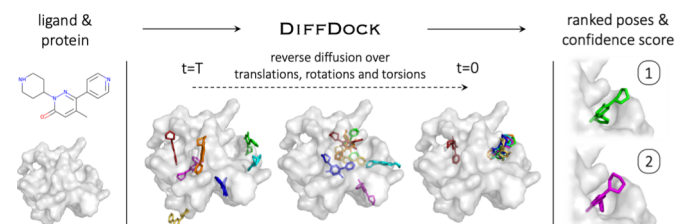
Hannes Stärk, Octavian-Eugen Ganeva, Lagnajit Pattanaik, Regina Barzilay, Tommi Jaakkola

Predicting how a drug-like molecule binds to a specific protein target is a core problem in drug discovery. An extremely fast computational binding method would enable key applications such as fast virtual screening or drug engineering. Existing methods are computationally expensive as they rely on heavy candidate sampling coupled with scoring, ranking, and fine-tuning steps. We challenge this paradigm with EquiBind, an SE(3)-equivariant geometric deep learning model performing direct-shot prediction of both i) the receptor binding location (blind docking) and ii) the ligand's bound pose and orientation. EquiBind achieves significant speed-ups and better quality compared to traditional and recent baselines. Further, we show extra improvements when coupling it with existing fine-tuning techniques at the cost of increased running time. Finally, we propose a novel and fast fine-tuning model that adjusts torsion angles of a ligand's rotatable bonds based on closed-form global minima of the von Mises angular distance to a given input atomic point cloud, avoiding previous expensive differential evolution strategies for energy minimization.



DIFFDOCK: DIFFUSION STEPS, TWISTS, AND TURNS FOR MOLECULAR DOCKING

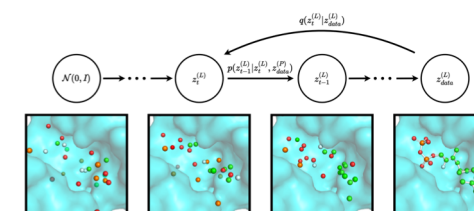
Gabriele Corso*, Hannes Stärk*, Bowen Jing*, Regina Barzilay & Tommi Jaakkola
CSAIL, Massachusetts Institute of Technology



STRUCTURE-BASED DRUG DESIGN WITH EQUIVARIANT DIFFUSION MODELS

Arne Schneuing^{1*}, Yuanqi Du^{2*}, Charles Harris³, Arian Jamash³, Ilia Igashov¹, Weitao Du¹, Tom Blundell³, Pietro Lió³, Carla Gomes², Max Welling⁵, Michael Bronstein⁶ & Bruno Correia¹

¹École Polytechnique Fédérale de Lausanne, ²Cornell University, ³University of Cambridge, ⁴USTC, ⁵Microsoft Research AI4Science, ⁶University of Oxford



Generate protein with text prompt

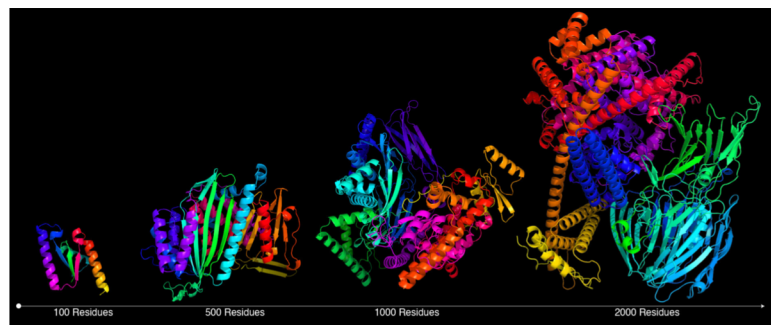
Inspired by multi-modal image-language models s.a. OpenAI's DALL-E, Google Brain's Imagen and StabilityAI's Stable Diffusion :

Illuminating protein space
with a programmable generative model

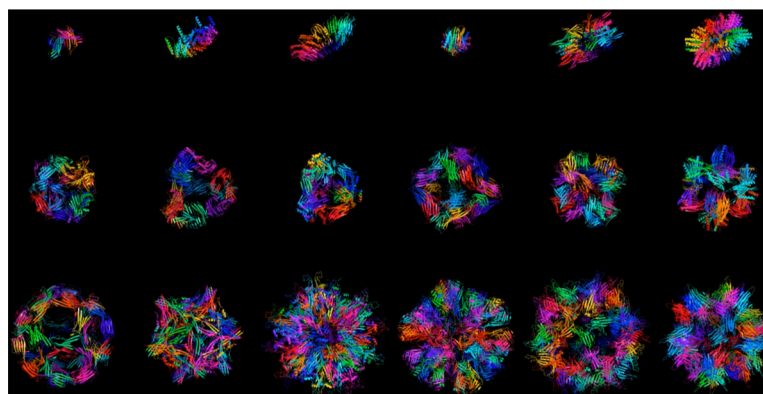
John Ingraham, Max Baranov, Zak Costello, Vincent Frappier,
Ahmed Ismail, Shan Tie, Wujie Wang, Vincent Xue, Fritz Obermeyer,
Andrew Beam, Gevorg Grigoryan

<https://www.generatebiomedicines.com/chroma>

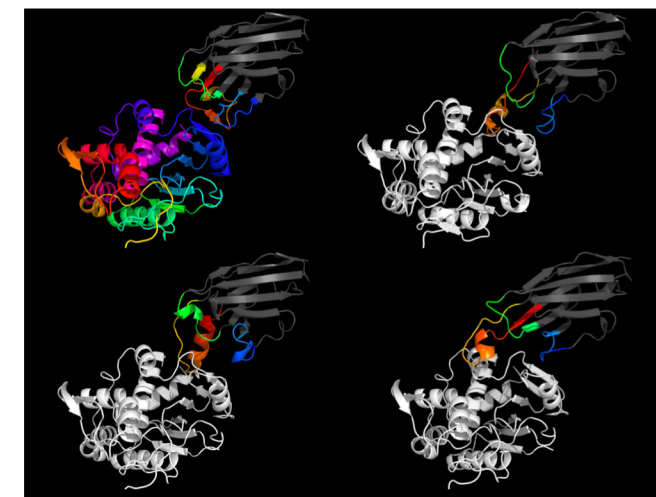
Programming proteins with Chroma



Making giants



Symmetry groups



Protein infilling

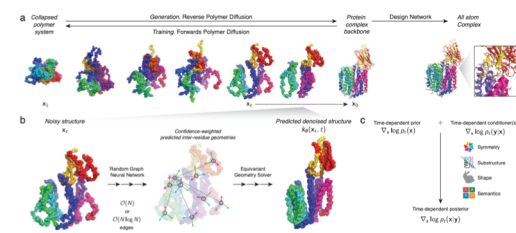
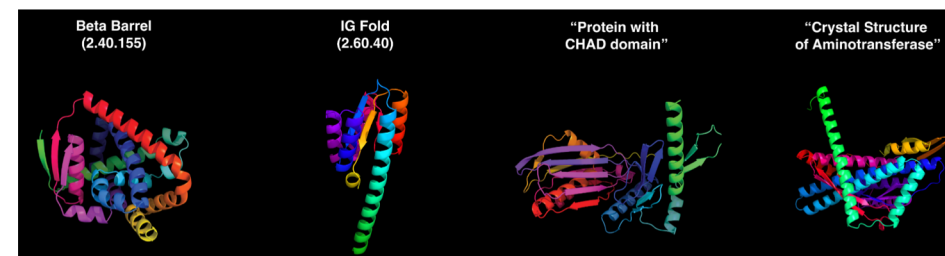


Figure 1: *Chroma* is a generative model for proteins and protein complexes that combines a structured diffusion model for protein backbones with scalable molecular neural networks for backbone synthesis and all-atom design. a, A correlated diffusion process with chain and ra-

Semantic conditioning



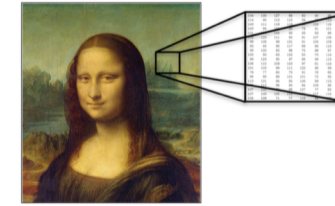
Chroma: "Generate a protein with CHAD domain"
(with small GPT trained on protein captioning)

Outline

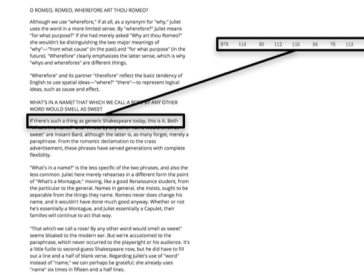
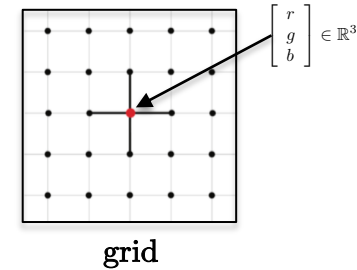
- DL for molecular science
- **Review of graph network architectures**
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- Proposed architecture
- Numerical experiments
- Conclusion

Neural network architectures

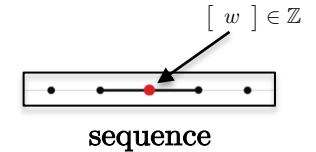
- To enable DL+X, we need three key ingredients :
 - Data, GPU and neural network
- There are four classes of networks
 - CNNs/RNNs/Transformers are designed for grids, sequences and sets.
 - GPUs/DL libraries PyTorch^[1], TensorFlow^[2] are optimized for these architectures.
 - Graph Neural Networks (GNNs) are more universal and broadly applicable architectures.
 - GPUs are not optimized for sparse linear algebra.



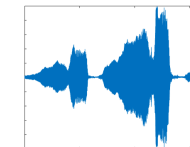
Computer Vision (CV)



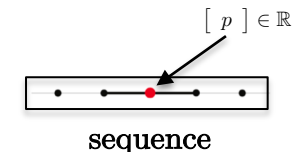
Natural Language Processing (NLP)



Graph Analysis



Speech Recognition (SR)



[1] Paszke et-al, Automatic differentiation in pytorch, 2017

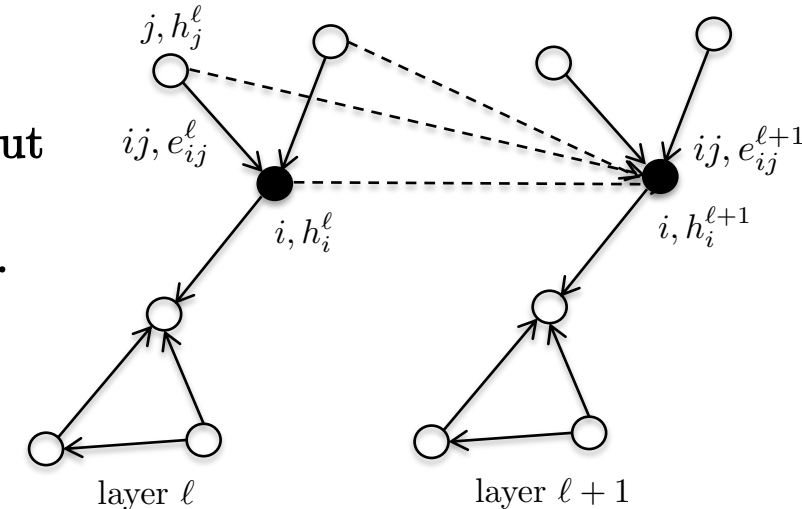
[2] Abadi et-al, TensorFlow: a system for Large-Scale machine learning, 2016

GNN architectures

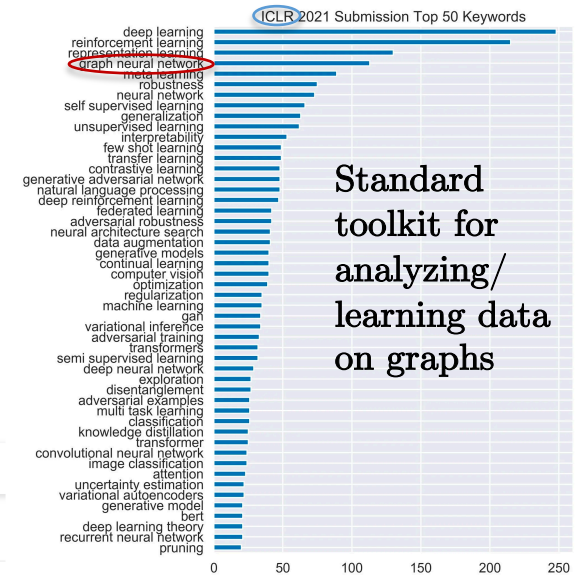
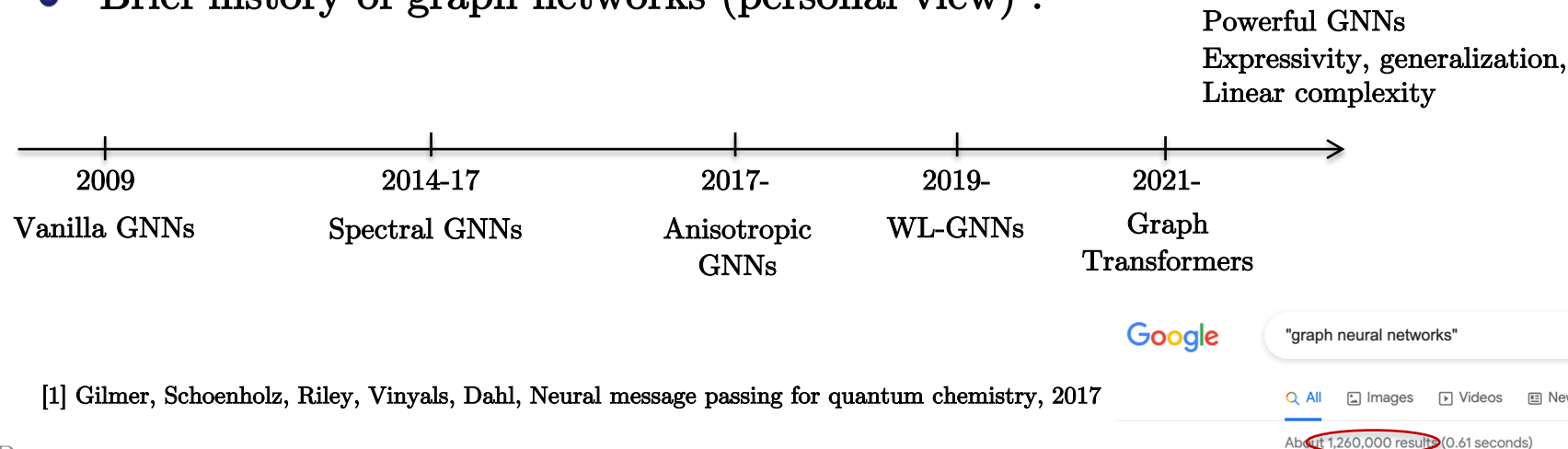
- What is a graph network?
 - Compositional parametric equivariant function that maps an input pair (graph, feature) to continuous vectorial representation for node/edge/graph which can be used for downstream graph tasks.
 - Often defined as a message-passing process^[1] :

$$\text{Node-update : } h_i^{\ell+1} = f_{\text{node}}(h_i^{\ell}, \{h_j^{\ell}, e_{ij}^{\ell} : j \in \mathcal{N}_i\}) \in \mathbb{R}^d$$

$$\text{Edge-update : } e_{ij}^{\ell+1} = f_{\text{edge}}(e_{ij}^{\ell}, h_i^{\ell}, h_j^{\ell}) \in \mathbb{R}^d$$



