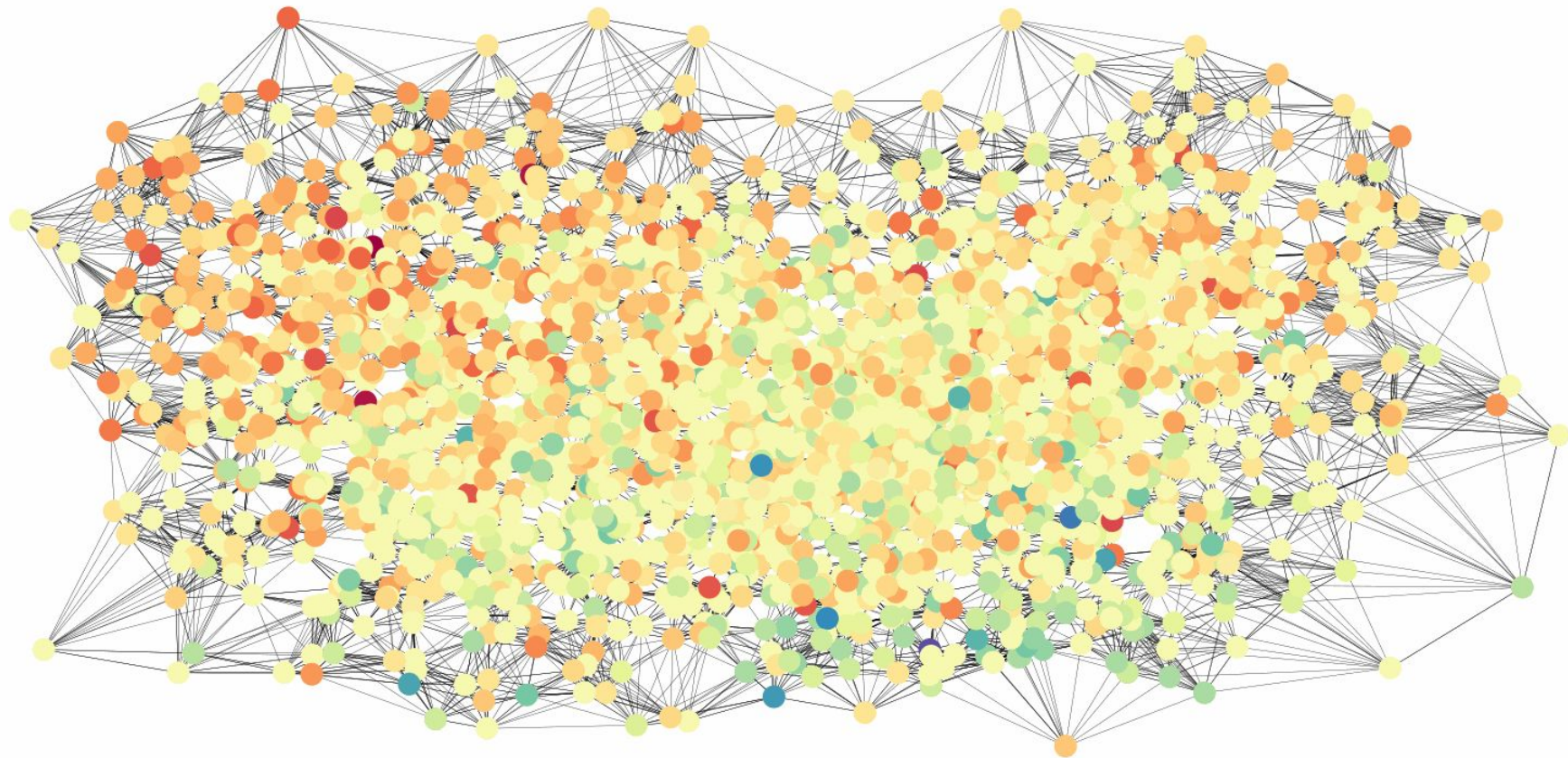


# DL в графах

НИС 4.12 Потапов Юрий

Beltrami Flow, diffusion time=0



# Для каких задач нужен графовый DL?

## Задачи на вершинах

- **Классификация вершин:**  
соцсети -> определение типа пользователя на основе его активности.
- **Предсказание свойств вершин:**  
соцсети -> предсказание возраста, пола или интересов пользователя основе его связей и взаимодействий.

