

# Deep Learning on Graphs

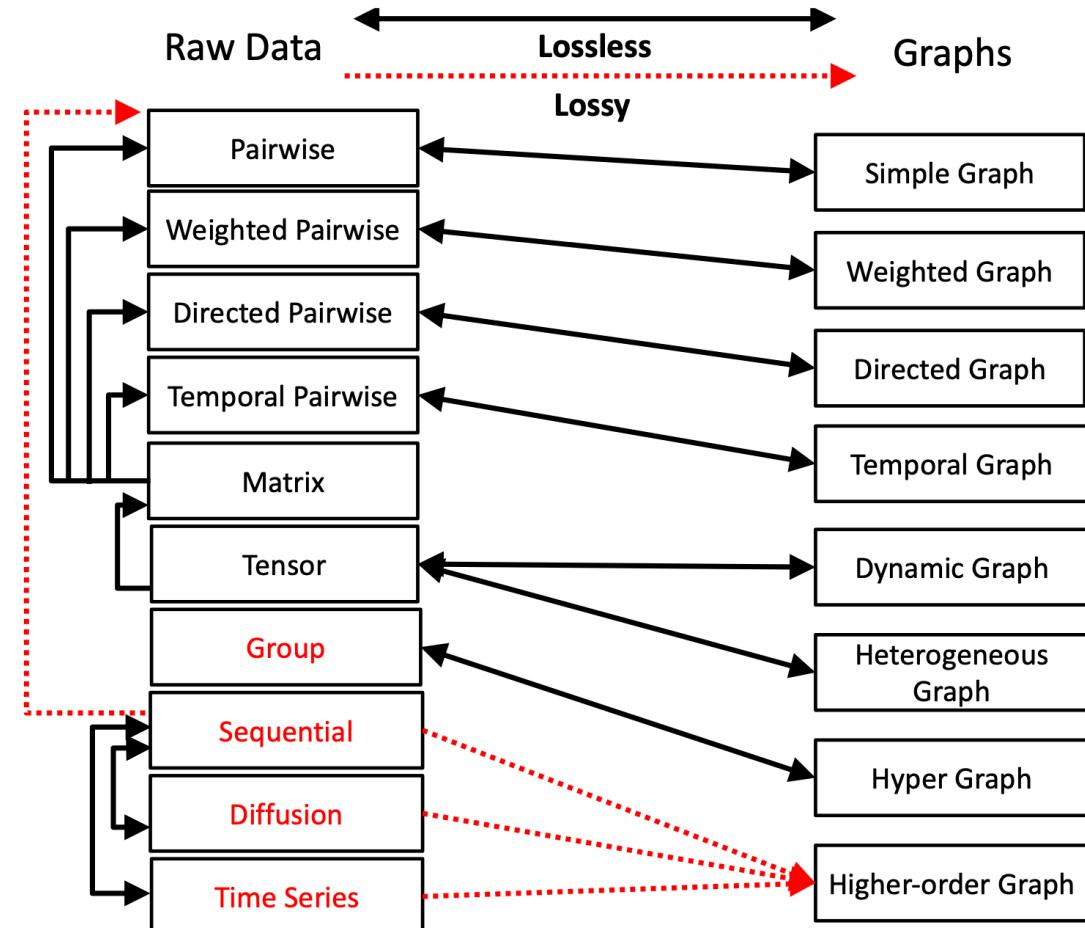
Семерова Елена БПМИ-182

# Почему DL на графах?

1. DL – это круто

2. Графы – это тоже круто:

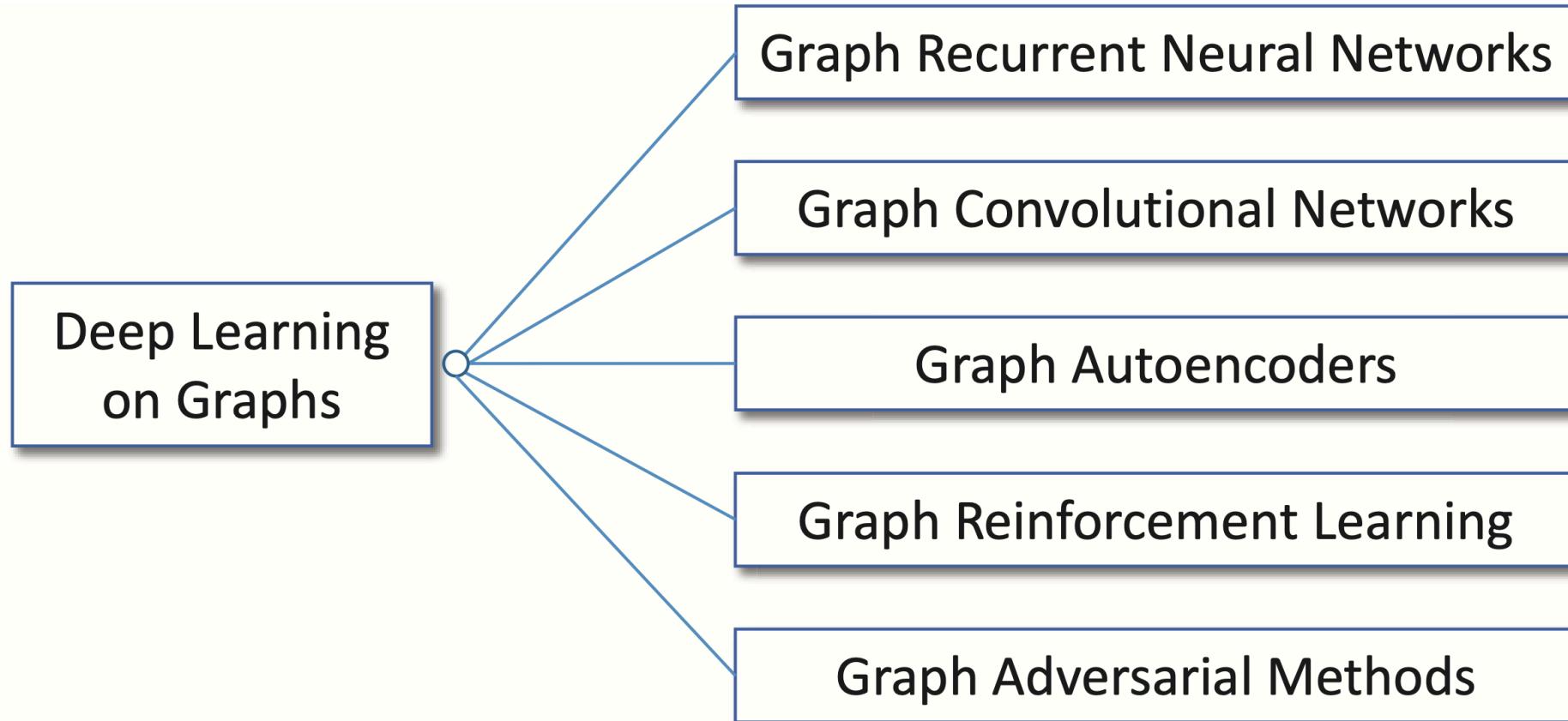
- Они везде встречаются
- Существует множество видов графов, которые могут давать полезную информацию
- Реальные проблемы могут быть интерпретированы как задачи на графах