- Brief history of graph networks (personal view) :



[1] Gilmer, Schoenholz, Riley, Vinyals, Dahl, Neural message passing for quantum chemistry, 2017

Vanilla GNNs^[1]

- Contributions
 - First NN with layers equivariant/invariant w.r.t. index permutation, independent to neighborhood and graph size, local reception field, weight sharing.
 - Linear complexity $O(N+E)$
- Limitations
 - Simple model (vanilla RNN w/ aggregation of neighbors)
 - Vanishing gradient problem

$$\begin{aligned}h_i^{\ell+1} &= f_{\text{VGNN}}(h_i^\ell, \{h_j^\ell : j \in \mathcal{N}_i\}) \\ &= \sum_{j \rightarrow i} \sigma(Ux_i + Vh_j^\ell) \in \mathbb{R}^d\end{aligned}$$

IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 20, NO. 1, JANUARY 2009

61

The Graph Neural Network Model

Franco Scarselli, Marco Gori, *Fellow, IEEE*, Ah Chung Tsoi, Markus Hagenbuchner, *Member, IEEE*, and Gabriele Monfardini

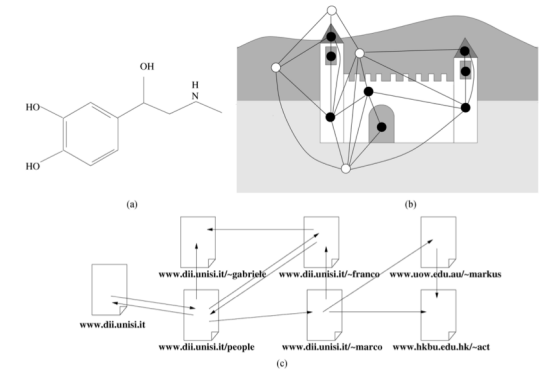


Fig. 1. Some applications where the information is represented by graphs: (a) a chemical compound (adrenaline), (b) an image, and (c) a subset of the web.

[1] Scarselli, Gori, Tsoi, Hagenbuchner, Monfardini, The Graph Neural Network Model, 2009

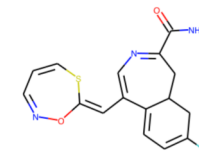
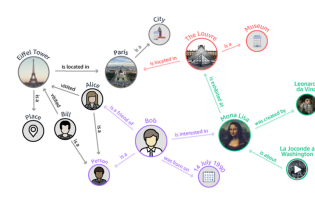
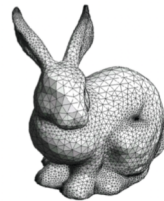
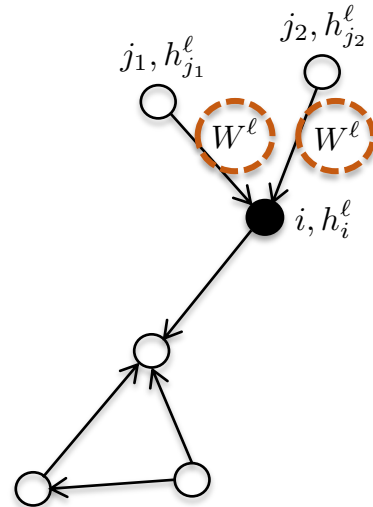
Spectral GNNs^[1,2,3]

- Contributions

- Leverage graph spectral theory to define convolution on graphs
- Stable, robust to perturbation, exact k-hop kernel support
- Linear complexity $O(N+E)$
- GCN^[3] is the most popular GNN technique

- Limitation

- Isotropic kernels



Most data is anisotropic

$$\begin{aligned}
 h^{\ell+1} &= \sigma(s^\ell *_G h^\ell) \in \mathbb{R}^{N \times d} \\
 &= \sigma(U \hat{s}^\ell(\Lambda) U^T h^\ell) \\
 &= \sigma(\hat{s}^\ell(\Delta) h^\ell), \quad \Delta = U \Lambda U^T \\
 &= \sigma(\sum_{k=0}^{K-1} T_k(\Delta) h^\ell W_k^\ell) \\
 &\quad \text{Chebyshev functions} \\
 &= \sigma(D^{-1/2} A D^{-1/2} h^\ell W^\ell) \\
 &\quad \text{1st order approx} \\
 h_i^{\ell+1} &= \sigma(\frac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} \frac{1}{\sqrt{d_j}} A_{ij} W^\ell h_j^\ell) \in \mathbb{R}^d
 \end{aligned}$$

[1] Bruna, Zaremba, Szlam, LeCun, Spectral networks and locally connected networks on graphs, 2013

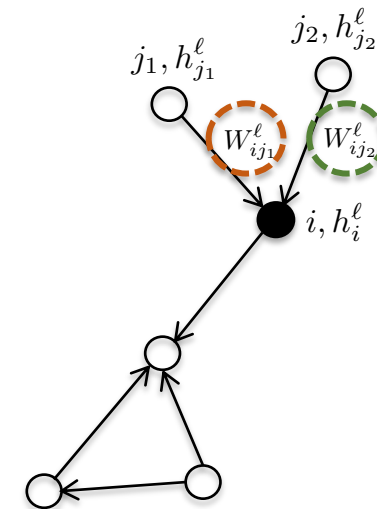
[2] Defferrard, Bresson, Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, 2016

[3] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017

Anisotropic GNNs

- Contributions

- Design anisotropic equivariant mechanisms that treat neighbors differently.
- GatedGCNs^[1] with anisotropic diffusion^[2], GAT^[3] with softmax attention^[4], Mesh CNNs^[5] with anisotropic gauge equivariant kernels
- Stable, robust, interpretable, linear complexity $O(N+E)$



$$h_i^{\ell+1} = h_i^{\ell} + \text{ReLU}\left(\text{BN}\left(W_1^{\ell} h_i^{\ell} + \sum_{j \in \mathcal{N}_i} e_{ij}^{\ell} \odot W_2^{\ell} h_j^{\ell}\right)\right)$$

$$e_{ij}^{\ell} = \frac{\sigma(\hat{e}_{ij}^{\ell})}{\sum_{j' \in \mathcal{N}_i} \sigma(\hat{e}_{ij'}^{\ell}) + \varepsilon}$$

$$\hat{e}_{ij}^{\ell} = \hat{e}_{ij}^{\ell-1} + \text{ReLU}\left(\text{BN}\left(V_1^{\ell} h_i^{\ell-1} + V_2^{\ell} h_j^{\ell-1} + V_3^{\ell} \hat{e}_{ij}^{\ell-1}\right)\right)$$

GatedGCNs^[1]

(based on Perona-Malik's anisotropic PDE^[2] generalized on graphs)



- Limitations

- Low expressivity, over-squashing issue

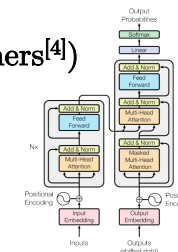
$$h_i^{\ell+1} = \text{Concat}_{k=1}^K \left(\text{ELU} \left(\sum_{j \in \mathcal{N}_i} e_{ij}^{k,\ell} W_1^{k,\ell} h_j^{\ell} \right) \right)$$

$$e_{ij}^{k,\ell} = \text{Softmax}_{\mathcal{N}_i}(\hat{e}_{ij}^{k,\ell}) = \frac{\exp(\hat{e}_{ij}^{k,\ell})}{\sum_{j' \in \mathcal{N}_i} \exp(\hat{e}_{ij'}^{k,\ell})}$$

$$\hat{e}_{ij}^{k,\ell} = \text{LeakyReLU} \left(W_2^{k,\ell} \text{Concat} \left(W_1^{k,\ell} h_i^{\ell}, W_1^{k,\ell} h_j^{\ell} \right) \right)$$

GAT^[3]

(based on Transformers^[4])



[1] Bresson, Laurent, Residual gated graph convnets, 2017

[2] Perona, Malik, Scale-space and edge detection using anisotropic diffusion, 1987

[3] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph Attention Networks, 2017

[4] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin, Attention is all you need, 2017

[5] De Haan, Weiler, Cohen, Welling, Gauge equivariant mesh CNNs: Anisotropic convolutions on geometric graphs, 2020

Low expressivity

- Most GNNs s.a. GCN^[1], GAT^[2], GatedGCNs^[3] have theoretically low expressivity/representation power to
 - Distinguish (simple) non-isomorphic graphs^[4,5].
 - Identify/count elementary sub-structures like cycles and cliques^[6].

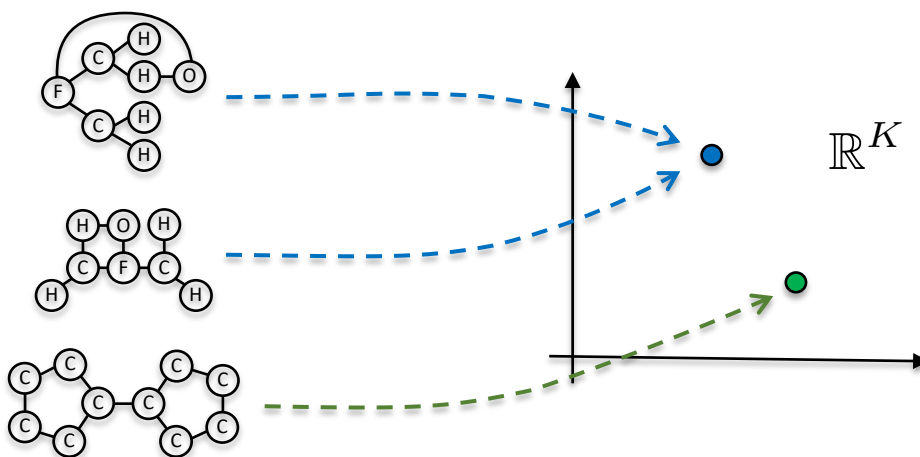


Fig 1. Vectorial graph representation

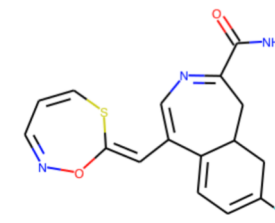


Fig 2. Graph with cycles (rings)

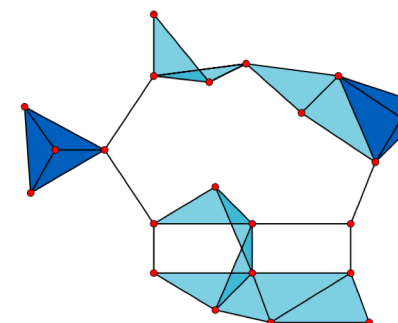


Fig 3. Graph with
19 × 3-vertex cliques (light and dark blue triangles)
and 2 × 4-vertex cliques (dark blue areas)

- [1] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
- [2] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph Attention Networks, 2018
- [3] Bresson, Laurent, Residual gated graph convnets, 2017
- [4] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks? 2019
- [5] Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe, Weisfeiler and leman go neural: Higher-order graph networks, 2019
- [6] Chen, Chen, Villar, Bruna, Can graph neural networks count substructures? 2020