## Задачи на ребрах

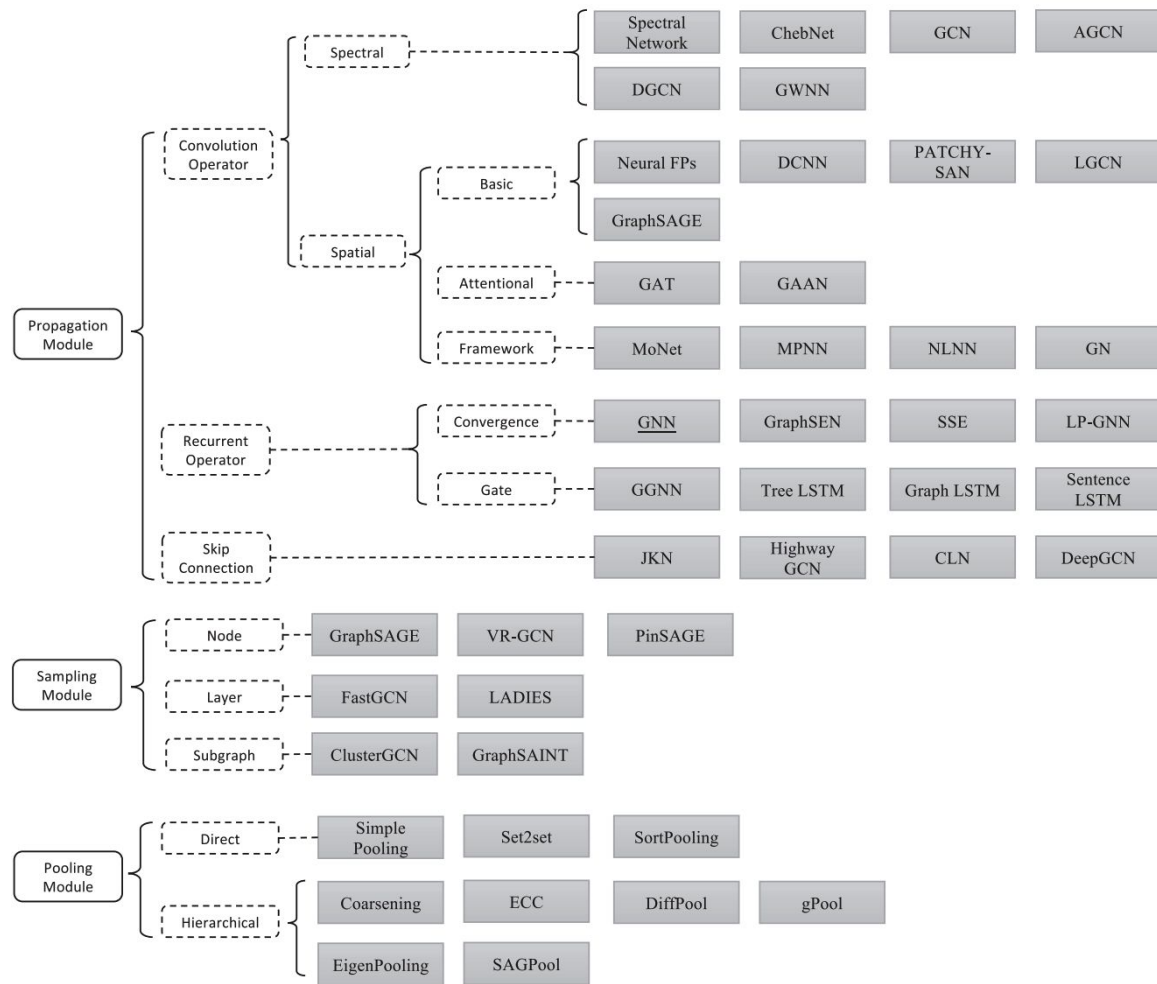
- **Прогнозирование связей:**  
соцсети -> вероятности появления связей между вершинами, например, в прогнозировании дружбы между пользователями.
- **Детекция аномалий:**  
финансы -> выявление аномальных связей в сети, таких как подозрительные транзакции в финансовых графах.

## Задачи на графах

- **Графовая классификация:**  
классификации целых графов.  
Дороги -> дать характеристику графу дорог, его плюсы и минусы
- **Анализ схожести:**  
определение структурной или функциональной схожести между различными графами (любой домен)

## Совместные задачи

- **Графовая генерация:**  
Генерация новых графов с заданными свойствами, например, создание химических молекул с определенными химическими свойствами.



# Типы модулей в графовых нейронных сетях

Fig. 3. An overview of computational modules.