# Проблемы при работе с графами

1. Непростая структура
2. Неоднородность и разнообразие графов
3. Графы большого масштаба
4. Плохая способность обобщения методов к разным областям

# Виды методов



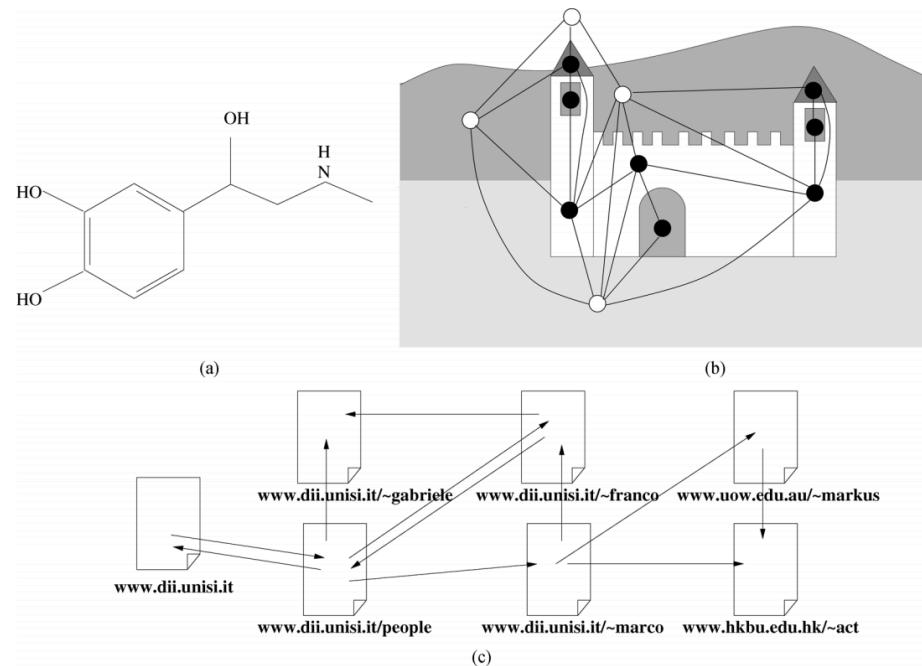
# Graph Neural Networks – основные виды

Общая постановка задачи:

обучение функции  $\tau$ , которая отображает граф  $G$  и узел  $n$  в вектор вещественных чисел:  $\tau(G, n)$

## Graph-focused

Функция  $\tau$  не зависит от узла  $n$ , алгоритм реализуется для набора структурированных данных графа



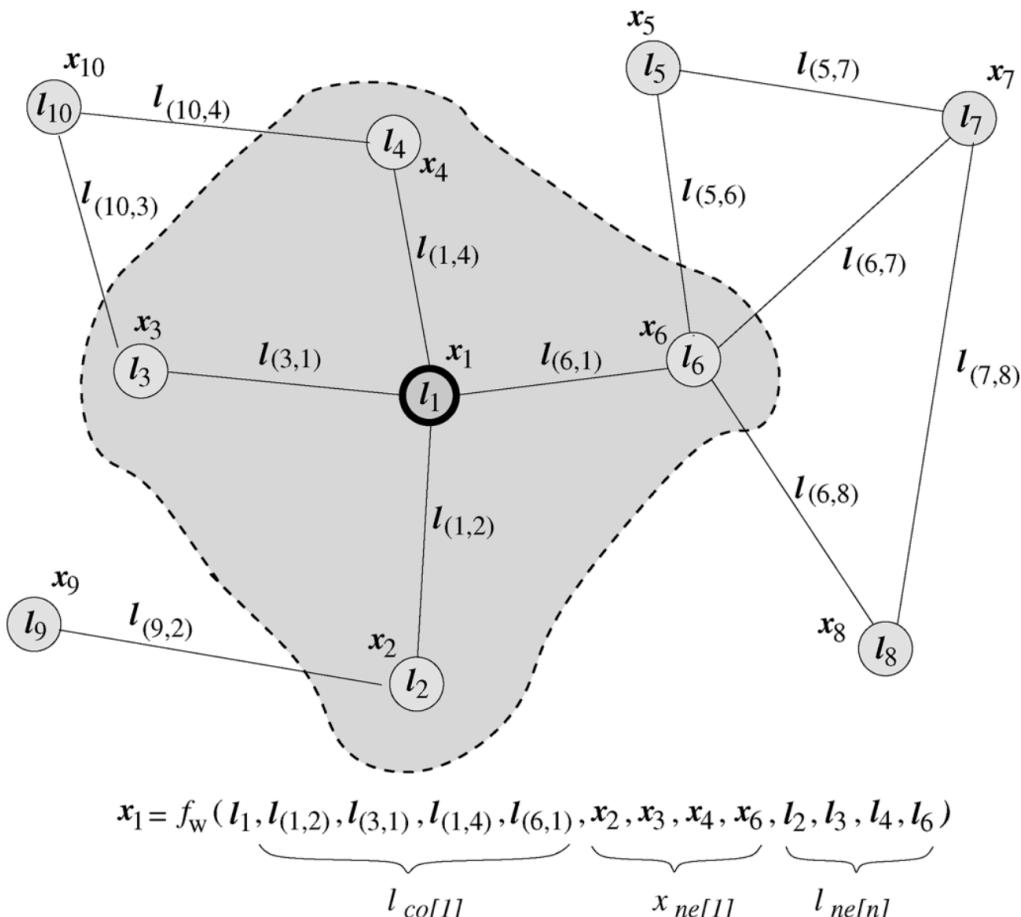
## Node-focused

Функция  $\tau$  зависит от узла  $n$ , алгоритм зависит от свойств уже каждого узла