Graph isomorphism

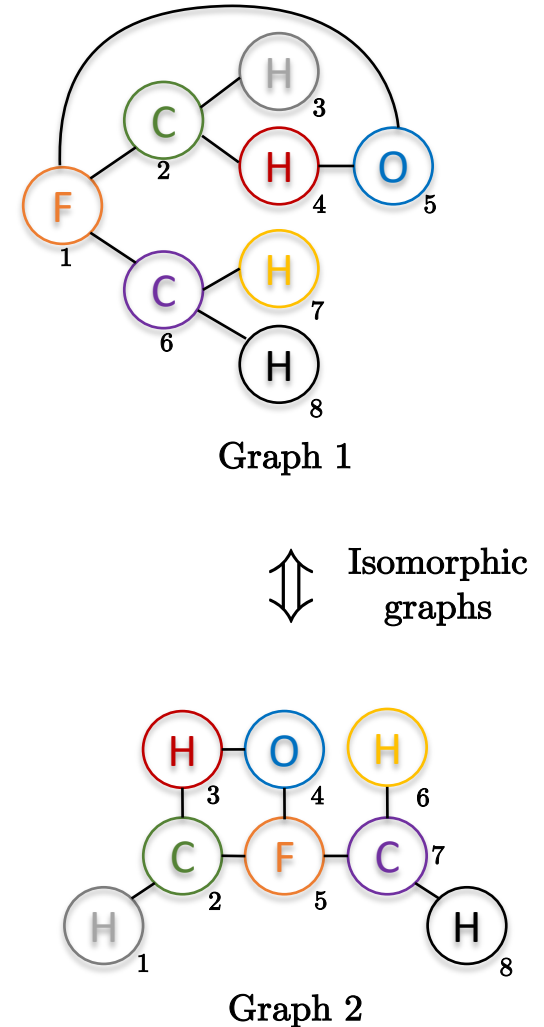
- Graph isomorphism : Two graphs are isomorphic if there exists an index permutation between the nodes that preserves node neighbors.
- Determining whether two graphs are isomorphic is NP-intermediate. It is not known if a polynomial time algorithm exists, or the problem is NP-hard.
- Weisfeiler-Lehman test^[1] provides a necessary (but not sufficient) condition to guarantee that two graphs are isomorphic.
 - Design an injective coloring function f_{WL} that takes a pair (node, its neighborhood) as input, and outputs a new node color :

$$f_{\text{WL}}(c_i^\ell, \{c_j^\ell\}_{j \in \mathcal{N}_i}) = c_i^{\ell+1}$$

iteration

↙ ↘

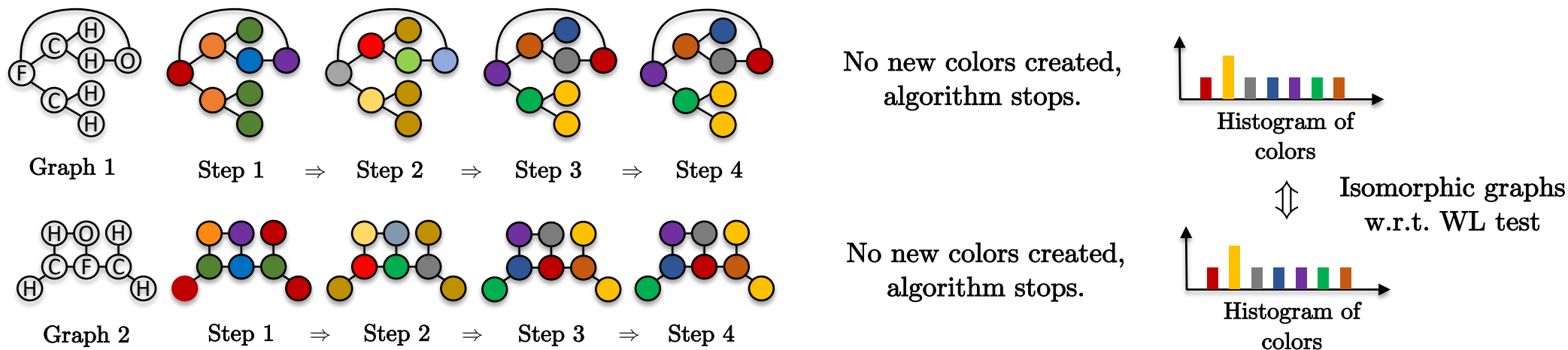
Multiset (set of unordered
and repetitive elements)



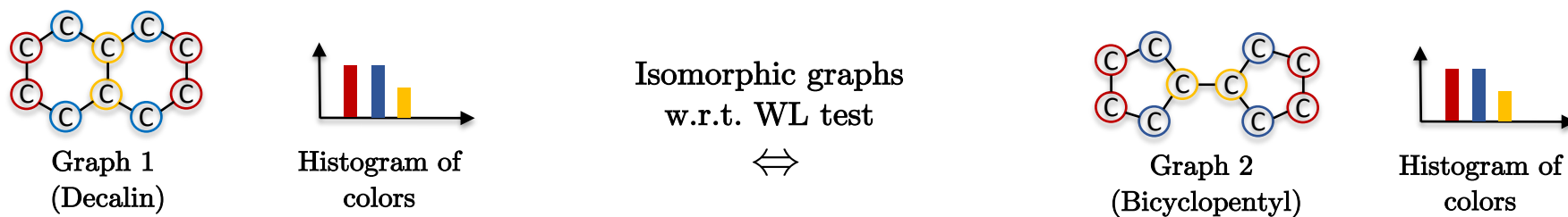
[1] Weisfeiler, Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968

Weisfeiler-Lehman test^[1]

- WL algorithm iteratively applies the coloring function f_{WL} until no new colors are created :



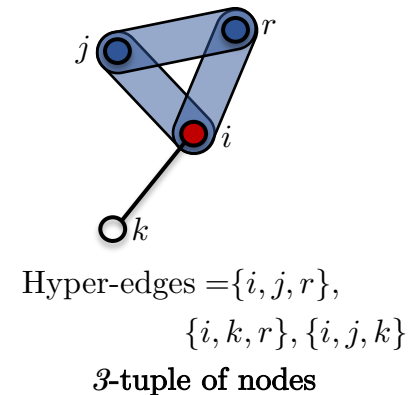
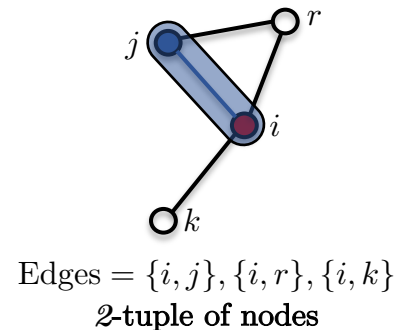
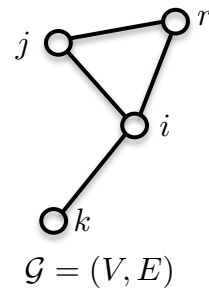
- However the 1-WL test can fail to distinguish (simple) non-isomorphic graphs :



[1] Weisfeiler, Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968

WL-GNNs

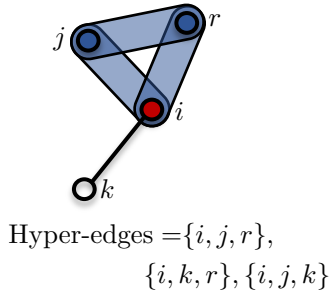
- GIN^[1] designed to be as maximally expressive as the original 1-WL test^[2].
- k-WL tests : Original test uses 2-tuple of nodes. To improve expressivity power of WL test, higher-order interactions between nodes with k-tuple of nodes with $k \geq 3$ can be used.
 - k-order equivariant GNNs^[3] are theoretically more expressive but
 - These networks require $O(N^k)$ memory/speed complexities, with at least $k=3$ to be more powerful than GIN, which means $O(N^3)$ and thus not practical.
 - 3-WL/Ring/2-FGNN GNNs^[4,5,6] have $O(N^2)$ -memory but $O(N^3)$ -speed complexities.
 - Expressivity does not necessarily imply generalization^[7].



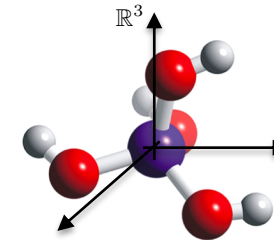
- [1] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019
- [2] Weisfeiler, Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968
- [3] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019
- [4] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019
- [5] Chen, Villar, Chen, Bruna, On the equivalence between graph isomorphism testing and function approximation with gnns, 2019
- [6] Azizian, Lelarge, Expressive power of invariant and equivariant graph neural networks, 2020
- [7] Dwivedi, Joshi, Laurent, Bengio, Bresson, Benchmarking graph neural networks, 2020

Higher expressivity power

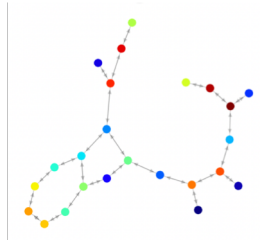
- How to improve the limited expressivity of MP-GNNs ?
 - Consider higher-order node interactions (k-tuples with $k > 2$) with WL-GNNs^[1,2,3,4].
 - Theorem^[5] (universal approximator) : Any continuous function invariant by permutation can be arbitrarily approximated by WL-GNNs, with the necessary condition the network has higher-order tensor of order $k = \text{poly}(N) = N(N-1)/2$.
 - These networks are computationally expensive, at least $O(N^3)$ for 3-WL expressivity.
 - Provide a unique ID for each node, i.e. positional encoding (PE).
 - Theorem^[6] : MP-GNNs are provable more expressive than the 1-WL test when considering node positional encoding.
 - Theorem^[7] : MP-GNNs are Turing-complete when depth $d \geq \delta_G$ layers (graph diameter), width is unbounded, and each node is uniquely identified.



- [1] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019
- [2] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019
- [3] Chen, Chen, Villar, Bruna, Can graph neural networks count substructures? 2020
- [4] Azizian, Lelarge, Expressive power of invariant and equivariant graph neural networks, 2020
- [5] Maron Fetaya, Segol, Lipman, On the universality of invariant networks, 2019
- [6] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019
- [7] Loukas, What graph neural networks cannot learn: depth vs width, 2019



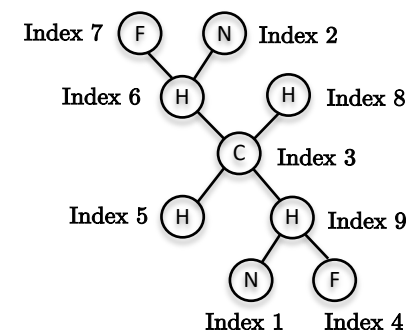
Geometric PE
(e.g. 3D coordinates)



Structural PE
(V,E)

Which structural positional encoding?

- Theory does not provide guidance on the choice of structural PE for the class of graphs and task.
- The simplest PE is an (arbitrary) indexing of the nodes^[1], among $N!$ possible indexings.
 - During training, indexing are uniformly sampled from the $N!$ possible choices in order for the network to learn to be independent to these arbitrary choices.
- Laplacian eigenvectors^[2,3]
 - Spectral techniques that embed graphs into an Euclidean space, while preserving the global graph structure.
 - Define via factorization (EVD) of the graph Laplacian matrix.
 - Computational complexity is $O(E^{3/2})$ and $O(N)$ with approximate Nystrom method^[4].
 - LapPE have arbitrary sign. During training, sign of eigenvectors must be uniformly sampled at random between the 2^k possibilities^[3] for the network to learn this invariance.



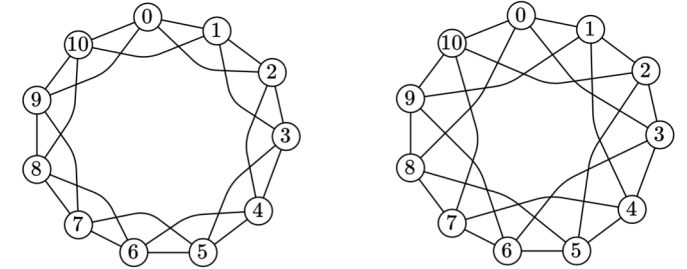
$$\Delta = I - D^{-1/2} A D^{-1/2} = U^T \Lambda U$$

↑ Graph Laplacian ↑ Degree matrix ↑ Adjacency matrix ↑ Eigenvalue matrix ↑ (Laplacian) Eigenvector matrix

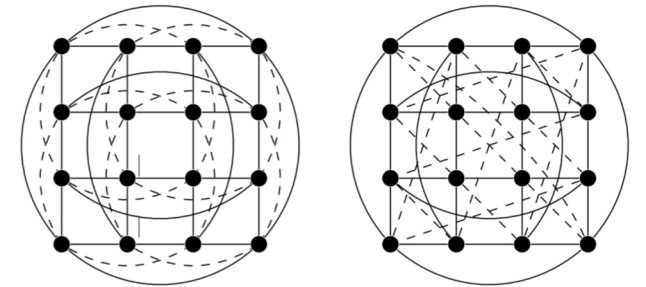
- [1] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019
- [2] Belkin, Niyogi, Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering, 2001
- [3] Dwivedi, Joshi, Laurent, Bengio, Bresson, Benchmarking graph neural networks, 2020
- [4] Fowlkes, Belongie, Chung, Malik, Spectral grouping using the nystrom method, 2004

Synthetic datasets for expressivity

- Circular Skip Link (CSL) dataset^[1]
 - Classify isomorphic graphs that differ by the skip link value.
 - CSL contains 150 4-regular graphs (1-WL failed) divided into 10 isomorphism classes.
- EXP dataset^[2]
 - EXP contains 1200 1&2-WL failed graphs that are split into two classes.
- SR25 dataset^[3]
 - SR25 has 15 strongly regular 3-WL failed graphs with 25 nodes each with the goal of classifying them.



Example of non-isomorphic graphs where 1-WL test failed to distinguish (w/ skip-link 2 and 3).