# Появление графовых нейронок (2005г.)

The state  $x_n$  is defined as the solution of the system of equations:

$$x_n = f_w(l_n, x_{ne[n]}, l_{ne[n]}), \quad n \in N \tag{1}$$

where  $l_n, x_{ne[n]}, l_{ne[n]}$  are the label of  $n$ , and the states and the labels of the nodes in the neighborhood of  $n$ , respectively.

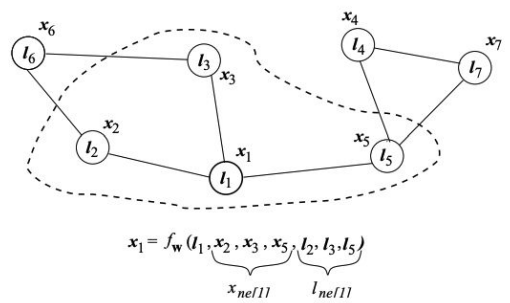


Fig. 2. State  $x_1$  depends on the neighborhood information.

For each node  $n$ , an output vector  $o_n \in \mathbb{R}^m$  is also defined which depends on the state  $x_n$  and the label  $l_n$ . The dependence is described by a parametric *output function*  $g_w$

$$o_n = g_w(x_n, l_n), \quad n \in N. \tag{2}$$

Let  $x$  and  $l$  be respectively the vectors constructed by stacking all the states and all the labels. Then, Equations (1) and (2) can be written as:

$$\begin{aligned} x &= F_w(x, l) \\ o &= G_w(x, l) \end{aligned} \tag{3}$$

А дальше авторы  
минимизируют  $t_i -$   
нужный выход на вершине  
 $\phi_w = o_n$ .

$$e_w = \sum_{i=1}^p (t_i - \phi_w(G_i, n_i))^2.$$

F и G - multilayer perceptron / matrix



# Сверточная графовая нейронная сеть (2014г.)

$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$

Node  $v$ 's  
initial  
embedding.  
... is just node  $v$ 's  
original features.

and for  $k = 1, 2, \dots$  upto  $K$ :

$$h_v^{(k)} = f^{(k)} \left( W^{(k)} \cdot \frac{\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)} \right) \quad \text{for all } v \in V.$$

Node  $v$ 's  
embedding at  
step  $k$ .

Mean of  $v$ 's  
neighbour's  
embeddings at  
step  $k - 1$ .

Node  $v$ 's  
embedding at  
step  $k - 1$ .

Color Codes:

- Embedding of node  $v$ .
- Embedding of a neighbour of node  $v$ .
- (Potentially) Learnable parameters.

# Сверточная графовая нейронная сеть (2017г)

$$H^{(l+1)} = \sigma \left( \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right)$$

$$H^{(l+1)} = \sigma \left( \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right)$$

где:

- $H^{(l)}$  имеет размерность  $N \times F^{(l)}$ ,
- $W^{(l)}$  имеет размерность  $F^{(l)} \times F^{(l+1)}$ ,
- $\hat{A} = A + I$  имеет размерность  $N \times N$  (где  $N$  - количество узлов в графе),
- $\hat{D}$  - диагональная матрица степеней  $\hat{A}$  размерности  $N \times N$ ,
- $\sigma$  - функция активации.

<https://arxiv.org/pdf/1312.6203.pdf>

<https://arxiv.org/pdf/1609.02907.pdf>



# Graph Sage Convolution Network

$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$




Node  $v$ 's initial embedding. ... is just node  $v$ 's original features.

and for  $k = 1, 2, \dots$  upto  $K$ :

$$h_v^{(k)} = f^{(k)} \left( W^{(k)} \cdot \left[ \text{AGG}_{u \in \mathcal{N}(v)}(\{h_u^{(k-1)}\}), h_v^{(k-1)} \right] \right) \quad \text{for all } v \in V.$$

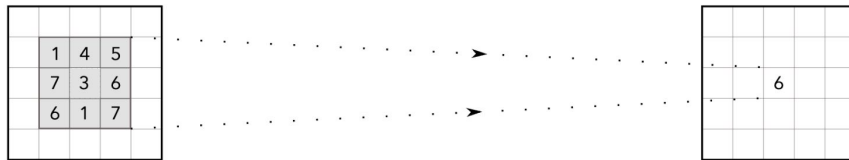
Node $v$ 's embedding at step $k$ .	Aggregation of $v$ 's neighbour's embeddings at step $k - 1$ ...	... Node $v$ 's embedding at step $k - 1$ .
	... concatenated with ...	