# Graph RNN (2009)

$\mathbf{l}_n$  – метка узла  $n$ ;  $\mathbf{l}_{\text{co}[n]}$  – метки ребер из узла  $n$ ;  $\mathbf{l}_{\text{ne}[n]}$  – метки узлов в окрестности  $n$  (т.е. соседей)

$\mathbf{x}_{\text{ne}[n]}$  – состояния узлов в окрестности  $n$



Функция  $f_w$  – параметрическая, она называется локальной функцией перехода.

Функция  $g_w$  – локальная функция вывода.

Сам вывод для узла  $n$  будем обозначать так:  $\mathbf{o}_n$

Тогда:

$$\mathbf{x}_n = f_w(\mathbf{l}_n, \mathbf{l}_{\text{co}[n]}, \mathbf{x}_{\text{ne}[n]}, \mathbf{l}_{\text{ne}[n]})$$

$$\mathbf{o}_n = g_w(\mathbf{x}_n, \mathbf{l}_n)$$

# Graph RNN (2009)

Можно представить в виде векторов:

вектор состояний:  $\mathbf{x}$

вектор выходов:  $\mathbf{o}$

вектор меток ребер:  $\mathbf{l}$

вектор меток узлов:  $\mathbf{l}_N$

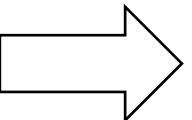
Тогда:

$$\mathbf{x} = F_{\mathbf{w}}(\mathbf{x}, \mathbf{l})$$

$$\mathbf{o} = G_{\mathbf{w}}(\mathbf{x}, \mathbf{l}_N)$$

$F_{\mathbf{w}}$  – глобальная функция перехода

$G_{\mathbf{w}}$  – глобальная функция выхода



$F_{\mathbf{w}}$  – contraction map:

$$\exists \mu, 0 < \mu < 1$$

$$\|F_{\mathbf{w}}(\mathbf{x}, \mathbf{l}) - F_{\mathbf{w}}(\mathbf{y}, \mathbf{l})\| \leq \mu \|\mathbf{x} - \mathbf{y}\|$$

for any  $\mathbf{x}, \mathbf{y}$

# Graph RNN (2009)

Как же вычислять состояния?

Banach's fixed point theorem:

$$\mathbf{x}(t+1) = F_{\mathbf{w}}(\mathbf{x}(t), \mathbf{l})$$

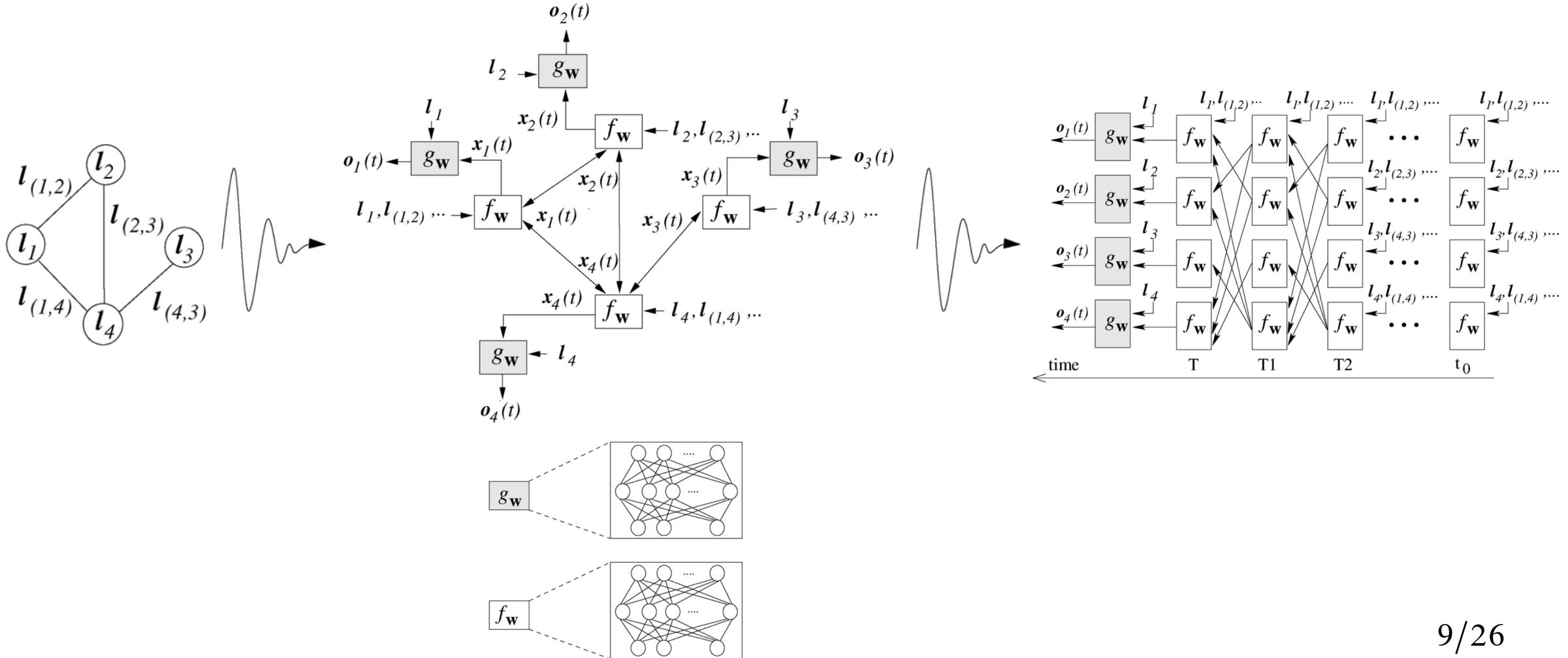
Тогда можем принять  $\mathbf{x}(t)$  за состояние.