Example of non-isomorphic graphs where 3-WL test failed to distinguish (strongly regular graphs known as 4×4-Rook and Shrikhande graphs)

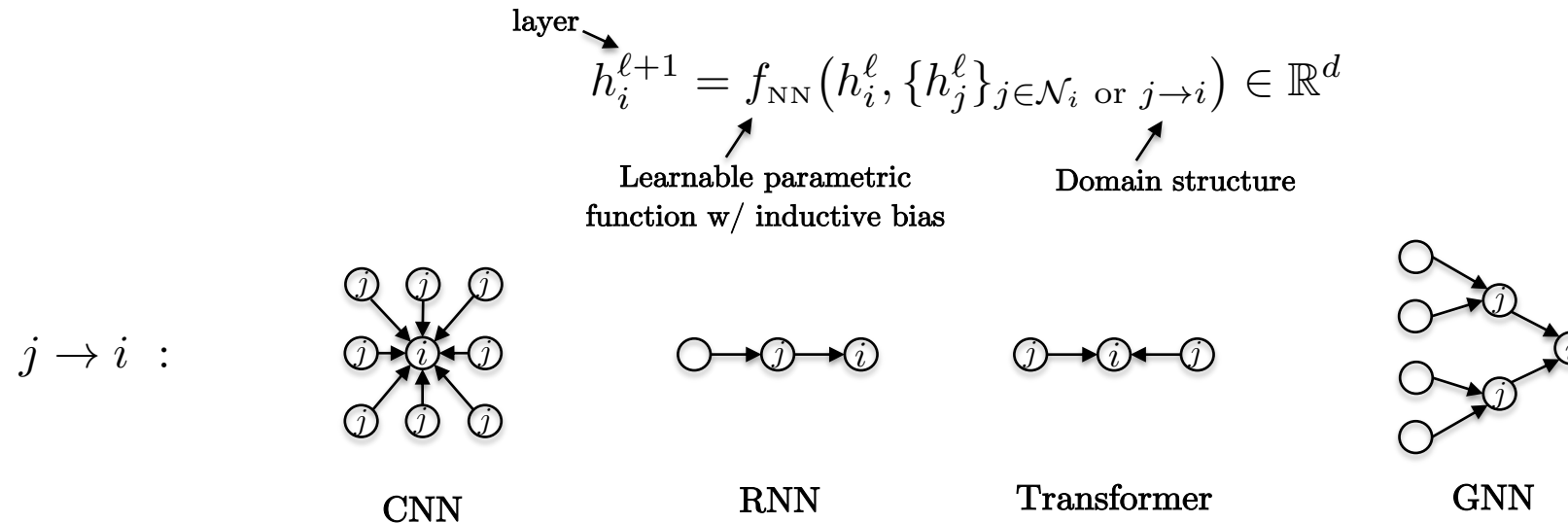
[1] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019

[2] Abboud, Ceylan, Grohe, Lukasiewicz, The surprising power of graph neural networks with random node initialization, 2020

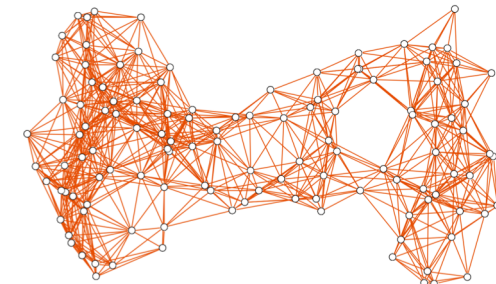
[3] Balciilar, Heroux, Gauzere, Vasseur, Adam, Honeine, Breaking the limits of message passing graph neural networks, 2021

Message-passing mechanism

- Most neural networks are message-passing techniques s.a. CNNs, RNNs, Transformers, GNNs.
 - New representation is computed by aggregating (with f_{NN}) the neighborhood information $j \rightarrow i$:



- As the MP mechanism is local for CNNs/RNNs/GNNs, they require to stack multiple layers to propagate information over long-range distances (ideally over the whole domain).

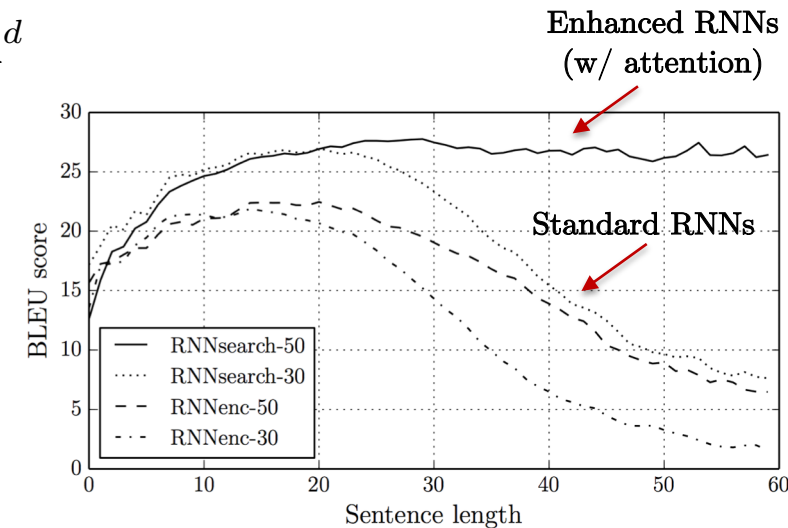


Over-squashing

- Stacking layers to propagate information through the domain can be a major issue.
 - The reception field grows as $O(L^2)$ for CNNs, $O(L)$ for RNNs and $O(2^{L+1})$ for GNNs (for tree graphs), L being the number of layers.
- The reception field size gives the number of neighbors used to update the representation.
 - For RNNs, this requires aggregating $O(L)$ number of vectors and $O(2^{L+1})$ for GNNs.

$$\begin{aligned} h_i^{\ell+1} &= F_{\text{RNN}}(h_i^0, h_{i-1}^0, h_{i-2}^0, \dots, h_{i-L}^0) \in \mathbb{R}^d \\ &= F_{\text{GNN}}(h_i^0, h_{i-1}^0, h_{i-2}^0, \dots, h_{i-2^{L+1}}^0) \in \mathbb{R}^d \end{aligned}$$

- This issue is known as over-squashing.
 - NNs w/ local reception field cannot learn long-range dependency.
 - RNNs cannot process sequences more than 30 words^[1].
 - MP-GNNs cannot stack many layers.

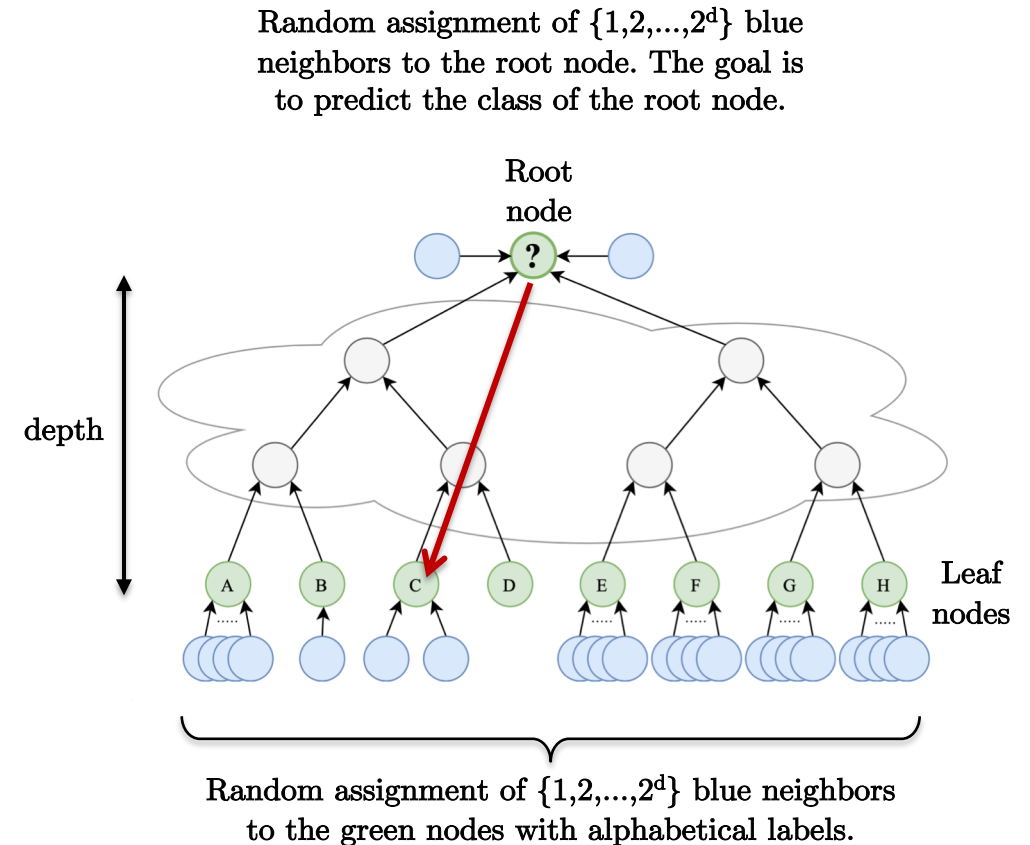


Figure^[1]. Accuracy of translated sequences (higher is better).

[1] Bahdanau, Cho, Bengio, Neural machine translation by jointly learning to align and translate, 2014

Synthetic graph for over-squashing

- TreeNeighboursMatch^[1] is a synthetic dataset to simulate the exponentially-growing receptive field in GNNs and controls the intensity of over-squashing with the tree depth.
- The experiment consists in classifying the root node with the letter of the green labeled node with the same number of blue neighbors than the root node.
- To succeed, the network must be able to communicate across long distance (from the root to the leaf) and extract the information from $O(2^{L+1})$ aggregated nodes.
- Standard MP-GNNs s.a. GCN, GAT and GIN fail to generalize for a depth ≥ 4 because of over-squashing.



[1] Alon, Yahav, On the bottleneck of graph neural networks and its practical implications, 2020

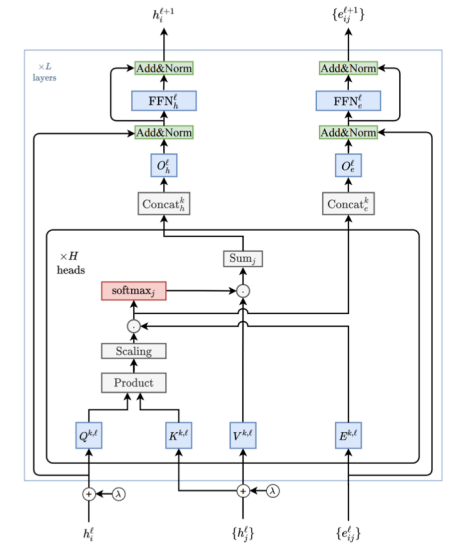
Graph Transformers^[1]

- Contributions

- Generalize Transformer to graphs to overcome over-squashing and poor long-range dependency (same idea from NLP^[2,3]).
- Leverage graph as topological inductive bias. Generalize cos/sin positional encoding to graphs (node ordering) with Laplacian eigenvectors. Introduce edge features through bi-linear product (for molecular bounds).
- Several improvements : SAN^[4], GraphiT^[5], Graphormer^[6]
- Popular class of architectures in biology (top 1&2 winners at NeurIPS'22 OGB Large-Scale Challenge competition)^[7]

- Limitations

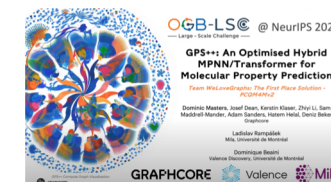
- Quadratic complexity $O(N^2+E)$, limited to small molecular graphs.



OGB-LS @ NeurIPS 2022
— Large - Scale Challenge —

- 10:50PM–11:40PM, UTC: Graph-Level Track (PCQM4Mv2)

- Intro to the task (5min) [\[Video\]](#)
- Live presentation by 1st place winner: **WeLoveGraphs** (10min) [\[Video\]](#)
- Live presentation by 2nd place winner: **NVIDIA-PCQM4Mv2** (10min) [\[Video\]](#)
- Live presentation by 2nd place winner: **VISM** (10min) [\[Video\]](#)
- Live Q&A and discussion (15min) [\[Video\]](#)



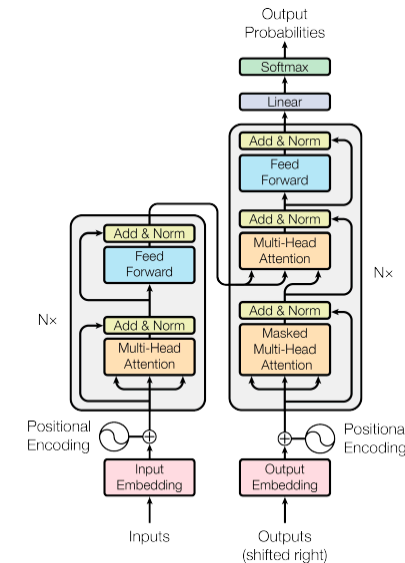
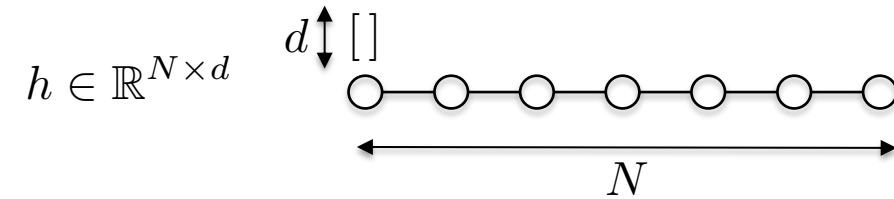
- [1] Dwivedi, Bresson, A generalization of transformer networks to graphs, AAAI 2021
- [2] Bahdanau, Cho, Bengio, Neural machine translation by jointly learning to align and translate, 2014
- [3] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin, Attention is all you need, 2017
- [4] Kreuzer, Beaini, Hamilton, Letourneau, Tossou, Rethinking Graph Transformers with Spectral Attention, 2021
- [5] Mialon, Chen, Selosse, Mairal, GraphiT: Encoding Graph Structure in Transformers, 2021
- [6] Ying, Cai, Luo, Zheng, Ke, He, Shen, Liu, Do Transformers Really Perform Bad for Graph Representation? 2021
- [7] <https://ogb.stanford.edu/neurips2022/workshop>