Color Codes:

-  Embedding of node  $v$ .
-  Embedding of a neighbour of node  $v$ .
-  (Potentially) Learnable parameters.

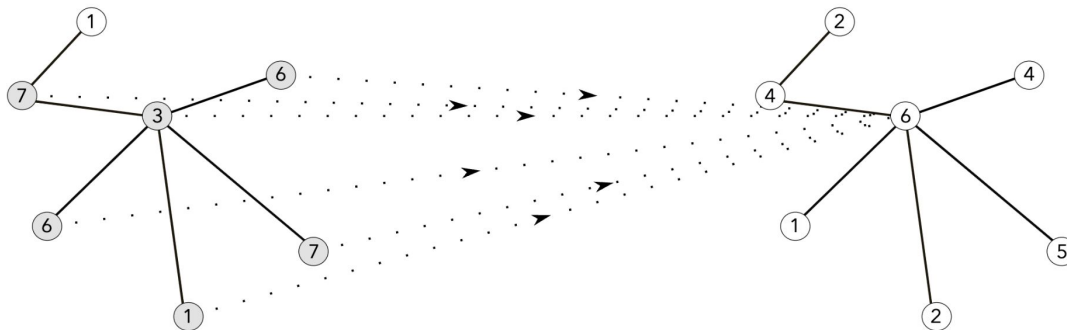


# ChebNet (2017)



## Convolution in CNNs

Convolutions in CNNs are inherently localized.  
Neighbours participating in the convolution at the  
center pixel are highlighted in gray.

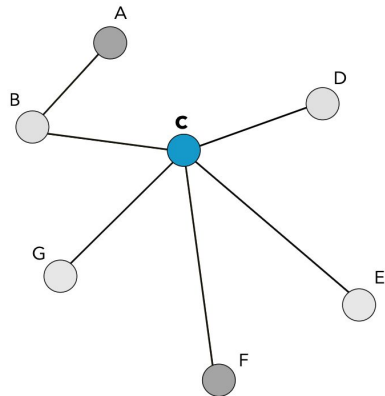


## Localized Convolution in GNNs

GNNs can perform localized convolutions mimicking CNNs. Hover over  
a node to see its immediate neighbourhood highlighted on the left.  
The structure of this neighbourhood changes from node to node.

# Как это работает?

Then, the graph Laplacian  $L$  is the square  $n \times n$  matrix defined as:  $L = D - A$ .



Input Graph  $G$

	A	B	C	D	E	F	G
A	1	-1					
B	-1	2	-1				
C	-1	-1	5	-1	-1	-1	-1
D		-1		1			
E		-1			1		
F		-1				1	
G		-1					1

Laplacian  $L$  of  $G$

## Polynomials of the Laplacian

Now that we have understood what the graph Laplacian is, we can build polynomials of the form:

$$p_w(L) = w_0 I_n + w_1 L + w_2 L^2 + \dots + w_d L^d = \sum_{i=0}^d w_i L^i.$$

Each polynomial of this form can alternately be represented by its vector of coefficients  $w = [w_0, \dots, w_d]$ . Note that for every  $w$ ,  $p_w(L)$  is an  $n \times n$  matrix, just like  $L$ .

These polynomials can be thought of as the equivalent of 'filters' in CNNs, and the coefficients  $w$  as the weights of the 'filters'.

Once we have constructed the feature vector  $x$ , we can define its convolution with a polynomial filter  $p_w$  as:

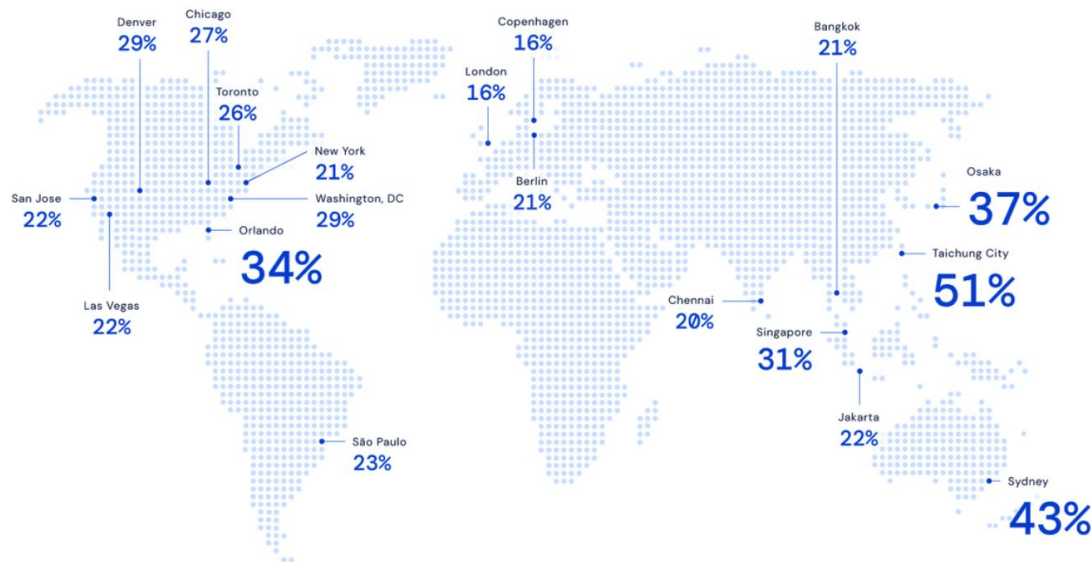
$$x' = p_w(L) x$$

Далее можно придумывать специальные многочлены (ChebNet), нормализовать Лапласиан.

Обучение далее не будет особо отличаться от других графов

# GNN usage in 2019

Google Maps ETA Improvements Around the World



The ever-industrious DeepMind researchers meanwhile have been working on further improving Google Maps, and this week the UK-based AI company and research lab unveiled a partnership with Google Maps that has leveraged advanced Graph Neural Networks (GNNs) to improve estimated time of arrival (ETA) accuracy.

The coordinated efforts have boosted the accuracy of real-time ETAs by up to 50 percent in cities such as Berlin, Jakarta, São Paulo, Sydney, Tokyo and Washington DC.

# Graph Attention Network

$$h_v^{(0)} = x_v \quad \text{for all } v \in V.$$

Node  $v$ 's  
initial  
embedding. ... is just node  $v$ 's  
original features.

and for  $k = 1, 2, \dots$  upto  $K$ :

$$h_v^{(k)} = f^{(k)} \left( W^{(k)} \cdot \left[ \sum_{u \in \mathcal{N}(v)} \alpha_{vu}^{(k-1)} h_u^{(k-1)} + \alpha_{vv}^{(k-1)} h_v^{(k-1)} \right] \right) \quad \text{for all } v \in V.$$

Node  $v$ 's  
embedding at  
step  $k$ .

Weighted mean of  
 $v$ 's neighbour's  
embeddings at  
step  $k - 1$ .

Node  $v$ 's  
embedding at  
step  $k - 1$ .

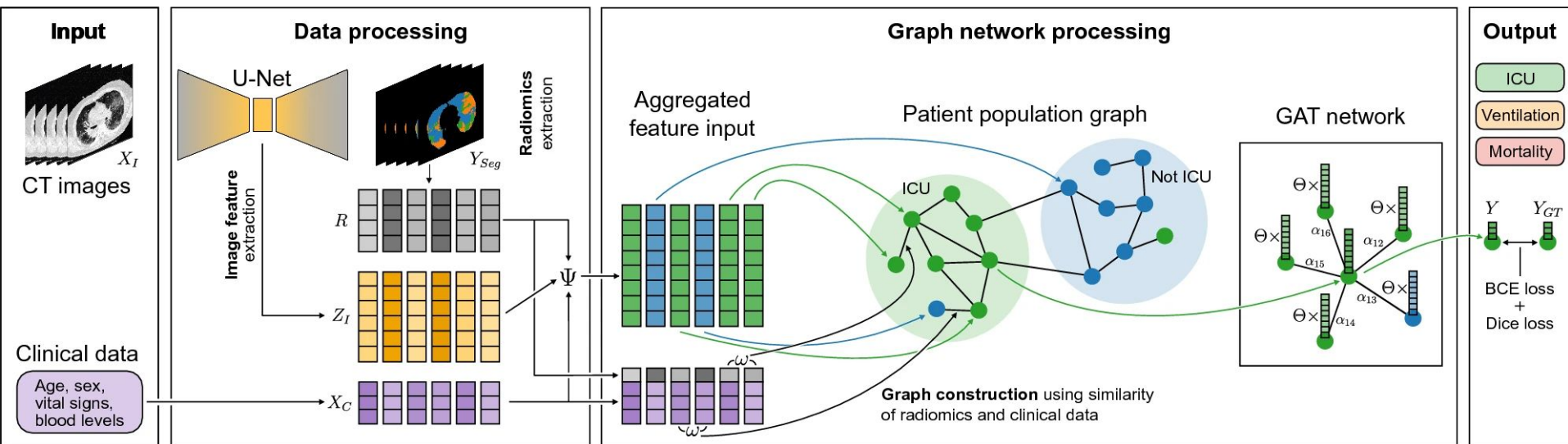
where the attention weights  $\alpha^{(k)}$  are generated by an attention mechanism  $A^{(k)}$ , normalized such that the sum over all neighbours of each node  $v$  is 1:

$$\alpha_{vu}^{(k)} = \frac{A^{(k)}(h_v^{(k)}, h_u^{(k)})}{\sum_{w \in \mathcal{N}(v)} A^{(k)}(h_v^{(k)}, h_w^{(k)})} \quad \text{for all } (v, u) \in E.$$

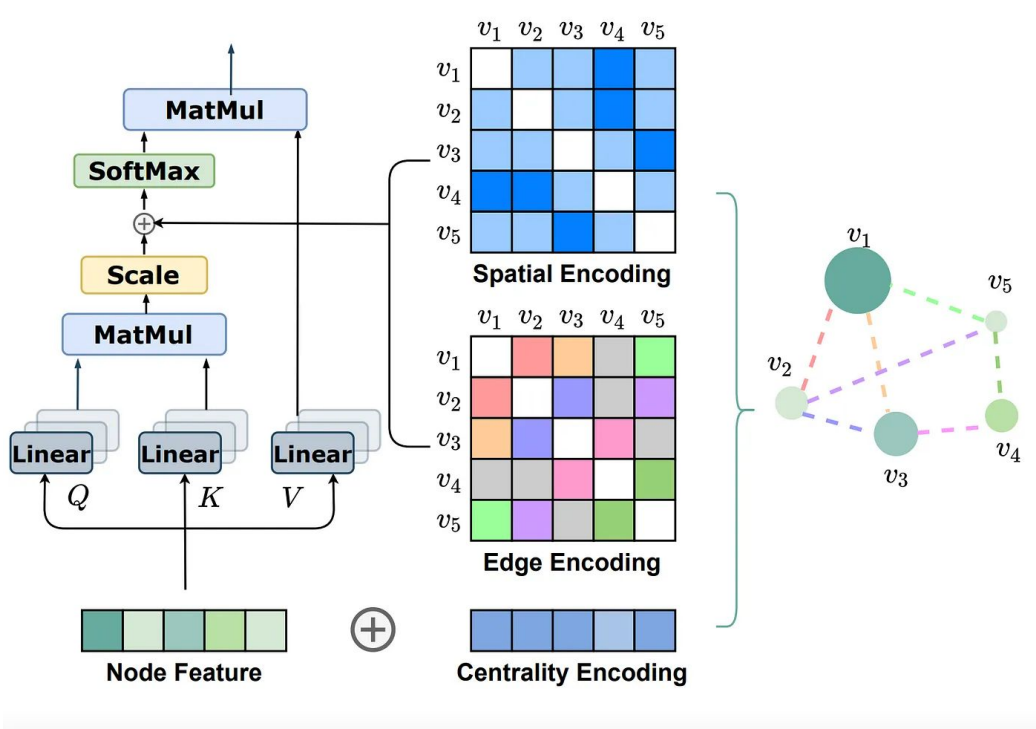
Color Codes:

- Embedding of node  $v$ .
- Embedding of a neighbour of node  $v$ .
- (Potentially) Learnable parameters.

# Multimodal graph attention network for COVID-19 outcome prediction (2023)



# Graphormer (2021)



# Graphormer Usage (2023)



# Graphormer

Graphormer - это пакет глубокого обучения разработанный Microsoft для моделирования молекул.

Разработан для ускорения исследований в области искусственного интеллекта в молекулярной науке, таких как открытие материалов и лекарств.

Поддерживает различные задачи, включая предсказание свойств и молекулярную динамику. Победил в соревнованиях по квантовой химии и молекулярной динамике, подтверждая свою эффективность. В будущем планируется расширение функционала для важных задач, таких как предсказание реакций и генерация молекул.



# Выводы

- 1) графовые данные – особый вид данных, который редко получается эффективно заменить на более удобный для машины
- 2) важно использовать структуру графа в обучении
- 3) графовые методы продолжают улучшаться, потому что есть тенденция к использованию DL подходов в медицине/химии/биологии/физике, где графовых данных очень много