Получим для каждого узла  $n$ :

$$\begin{aligned}\mathbf{x}_n(t+1) &= f_{\mathbf{w}}(\mathbf{l}_n, \mathbf{l}_{\text{co}[n]}, \mathbf{x}_{\text{ne}[n]}(t), \mathbf{l}_{\text{ne}[n]}) \\ \mathbf{o}_n(t) &= g_{\mathbf{w}}(\mathbf{x}_n(t), \mathbf{l}_n), \quad n \in N.\end{aligned}$$

# Graph RNN (2009)

Как же вычислять состояния?



# Graph RNN (2009)

Немного про обучение  $f_{\mathbf{w}}$  и  $g_{\mathbf{w}}$

Функция  $\varphi_{\mathbf{w}}$

Ее параметры  $\mathbf{w}$

Данные обучения  $\mathcal{L} = \{(\mathbf{G}_i, n_{i,j}, \mathbf{t}_{i,j}) |, \mathbf{G}_i = (\mathbf{N}_i, \mathbf{E}_i) \in \mathcal{G};$   
 $n_{i,j} \in \mathbf{N}_i; \mathbf{t}_{i,j} \in \mathbb{R}^m, 1 \leq i \leq p, 1 \leq j \leq q_i\}$

Минимизируем:

$$e_{\mathbf{w}} = \sum_{i=1}^p \sum_{j=1}^{q_i} (\mathbf{t}_{i,j} - \varphi_{\mathbf{w}}(\mathbf{G}_i, n_{i,j}))^2$$

# Graph RNN (2009)

Немного про обучение  $f_{\mathbf{w}}$  и  $g_{\mathbf{w}}$

Алгоритм обучения:

1. Итеративное обновление состояний  $\mathbf{x}_n(t)$  до приближения к фиксированной точке в момент  $T$ :  $\mathbf{x}(T) \approx \mathbf{x}$
2. Вычисление градиента  $\partial e_{\mathbf{w}}(T) / \partial \mathbf{w}$
3. Обновление весов  $\mathbf{w}$  в соответствии с градиентом из 2-го шага

# Graph RNN (2009)

Немного про обучение  $f_{\mathbf{w}}$  и  $g_{\mathbf{w}}$

*Theorem 1 (Differentiability):* Let  $F_{\mathbf{w}}$  and  $G_{\mathbf{w}}$  be the global transition and the global output functions of a GNN, respectively. If  $F_{\mathbf{w}}(\mathbf{x}, \mathbf{l})$  and  $G_{\mathbf{w}}(\mathbf{x}, \mathbf{l}_N)$  are continuously differentiable w.r.t.  $\mathbf{x}$  and  $\mathbf{w}$ , then  $\varphi_{\mathbf{w}}$  is continuously differentiable w.r.t.  $\mathbf{w}$ .

*Theorem 2 (Backpropagation):* Let  $F_{\mathbf{w}}$  and  $G_{\mathbf{w}}$  be the transition and the output functions of a GNN, respectively, and assume that  $F_{\mathbf{w}}(\mathbf{x}, \mathbf{l})$  and  $G_{\mathbf{w}}(\mathbf{x}, \mathbf{l}_N)$  are continuously differentiable w.r.t.  $\mathbf{x}$  and  $\mathbf{w}$ . Let  $\mathbf{z}(t)$  be defined by

$$\mathbf{z}(t) = \mathbf{z}(t+1) \cdot \frac{\partial F_{\mathbf{w}}}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{l}) + \frac{\partial e_{\mathbf{w}}}{\partial \mathbf{o}} \cdot \frac{\partial G_{\mathbf{w}}}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{l}_N). \quad (7)$$

Then, the sequence  $\mathbf{z}(T), \mathbf{z}(T-1), \dots$  converges to a vector  $\mathbf{z} = \lim_{t \rightarrow -\infty} \mathbf{z}(t)$  and the convergence is exponential and independent of the initial state  $\mathbf{z}(T)$ . Moreover

$$\frac{\partial e_{\mathbf{w}}}{\partial \mathbf{w}} = \frac{\partial e_{\mathbf{w}}}{\partial \mathbf{o}} \cdot \frac{\partial G_{\mathbf{w}}}{\partial \mathbf{w}}(\mathbf{x}, \mathbf{l}_N) + \mathbf{z} \cdot \frac{\partial F_{\mathbf{w}}}{\partial \mathbf{w}}(\mathbf{x}, \mathbf{l}) \quad (8)$$

holds, where  $\mathbf{x}$  is the stable state of the GNN.

# Недостатки метода

1. Ограничение на возможности моделирования из-за того, что глобальная функция перехода должна быть contraction map
2. Использование одних и тех же параметров в итерациях
3. Требуются большие вычислительные ресурсы
4. Нет адаптации для динамических графов

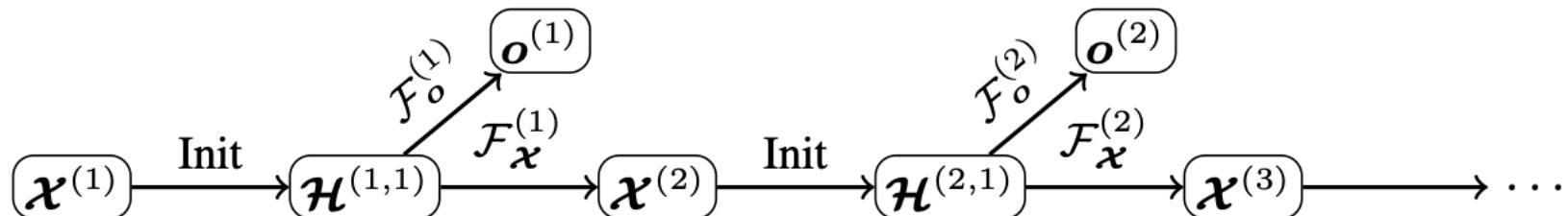
# Различные модификации

Рассмотрим следующие алгоритмы:

1. Gated Graph Sequence Neural Network
2. Stochastic Steak-steady Embedding
3. Dynamic Graph Neural Network
4. Variational Graph Recurrent Neural Network
5. Stochastic Graph Recurrent Neural Network

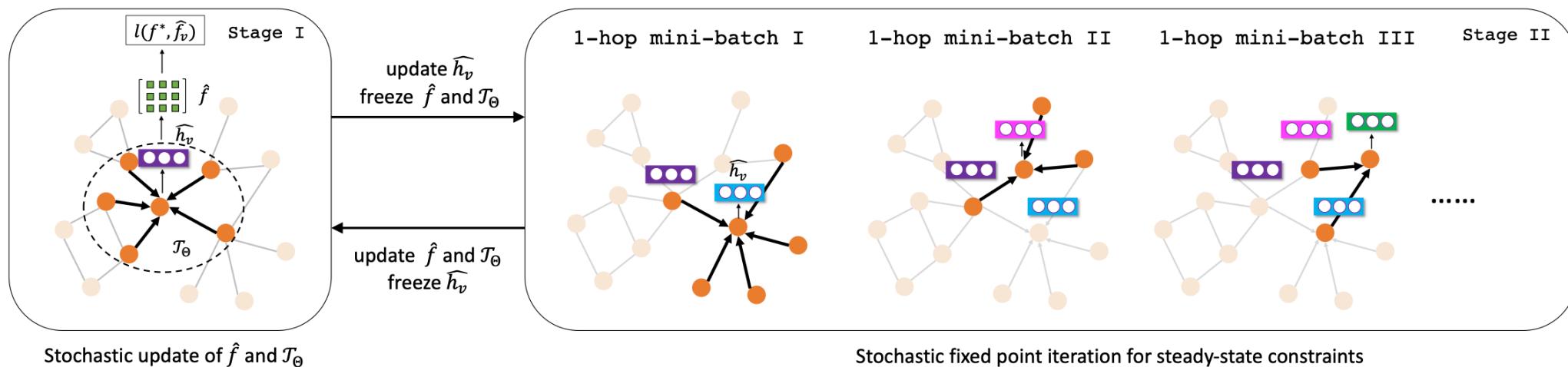
# GGS-NN (2017)

1. Используются GRU
2. Разворачивают повторение для фиксированного числа шагов  $T$
3. Используют BackPropagation при вычислении градиентов
4. Расширяют базовые представления и модель выводы
5. Предлагают нейронную сеть, где несколько GGS-NN работают последовательно для создания выходной последовательности



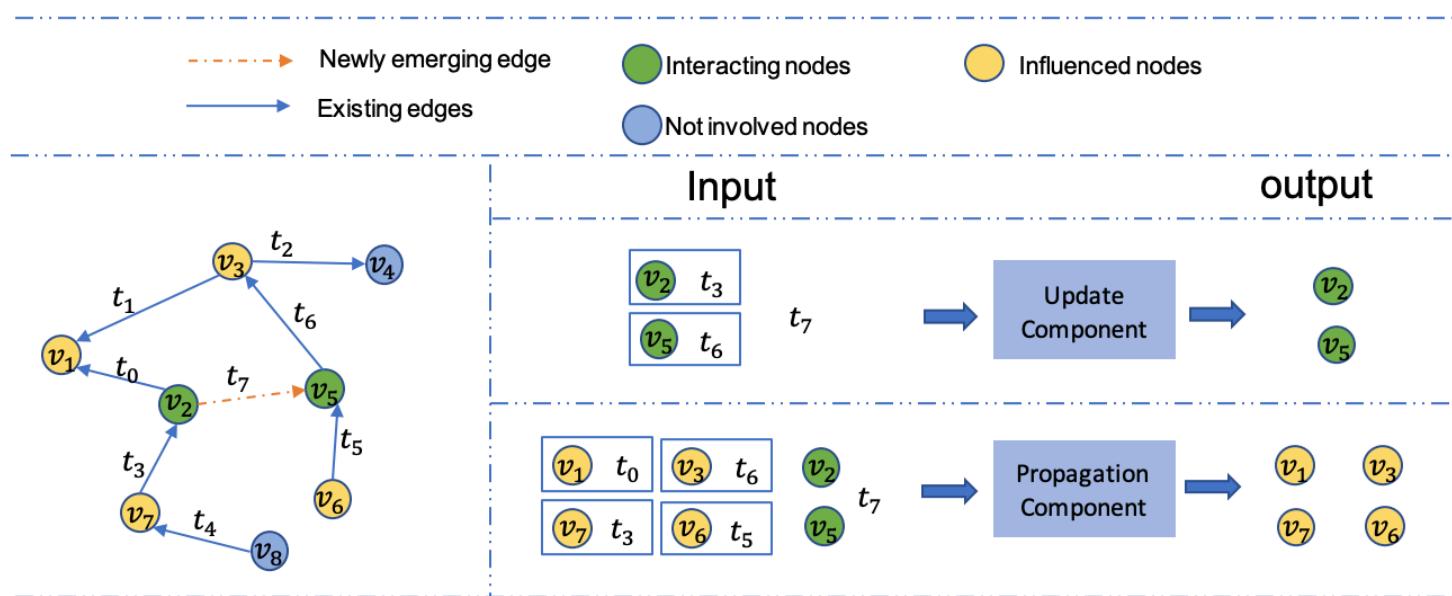
# SSE (2018)

1. Концентрация на соблюдении стабильности и быстроте обучения
2. Используется стохастический градиентный спуск с фиксированной точкой
3. Использование mini-batches