Outline

- DL for molecular science
- Review of graph network architectures
- **ViT/MLP-Mixer for images**
- Why MLP-Mixer for graphs?
- Proposed architecture
- Numerical experiments
- Conclusion

Transformer for long sequences

- Key ingredients of original Transformer^[1] :
 - Attention mechanism (self/cross attention)
 - Layer normalization + residual connection + MLP
 - Cos/sin functions as node positional encoding
- But attention mechanism is costly :
 - The reception field is the size of the domain $O(N)$.
 - Memory/speed complexity is $O(N^2d)$
 - This limits Transformer on GPUs to short sequences, $N \leq 1,000$ for $d=512$.
- How to scale-up to long sequences?
 - Linear approximations of the attention function^[2,3]



$$h^{\ell+1} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \in \mathbb{R}^{N \times d}$$

$$Q = h^\ell W_Q^\ell \in \mathbb{R}^{N \times d}$$

$$K = h^\ell W_K^\ell \in \mathbb{R}^{N \times d}$$

$$V = h^\ell W_V^\ell \in \mathbb{R}^{N \times d}$$

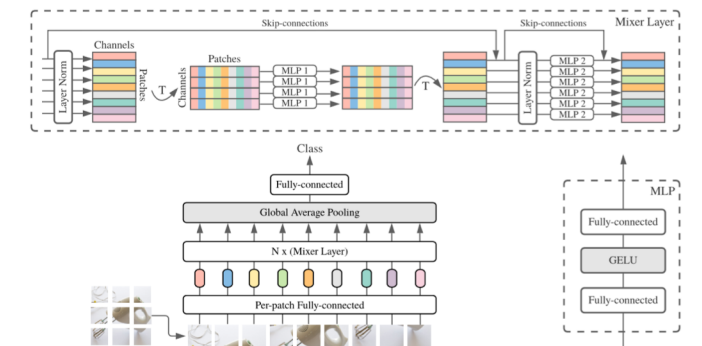
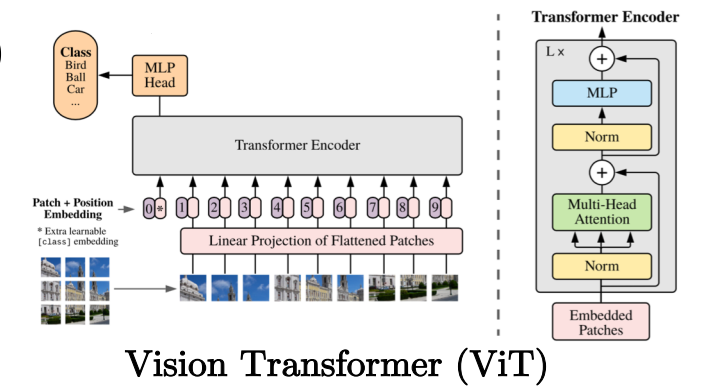
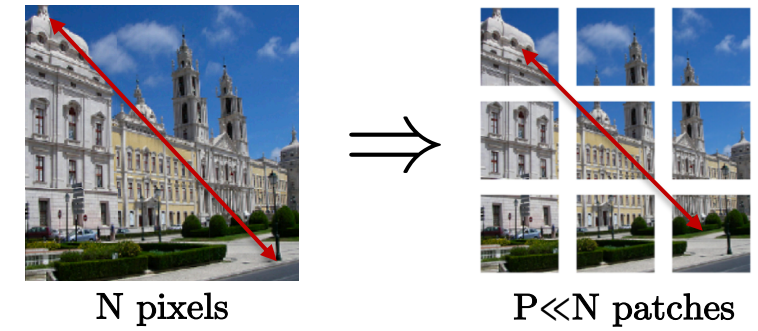
[1] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin, Attention is all you need, 2017

[2] Choromanski et-al, Rethinking attention with performers, 2020

[3] Jaegle, Gimeno, Brock, Vinyals, Zisserman, Carreira, Perceiver: General perception with iterative attention, 2021

ViT/MLP-Mixer for images

- How to scale Transformer to image grids (CV)?
E.g. $N = 1,000 \times 1,000$ pixels = 1M pixels/nodes
- ViT architecture^[1] :
 - Perform attention not on image pixels but on image patches :
 $O(N^2d) \searrow O(P^2d)$ w/ $P \ll N$ patches
 - Embed patches with MLP (or CNN with stride/kernel = patch size)
 - Same layer normalization + residual connection + MLP
 - Raster ordering of patches as node positional encoding
- MLP-Mixer architecture^[2] :
 - Replacing the costly attention function with simple MLPs reduces complexity from $O(P^2d) \searrow O(Pd)$ with comparable performance.
 - Token embedding + token mixer layer is enough to capture long-range dependency in images.



[1] Dosovitskiy et-al, An image is worth 16x16 words: Transformers for image recognition at scale, 2020

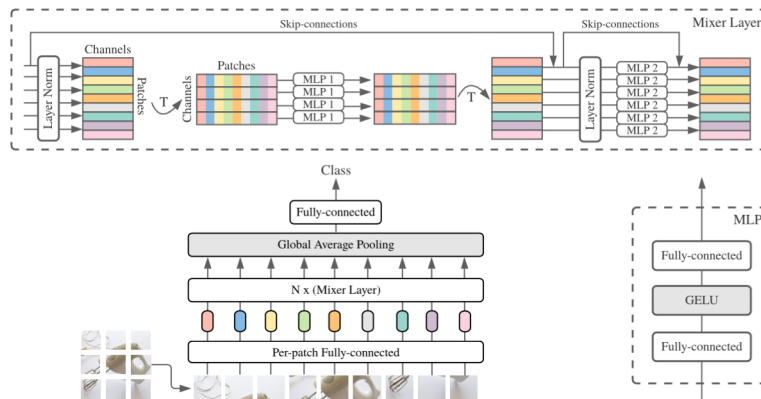
[2] Tolstikhin et-al, MLP-mixer: An all-MLP architecture for vision, 2021

Outline

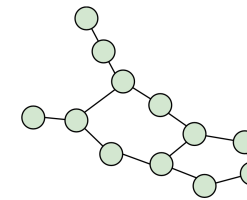
- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- **Why MLP-Mixer for graphs?**
- Proposed architecture
- Numerical experiments
- Conclusion

Why MLP-Mixer for graphs?

- Standard MLP-Mixer^[2] captures long-range interaction in images with low complexity.
- Our goal is to transfer these advantages to GNNs and design a new network that simultaneously
 - captures long-range dependency (mitigating the over-squashing issue),
 - keeps linear speed/memory complexity (similar to standard MP-GNNs),
 - achieves isomorphism expressivity (good representation power).
- However, generalizing MLP-Mixer is challenging due to the variable nature of graphs.



MLP-Mixer for images^[1]



MLP-Mixer for graphs

[1] Tolstikhin et-al, MLP-mixer: An all-MLP architecture for vision, 2021

Generalization challenges

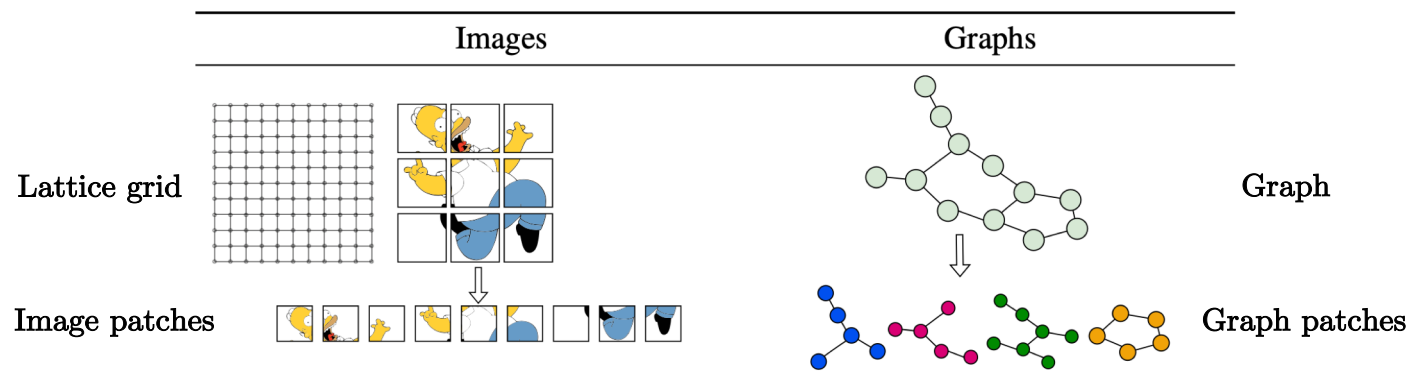
We identify four key challenges to generalize MLP-Mixer from grids to graphs.

(1) How to define and quickly extract graph patches(/tokens)?

- Images are supported by a regular lattice, easily split into same-size grid-like patches via fast pixel reordering.
- Graphs are irregular and cannot be divided into similar patches.

(2) How to encode graph patches into a vectorial representation?

- Same-size patches can be encoded with MLP.
- Graph patches have different topological structures (variable #nodes and #edges)
- Embedding process must be invariant to index permutation.



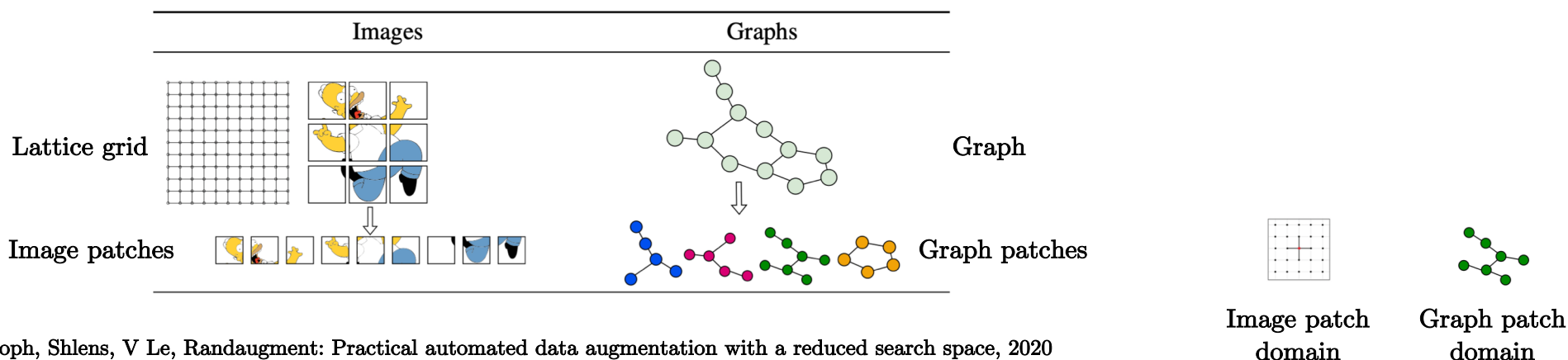
Generalization challenges

(3) How to preserve node and patch positional information?

- Image patches have implicit positions since images are always ordered the same way, but graphs are naturally not-aligned and the set of graph patches are therefore unordered.
- Pixels in each patch are ordered the same way, but nodes in graph tokens are generally unordered.

(4) How to reduce over-fitting for graphs?

- MLP-Mixer architectures are strong over-fitters. Images have a rich set of data augmentation and regularization techniques, e.g. cropping, flipping, RandAugment^[1], mixup^[2].
- Graph data augmentation methods^[3] are not yet as effective.



[1] Cubuk, Zoph, Shlens, V Le, Randaugment: Practical automated data augmentation with a reduced search space, 2020

[2] Zhang, Cisse, Dauphin, Lopez-Paz, mixup: Beyond empirical risk minimization, 2017

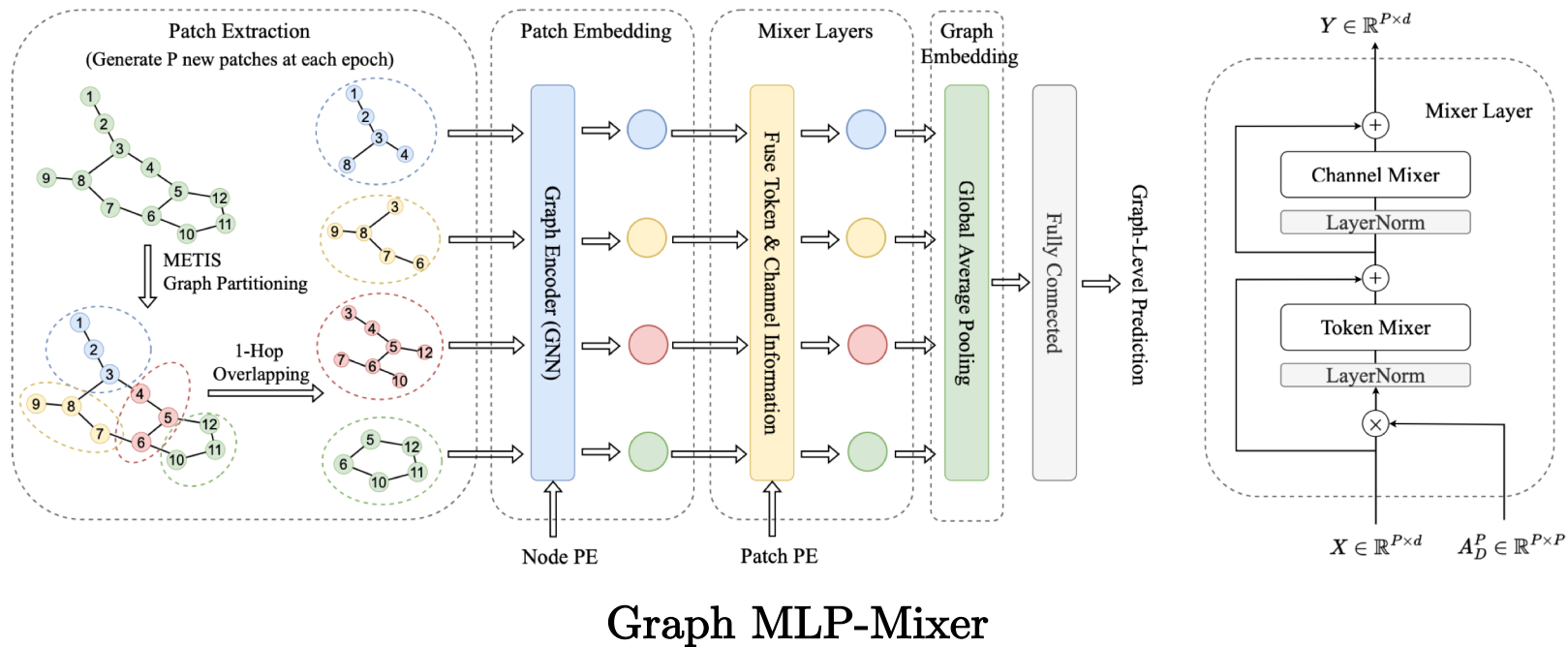
[3] Zhao, Liu, Neves, Woodford, Jiang, Shah, Data augmentation for graph neural networks, 2020

Outline

- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- **Proposed architecture**
- Numerical experiments
- Conclusion

Proposed architecture^[1]

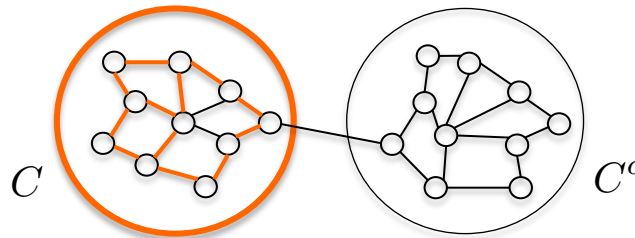
- Generic architecture of Graph MLP-Mixer is illustrated below.
- Implementation of each building block is arbitrary.
- Our choices lead to a simple framework that provides speed and accuracy.



[1] He, Hooi, Laurent, Perold, LeCun, Bresson, A Generalization of ViT/MLP-Mixer to Graphs, 2022

Patch extraction

- Graph patch extraction algorithm must satisfy the following conditions :
 - The clustering algorithm can be applied to any arbitrary graph.
 - The nodes in the patch must be more connected than for those outside the patch (s.a. compound of atoms).
 - The extraction must be fast, i.e. complexity at most linear w.r.t. the number of edges $O(E)$.
- Graph partitioning algorithms have a long and rich development^[1,2].
- They are NP-hard combinatorial problems and approximations are required.
- We select Metis^[3] that
 - Partitions a graph by maximizing the number of within-cluster links.
 - Presents one of the best trade-off accuracy and speed.



$$\max_C \frac{\text{Assoc}(C, C)}{\text{Vol}(C)} + \frac{\text{Assoc}(C^c, C^c)}{\text{Vol}(C^c)}$$
$$\text{Assoc}(C, C) = \sum_{i \in C, j \in C} A_{ij} \quad (\text{Adj. matrix})$$
$$\text{Vol}(C) = \sum_{i \in C} d_i \quad (\text{degree})$$

Normalized Association

[1] Von Luxburg, A tutorial on spectral clustering, 2007

[2] Buluc, Meyerhenke, Safro, Sanders, Schulz, Recent advances in graph partitioning, 2016

[3] Karypis, Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, 1998

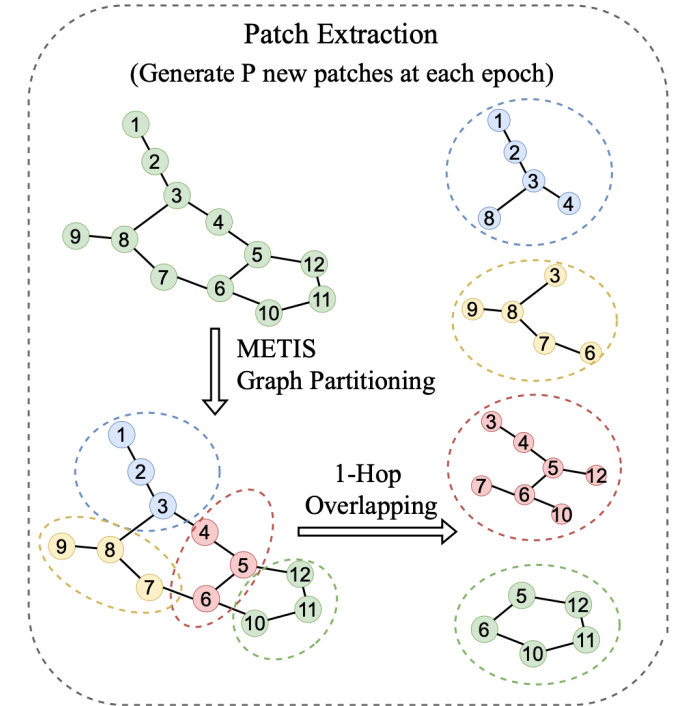
Metis limitations

- Two major limitations

- (1) Metis produces non-overlapping patches and loses the cutting edges. We expand the graph patches with a k-hop neighborhood.
- (2) Metis is a deterministic process, it always generates the same clusters which leads to over-fitting.

- Data augmentation

- At each epoch, Metis is applied on a perturbed graph (by randomly dropping a small set of edges) to get slightly different partitions.
- Then graph patches are extracted from the original graph (not the perturbed one) to retain the original nodes and edges.
- This new dropout technique produces distinct graph patches at each epoch that significantly improves the results at a small additional computational time.



Patch encoder

- Image patch embedding is fast and well-defined with a simple MLP.
- Graph patch embedding must handle heterogeneous structure, variable patch size, index permutation invariance, weight sharing and capable of distinguishing isomorphic patches.
- As a result, the graph patch encoder is a MP-GNN (e.g. GCN^[1], GAT^[2], GT^[3]) which can map a graph token into a fixed-size representation into 3 steps :

(1) Raw node and edge embedding :

$$h_i^0 = \underset{\text{Positional encoding}}{T^0} p_i + \underset{\text{Raw node features}}{U^0} \alpha_i + u^0 \in \mathbb{R}^d, \quad e_{ij}^0 = \underset{\text{Raw edge features}}{V^0} \beta_{ij} + v^0 \in \mathbb{R}^d$$

(2) Graph convolutional layers with MP-GNN :

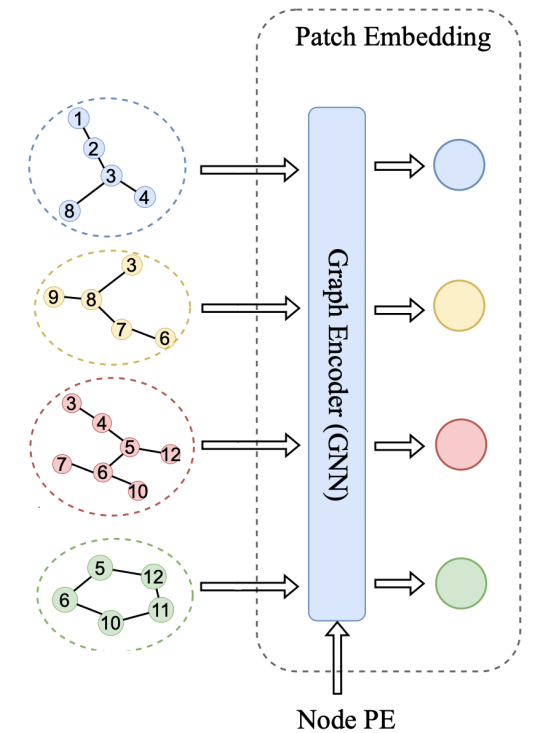
$$h_{i,p}^{\ell+1} = f_{\text{node}}(h_{i,p}^{\ell}, \{h_{j,p}^{\ell}, e_{ij,p}^{\ell} | j \in \mathcal{N}(i)\}) + g_{\text{patch-n}}(h_p^{\ell}) \in \mathbb{R}^d$$

$$e_{ij,p}^{\ell+1} = f_{\text{edge}}(h_{i,p}^{\ell}, h_{j,p}^{\ell}, e_{ij,p}^{\ell}) + g_{\text{patch-e}}(e_p^{\ell}) \in \mathbb{R}^d$$

(3) Pooling and readout : $x_{G_p} = \text{MLP}(h_p) \in \mathbb{R}^d$, $h_p = \frac{1}{|\mathcal{V}_p|} \sum_{i \in \mathcal{V}_p} h_{i,p}^{\ell=L} \in \mathbb{R}^d$

Patch embedding

- Note that the MP-GNN is applied to small patch graphs, which are not affected by poor long-range interaction and over-squashing.



[1] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
 [2] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph Attention Networks, 2018
 [3] Dwivedi, Bresson, A generalization of transformer networks to graphs, 2021

Positional information

- Regular grid offers an implicit arrangement for the sequence of image patches and for the pixels inside the image patch.
- General graphs do not have a canonical ordering of nodes and patches. This lack of positional information reduces the expressivity of the network.
- We use two explicit positional encoding (PE) :
 - Absolute node PE : We use random-walk structural encoding^[1] for molecular data and Laplacian eigenvectors^[2,3] encodings for synthetic graph datasets.
 - Relative patch PE : We use a n-step random walk diffusion process^[4] on the the adjacency matrix of the patch graphs (computed from the original graph adjacency matrix and the cluster extracted by Metis) :

$$A_D^P = (D^{-1} A^P)^n \in \mathbb{R}^{P \times P}$$

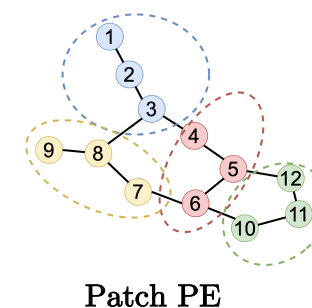
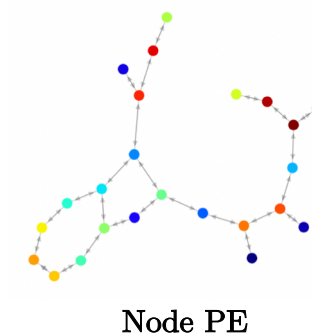
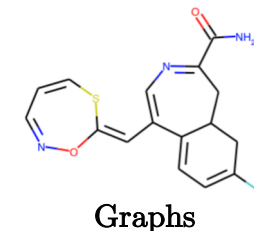
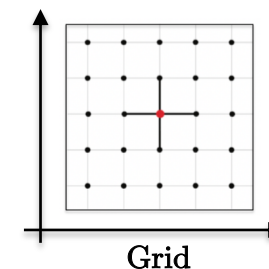
$$A_{ij}^P = |\mathcal{V}_i \cap \mathcal{V}_j| = \text{Cut}(\mathcal{V}_i, \mathcal{V}_j) = \sum_{k \in \mathcal{V}_i} \sum_{l \in \mathcal{V}_j} A_{kl}$$

[1] Dwivedi, Luu, Laurent, Bengio, Bresson, Graph neural networks with learnable structural and positional representations, 2021

[2] Dwivedi, Joshi, Laurent, Bengio, Bresson, Benchmarking graph neural networks, 2020

[3] Dwivedi, Bresson, A generalization of transformer networks to graphs, 2021

[4] Kondor, Vert, Diffusion kernels, 2014



Mixer layer and graph embedding

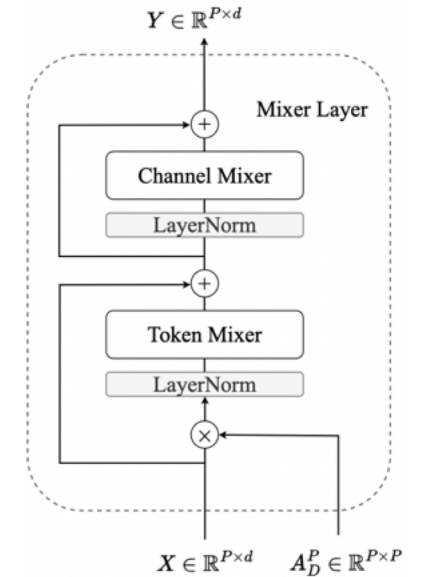
- The original mixer layer^[1] is a network that alternates channel and token mixing steps with two MLPs. These interleaved steps fuse token and channel information.
 - The simplicity of the mixer layer was important to understand the attention mechanism is not the only key component to get good performance.
 - This has led to a significant reduction in the computational cost, from $O(P^2d)$ to $O(Pd)$, with almost the same performance.