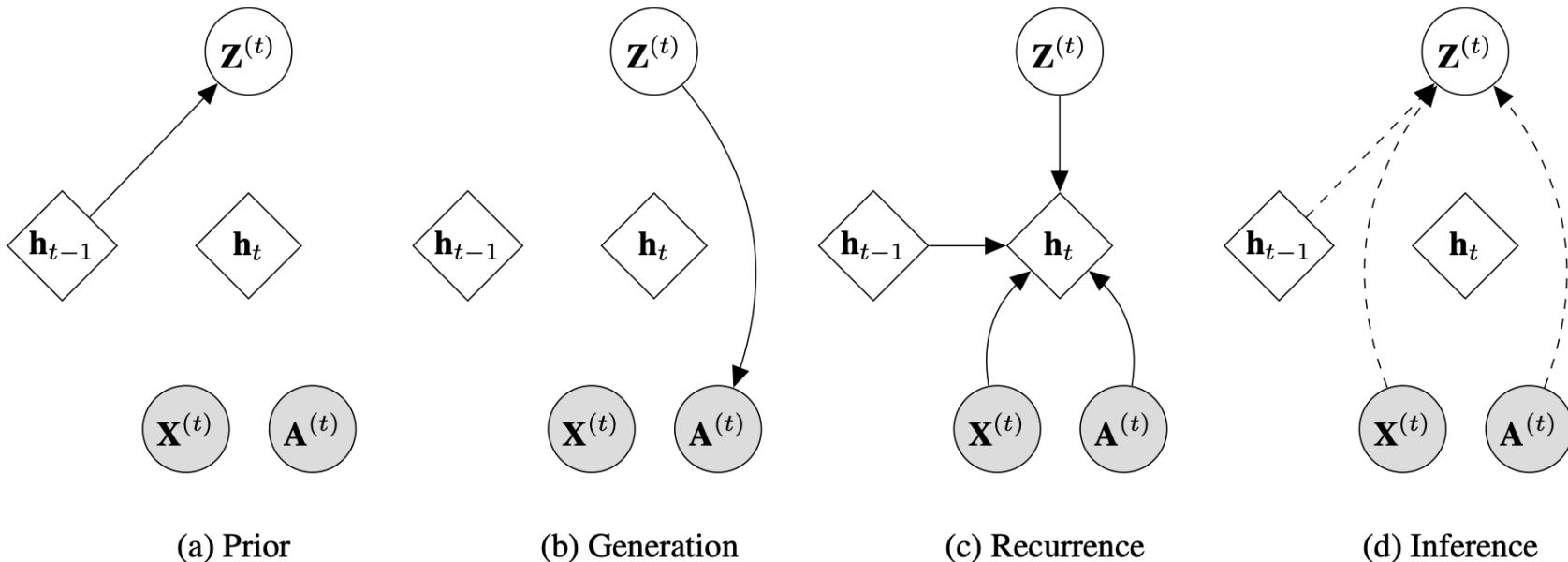
# DGNN (2018)

1. Алгоритм для динамических графов
2. Рассматриваются временные взаимодействия по добавлению ребер или узлов
3. Две основных компоненты: 1) компонента обновления и 2) компонента распространения.



# VGRNN (2019)

1. Интеграция GCN и RNN в структуре GRNN
2. Комбинирование GRNN и VGAE



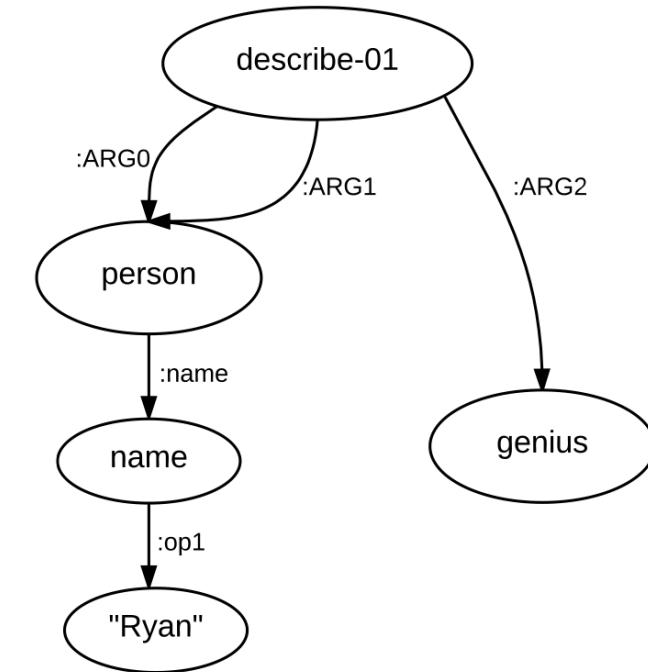
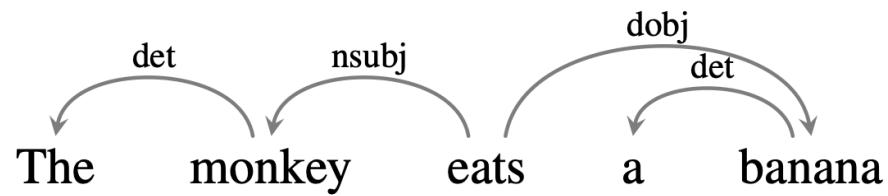
# SGRNN (2020)

1. Объединение GRNN и VAE
2. Отделение детерминированных состояний от стохастических в итерационном процессе
3. Сеть вывода основана на нижней границе KL-дивергенции

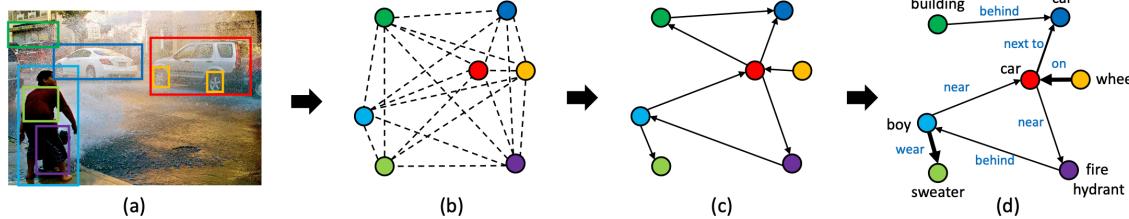
# Области применения GNN-s

1. Social Networks Analyses
2. Traffic Analyses
3. NLP
4. Computer Vision
5. Data Mining
6. Biochemistry and Healthcare
7. Другое