- Let $X \in \mathbb{R}^{P \times d}$ be the patch embedding matrix, then the graph mixer layer is defined as

$$\begin{aligned}
 U &= X + (W_2 \sigma(W_1 \text{LayerNorm}(\overset{\text{Patch PE}}{A_D^P X}))) \in \mathbb{R}^{P \times d} && \text{Token mixer} \\
 Y &= U + (W_4 \sigma(W_3 \text{LayerNorm}(U)^T))^T \in \mathbb{R}^{P \times d} && \text{Channel mixer}
 \end{aligned}$$

- The final graph-level representation is given by mean pooling all the (non-empty) patches and using a small MLP :

$$y_G = \text{MLP}(h_G) \in \mathbb{R} / \mathbb{R}^{n_c}, \quad h_G = \sum_p m_p \cdot x_{G_p} / \sum_p m_p \in \mathbb{R}^d$$



[1] Tolstikhin et-al, MLP-mixer: An all-MLP architecture for vision, 2021

Outline

- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- Proposed architecture
- **Numerical experiments**
- Conclusion

Numerical experiments

- We evaluate Graph MLP-Mixer on a range of benchmark graph datasets :
 - 1) Simulated datasets : CSL^[1], EXP^[2], SR25^[3] and TreeNeighbourMatch^[4] datasets.
 - 2) Small molecular datasets : ZINC from Benchmarking GNNs^[5] (for solubility) and MolTOX21 (for toxicity) and MolHIV from Open Graph Benchmark (OGB)^[6] (for HIV inhibition).
 - 3) Large molecular datasets : Peptides-func (for antibacterial, antiviral properties etc) and Peptides-struct (for 3D properties) from Long Range Graph Benchmark (LRGB)^[7]

[1] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019

[2] Abboud, Ceylan, Grohe, Lukasiewicz, The surprising power of graph neural networks with random node initialization, 2020

[3] Balcilar, Heroux, Gauzere, Vasseur, Adam, Honeine, Breaking the limits of message passing graph neural networks, 2021

[4] Alon, Yahav, On the bottleneck of graph neural networks and its practical implications, 2020

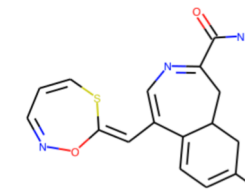
[5] Dwivedi, Joshi, Laurent, Bengio, Bresson, Benchmarking graph neural networks, 2020

[6] Hu et-al, Open graph benchmark: Datasets for machine learning on graphs, 2020

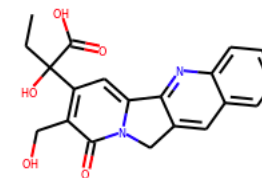
[7] Dwivedi, Rampavek, Galkin, Parviz, Wolf, Luu, Beaini, Long range graph benchmark, 2022

Benchmark datasets

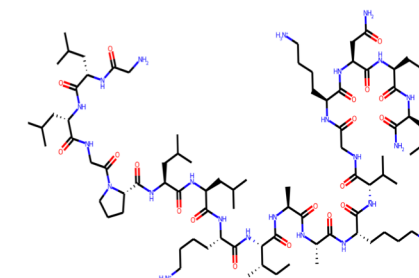
- Statistics of molecular and synthetic datasets :



ZINC

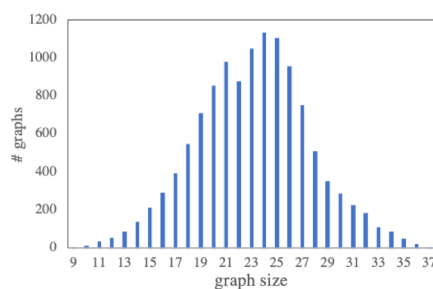


MolHIV

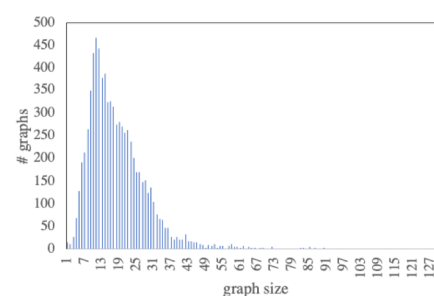


Peptide

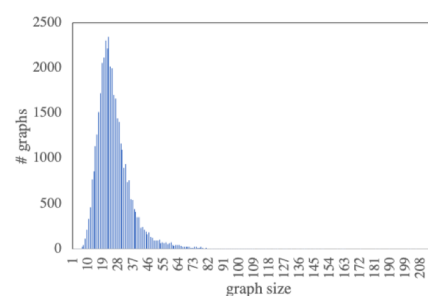
	Dataset	#Graphs	#Nodes	Avg. #Nodes	Avg. #Edges	Task	Metric
Simulated datasets	CSL	150	41	41	164	10-class classif.	Accuracy
	EXP	1,200	32-64	44.4	110.2	2-class classif.	Accuracy
	SR25	15	25	25	300	15-class classif.	Accuracy
	TreeNeighbourMatch (r=2)	96	7	7	6	4-class classif.	Accuracy
	TreeNeighbourMatch (r=3)	32,000	15	15	14	8-class classif.	Accuracy
	TreeNeighbourMatch (r=4)	64,000	31	31	30	16-class classif.	Accuracy
	TreeNeighbourMatch (r=5)	128,000	63	63	62	32-class classif.	Accuracy
	TreeNeighbourMatch (r=6)	256,000	127	127	126	64-class classif.	Accuracy
	TreeNeighbourMatch (r=7)	512,000	255	255	254	128-class classif.	Accuracy
	TreeNeighbourMatch (r=8)	640,000	511	511	510	256-class classif.	Accuracy
Small molecular datasets	ZINC	12,000	9-37	23.2	24.9	regression	MAE
	MolTOX21	7,831	1-132	18.57	38.6	12-task classif.	ROCAUC
	MolHIV	41,127	2-222	25.5	54.9	binary classif.	ROCAUC
Large molecular datasets	Peptides-func	15,535	8-444	150.9	307.3	10-class classif.	Avg. Precision
	Peptides-struct	15,535	8-444	150.9	307.3	regression	MAE



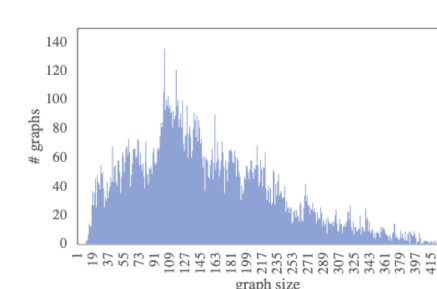
ZINC



MolTOX21



MolHIV



Peptides-func/struct

Comparison with standard MP-GNNs

- Graph MLP-Mixer lifts the performance of all base MP-GNNs across all datasets, which include GCN^[1], GatedGCN^[2], GINE^[3] and GraphTransformer^[4].
- We augmented all base models with the same PEs as Graph MLP-Mixer to ensure fair comparison.
- These promising results demonstrate the generic nature of our architecture which can be applied to any MP-GNN in practice.

Model	ZINC	MolTOX21	MolHIV	Peptide-func	Peptide-struct
	MAE ↓	ROCAUC ↑	ROCAUC ↑	Avg. Precision ↑	MAE ↓
GCN	0.1952 ± 0.0057	0.7525 ± 0.0031	0.7813 ± 0.0081	0.6328 ± 0.0086	0.2758 ± 0.0012
GCN-MLP-Mixer	0.1323 ± 0.0026	0.7829 ± 0.0058	0.7912 ± 0.0121	0.6810 ± 0.0067	0.2492 ± 0.0011
GatedGCN	0.1577 ± 0.0046	0.7641 ± 0.0057	0.7874 ± 0.0119	0.6300 ± 0.0029	0.2778 ± 0.0017
GatedGCN-MLP-Mixer	0.1249 ± 0.0020	0.7861 ± 0.0048	0.8073 ± 0.0037	0.6856 ± 0.0044	0.2484 ± 0.0016
GINE	0.1072 ± 0.0037	0.7730 ± 0.0064	0.7885 ± 0.0034	0.6405 ± 0.0077	0.2780 ± 0.0021
GINE-MLP-Mixer	0.0745 ± 0.0014	0.7866 ± 0.0022	0.7951 ± 0.0077	0.6921 ± 0.0054	0.2485 ± 0.0004
GraphTrans	0.1230 ± 0.0018	0.7646 ± 0.0055	0.7884 ± 0.0104	0.6313 ± 0.0039	0.2777 ± 0.0025
GraphTrans-MLP-Mixer	0.0798 ± 0.0032	0.7874 ± 0.0044	0.7892 ± 0.0116	0.6795 ± 0.0063	0.2475 ± 0.0015

Table: Results are averaged over 4 runs with 4 different seeds.

[1] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017

[2] Bresson, Laurent, Residual gated graph convnets, 2017

[3] Hu, Liu, Gomes, Zitnik, Liang, Pande, Leskovec, Strategies for pre-training graph neural networks, 2019

[4] Dwivedi, Bresson, A generalization of transformer networks to graphs, 2021

Comparison with SOTA results

- We compare against GNN models with SOTA results
 - Graph Transformers: GraphiT, GPS, SAN, etc
 - Expressive GNNs: GNN-AK₊ and SUN
- For small molecular graphs, our model achieved competitive results on ZINC and MolHIV.
- For larger molecular graphs, our model sets SOTA performance on Peptides-fun/struct.
- Graph MLP-Mixer offers better space-time complexity and scalability.
 - SAN+LapPE and SUN require 7.5× and 41× training time per epoch, and 12× and 18× memory respectively, compared to our model.

Model	ZINC	MolHIV	Peptides-func			Peptides-strcut		
	MAE ↓	ROCAUC ↑	Avg. Precision ↑	Time/Epoch	Memory	MAE ↓	Time/Epoch	Memory
GT [36]	0.226 ± 0.014	–	–	–	–	–	–	–
GraphiT [53]	0.202 ± 0.011	–	–	–	–	–	–	–
Graphormer [56]	0.122 ± 0.006	–	–	–	–	–	–	–
GPS [57]	0.070 ± 0.004	0.7880 ± 0.0101	0.6562 ± 0.0115	1.3×	6.9×	0.2515 ± 0.0012	1.1×	6.6×
SAN+LapPE [38]	0.139 ± 0.006	0.7775 ± 0.0061	0.6384 ± 0.0121	8.8×	12.4×	0.2683 ± 0.0043	7.4×	11.8×
SAN+RWSE [38]	–	–	0.6439 ± 0.0075	7.5×	19.6×	0.2545 ± 0.0012	6.5×	11.7×
GNN-AK+ [31]	0.080 ± 0.001	0.7961 ± 0.0119	0.6480 ± 0.0089	2.5×	7.8×	0.2736 ± 0.0007	2.1×	7.3×
SUN [32]	0.084 ± 0.002	0.8003 ± 0.0055 ²	0.6730 ± 0.0078	41.3×	18.8×	0.2498 ± 0.0008	35.7×	16.6×
Graph MLP-Mixer	0.075 ± 0.001	0.8073 ± 0.0037	0.6921 ± 0.0054	1.0×	1.0×	0.2475 ± 0.0015	1.0×	1.0×

Table: Results are averaged over 4 runs with 4 different seeds.

Graph MLP-Mixer mitigates over-squashing^[1]

- Standard MP-GNNs s.a. GCN, GGCN, GAT, GIN fail to generalize for depth ≥ 4 because of over-squashing that squeezes too much info from the exponential growth of the tree reception field.
- Our model is able to mitigate over-squashing and generalize until depth = 7 since it transmits the long-distance information directly with the mixer layer.

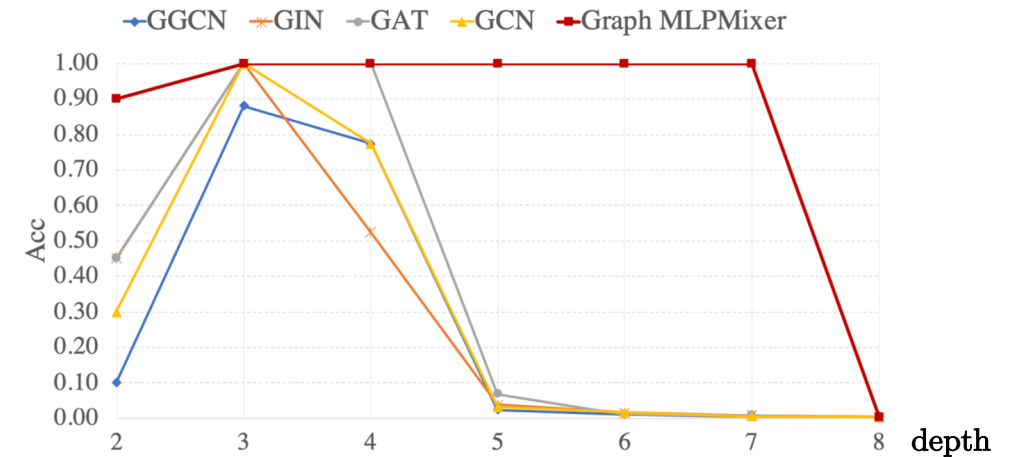
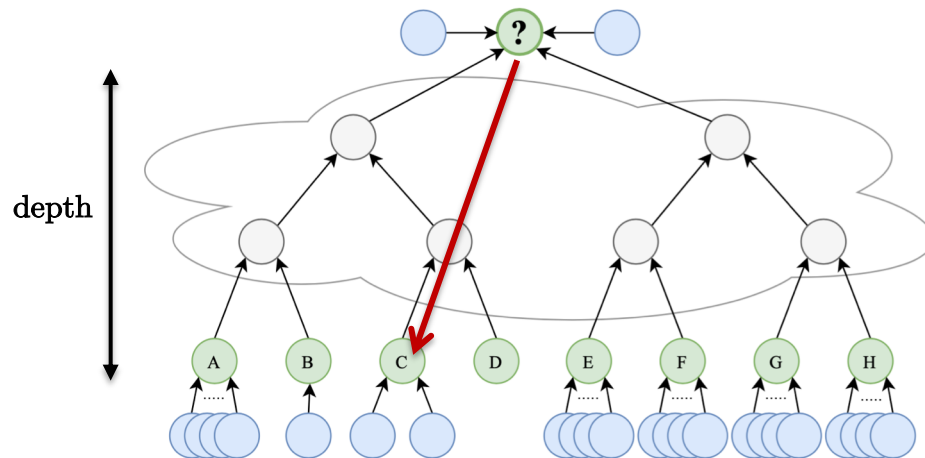


Figure. Test Accuracy w.r.t. the tree depth in the NEIGHBORMATCH problem.

[1] Alon, Yahav, On the bottleneck of graph neural networks and its practical implications, 2020

Graph MLP-Mixer achieves high expressivity

- Graph MLP-Mixer can be proved to be strictly more powerful than 1-WL.
 - Theorem^[1] : Laplacian eigenvectors or k-step Random Walk PE can distinguish non-isomorphic graphs for which the 1-WL test fails such as the CSL dataset^[2].
- Interestingly, experiments on EXP^[3] and SR25^[4] datasets show that our model is strictly more powerful than 1&2-WL and not less powerful than 3-WL (where all standard MP-GNNs fail).

Model	CSL (ACC)	EXP (ACC)	SR25 (ACC)
GCN	10.00 ± 0.00	51.90 ± 1.96	6.67 ± 0.00
GatedGCN	10.00 ± 0.00	51.73 ± 1.65	6.67 ± 0.00
GINE	10.00 ± 0.00	50.69 ± 1.39	6.67 ± 0.00
GraphTrans	10.00 ± 0.00	52.35 ± 2.32	6.67 ± 0.00
GCN-MLP-Mixer	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
GatedGCN-MLP-Mixer	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
GINE-MLP-Mixer	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
GraphTrans-MLP-Mixer	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00

Table: Results are averaged over 4 runs with 4 different seeds.

[1] Dwivedi, Luu, Laurent, Bengio, Bresson, Graph neural networks with learnable structural and positional representations, 2021

[2] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019

[3] Abboud, Ceylan, Grohe, Lukasiewicz, The surprising power of graph neural networks with random node initialization, 2020

[4] Balcilar, Heroux, Gauzere, Vasseur, Adam, Honeine, Breaking the limits of message passing graph neural networks, 2021

Ablation studies

- We evaluate various choices for each component of the architecture in the appendix of^[1].
 - Assess the benefits of Metis against random graph partitioning.
 - Evaluate the effect of number of graph patches.
 - Study the effect of patch overlapping with k-hop neighborhood extension.
 - Show the effects of node PE and patch PE.
 - Examine the effect of data augmentation.
 - Estimate the trade-off between performance and efficiency.
 - Compare the mixer layer vs. transformer layer as in ViT.

[1] He, Hooi, Laurent, Perold, LeCun, Bresson, A Generalization of ViT/MLP-Mixer to Graphs, 2022

Outline

- DL for molecular science
- Review of graph network architectures
- ViT/MLP-Mixer for images
- Why MLP-Mixer for graphs?
- Proposed architecture
- Numerical experiments
- **Conclusion**

Conclusion

- We introduce a novel GNN architecture that improves standard MP-GNN limitations (poor long-range dependency and low expressivity power) and has better complexity than Graph Transformers (linear complexity).
- We demonstrate its potential on small and large molecular datasets.
- Reproducibility
 - ArXiv paper : <https://arxiv.org/pdf/2212.13350.pdf>
 - GitHub repo : <https://github.com/XiaoxinHe/Graph-MLPMixer>
- Future work
 - Expand architecture to node and link prediction tasks
 - Prove network captures long-range dependency for trees and (possibly) more general graphs
 - Hierarchical architecture (next slide)
 - Speed up (next slide)

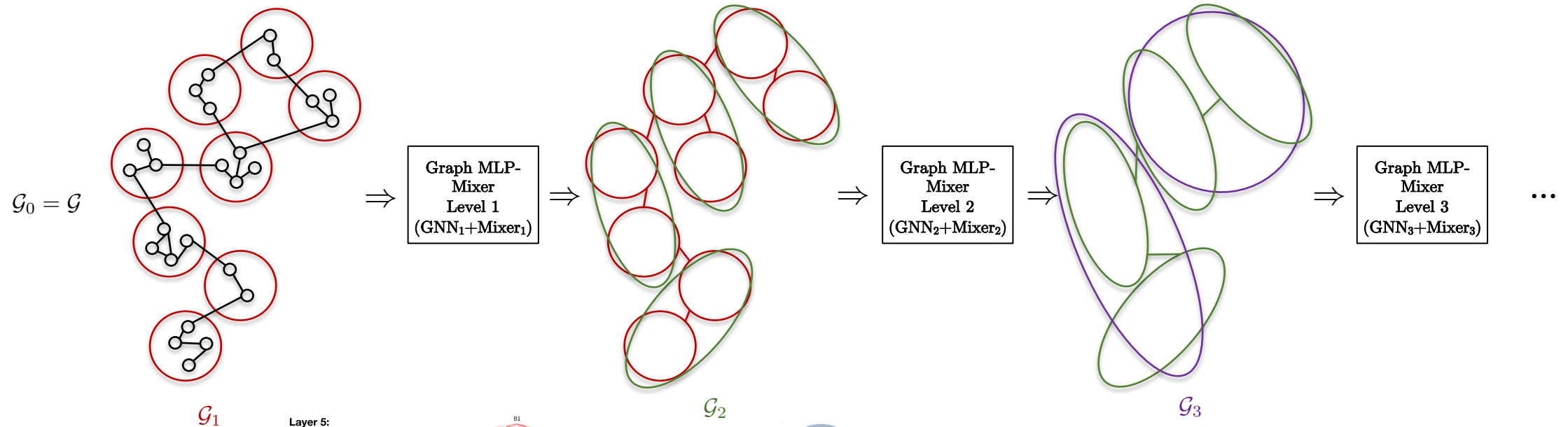
A GENERALIZATION OF ViT/MLP-MIXER TO GRAPHS

Xiaoxin He¹ Bryan Hooi¹ Thomas Laurent² Adam Perold³ Yann LeCun^{4,5} Xavier Bresson¹
{xiaoxin, bhooi, xaviercs}@comp.nus.edu.sg, tlaurent@lmu.edu
ap@discoverelement.com, yann@cs.nyu.edu

¹National University of Singapore ²Loyola Marymount University ³Element, Inc.
⁴New York University ⁵Meta AI

Future work

- Hierarchical architecture : Apply recursively the same architecture (with same/different MP-GNNs) at different coarsening levels.



Hierarchical scene graphs
Rosinol et-al, 2020

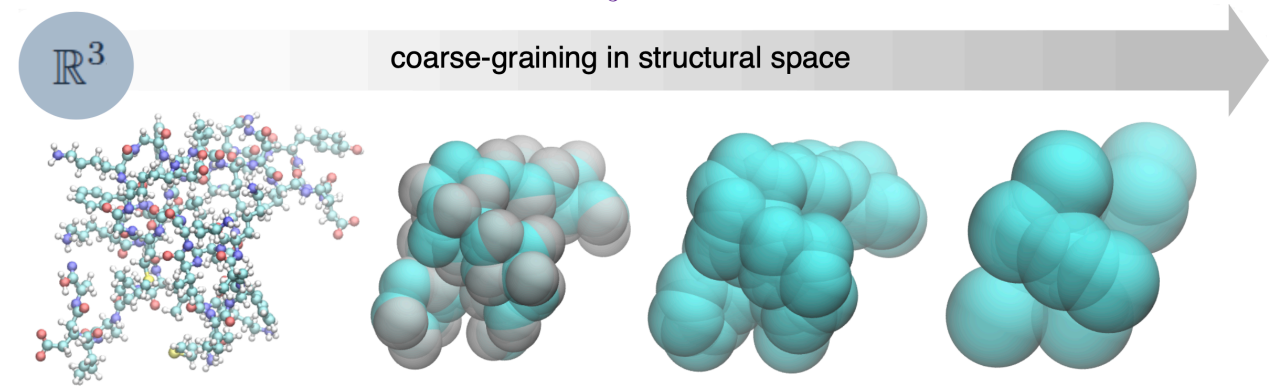
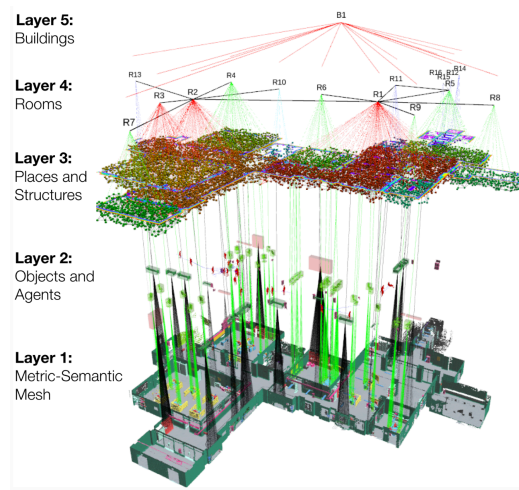


Figure from Cecilia Clementi (FU Berlin)

Future work

- Speed up : For balanced graph patches, it should be possible to drop specialized GNN libraries PyG^[1] or DGL^[2] by replacing sparse linear algebra operations with dense ones (optimized for GPUs).

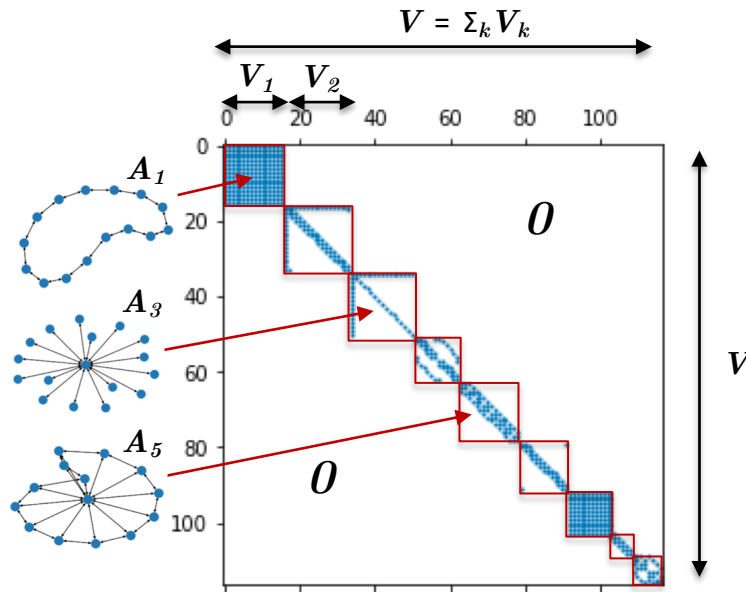


Figure 1. Adjacency matrix A^* of a batch of graphs in PyG and DGL (sparse representation)

$$A^* = \text{block_diag}(A_1, \dots, A_K) \in \mathbb{R}^{V \times V}$$

$$\text{with } V = \sum_{k=1}^K V_k$$

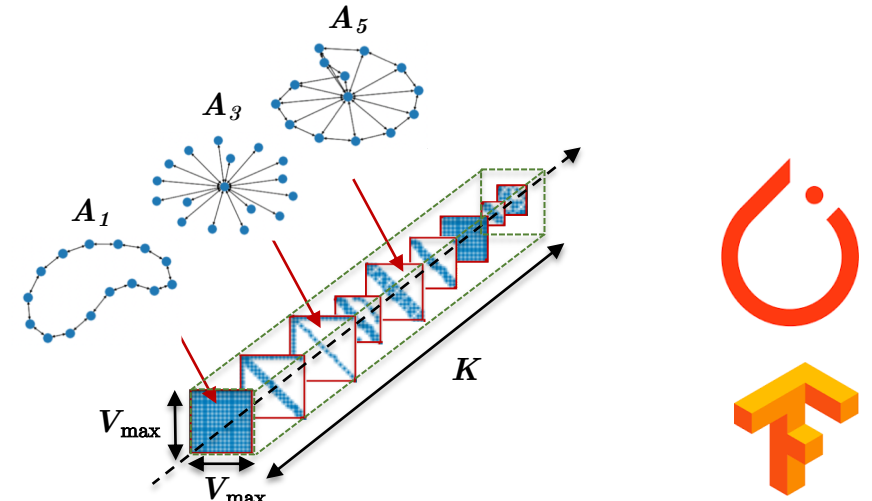


Figure 2. Adjacency matrix A^+ of a batch of small graphs (dense representation)

$$A^+ = \text{concat}(A_1, \dots, A_K) \in \mathbb{R}^{V_{\max} \times V_{\max} \times K}$$

$$\text{with } V_k \leq V_{\max} \forall k \text{ and zero padding}$$

$$\text{nb_zeros}(A^+) \ll \text{nb_zeros}(A^*)$$

[1] Fey and Lenssen, Fast graph representation learning with pytorch geometric, 2019

[2] Wang et-al, Deep graph library: Towards efficient and scalable deep learning on graphs, 2019



Thank you

Xavier Bresson
xaviercs@nus.edu.sg

<https://twitter.com/xbresson>

<https://scholar.google.com/citations?user=9pSK04MAAAAJ>

https://www.youtube.com/channel/UCeONAtqVKCS30Xn6zy1YQ_g

<https://github.com/xbresson>

<https://www.linkedin.com/in/xavier-bresson-738585b>

<https://www.facebook.com/xavier.bresson.1>

<https://graphdeeplearning.github.io>

<https://www.comp.nus.edu.sg/cs/people/xaviercs>