# Применения в NLP



# Применение в Computer Vision

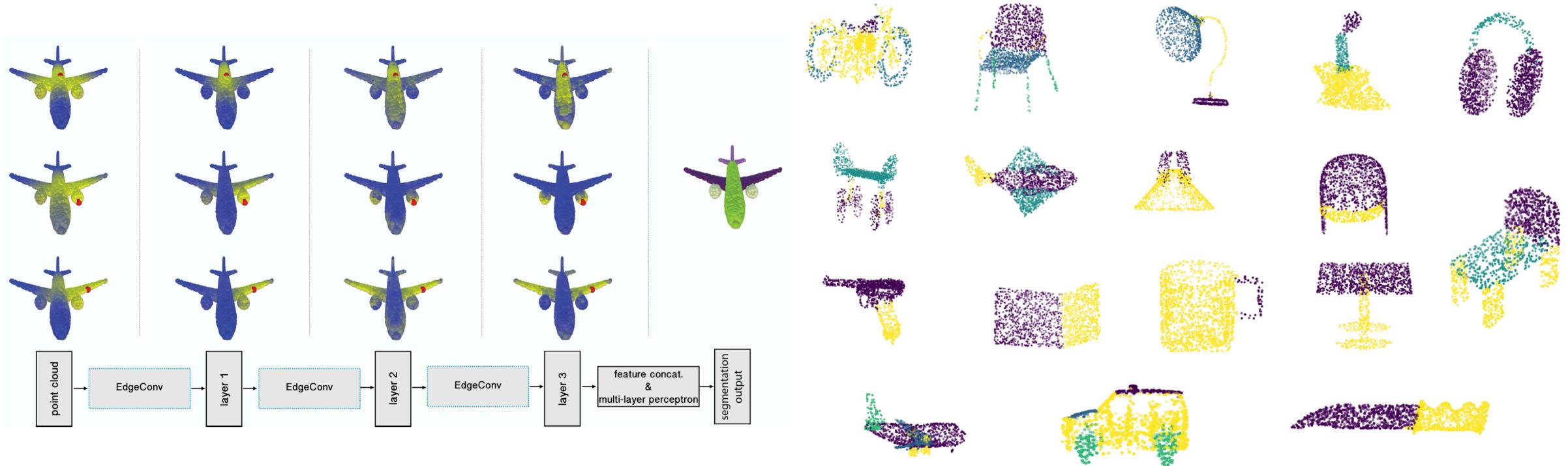


Переход от изображения к семантическому графу, который состоит из объектов и их семантических отношений



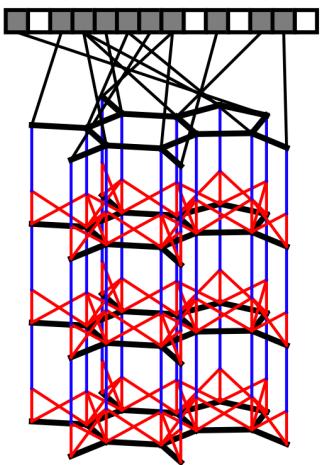
Генерация реалистичных изображений по графикам сцены

# Применение в Computer Vision

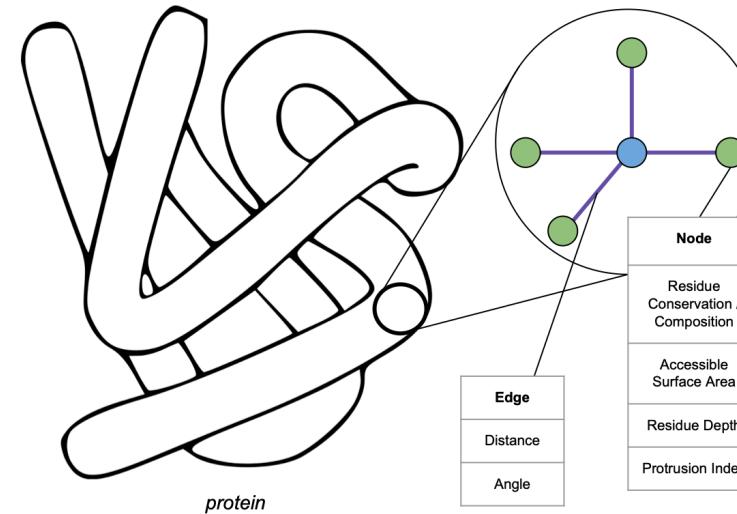
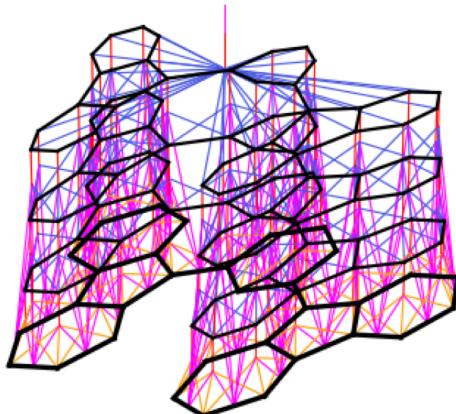


Сегментация

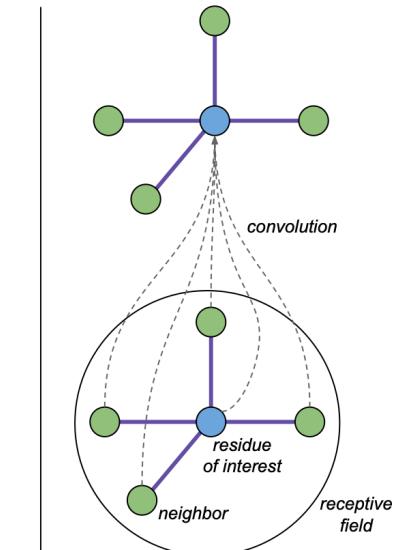
# Применения в биологии, химии, медицине



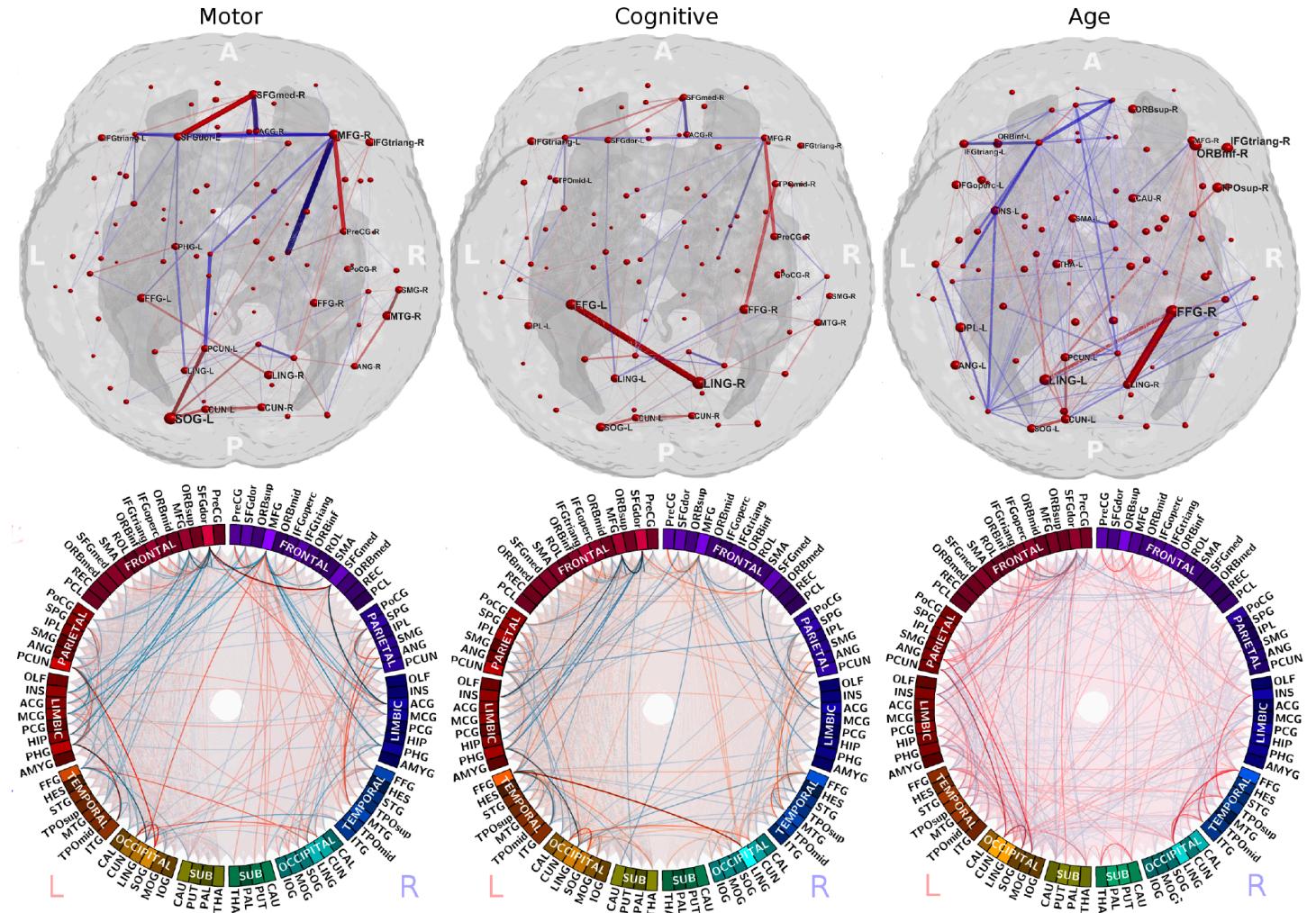
Представления отпечатков пальца



Представление протеина



# Применения в биологии, химии, медицине



# Brain networks

# Использованные источники

- Общие обзоры:
  1. Deep Learning on Graphs: A Survey  
<https://arxiv.org/pdf/1812.04202.pdf>
  2. Comprehensive Survey on Graph Neural Networks  
<https://arxiv.org/pdf/1901.00596.pdf>
  3. Deep Learning on Graphs  
[https://cse.msu.edu/~mayao4/dlg\\_book/](https://cse.msu.edu/~mayao4/dlg_book/)
- Методы:
  1. The Graph Neural Network Model  
<https://persagen.com/files/misc/scarselli2009graph.pdf>
  2. Gated Graph Sequence Neural Networks  
<https://arxiv.org/pdf/1511.05493.pdf>
  3. Learning Steady-States of Iterative Algorithms over Graphs  
<http://proceedings.mlr.press/v80/dai18a/dai18a.pdf>
  4. Streaming Graph Neural Networks  
<https://arxiv.org/pdf/1810.10627.pdf>
  5. Variational Graph Recurrent Neural Networks  
<https://arxiv.org/pdf/1908.09710.pdf>
  6. Stochastic Graph Recurrent Neural Network  
<https://arxiv.org/pdf/2009.00538.pdf>
- Применения:
  1. NLP  
<https://arxiv.org/pdf/1704.04675.pdf>  
<https://arxiv.org/pdf/1805.02473.pdf>
  2. CV  
<https://arxiv.org/pdf/1808.00191.pdf>  
<https://arxiv.org/pdf/1804.01622.pdf>  
<https://arxiv.org/pdf/1801.07829.pdf>  
<https://arxiv.org/pdf/1806.02952.pdf>
  3. Biology/chemistry/brain networks  
<https://arxiv.org/pdf/1509.09292.pdf>  
<https://proceedings.neurips.cc/paper/2017/file/f507783927f2ec2737ba40afbd17efb5-Paper.pdf>  
[https://www.researchgate.net/publication/308308533\\_BrainNetCNN\\_Convolutional\\_Neural\\_Networks\\_for\\_Brain\\_Networks\\_Towards\\_Predicting\\_Neurodevelopment](https://www.researchgate.net/publication/308308533_BrainNetCNN_Convolutional_Neural_Networks_for_Brain_Networks_Towards_Predicting_Neurodevelopment)