

**іТМО**

**Библиотека автоматического  
обучения объяснимых  
графовых нейронных сетей**

Андреева Полина  
polinaspb@ya.ru

# Графовые нейронные сети

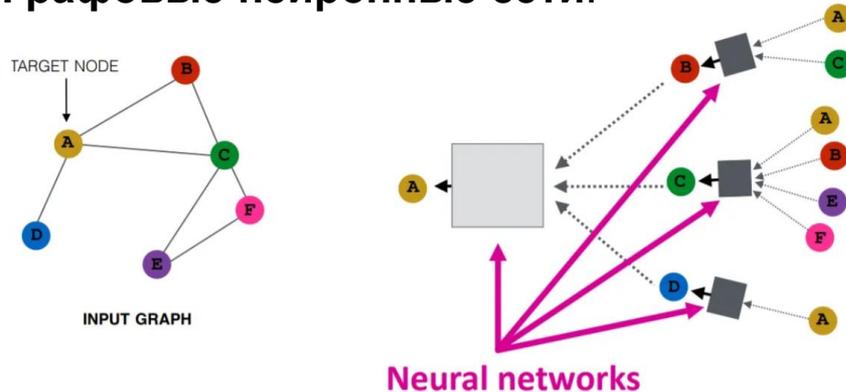
## Графами можно моделировать:

- химические молекулы;
- транспортные системы;
- социальные сети;
- сети банковских транзакций;
- ...

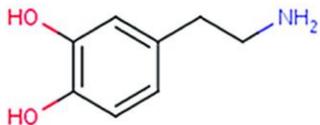
## Задачи ГНС:

- Прогнозирование растворимости молекул (задача регрессии/классификации графов);
- Предсказание характеристики пользователей в соц. сети (задача регрессии/классификации вершин);
- Рекомендательная система – предсказать купит ли человек определенный товар на основе информации о покупках интересующего человека и других людей (задача предсказания связей);

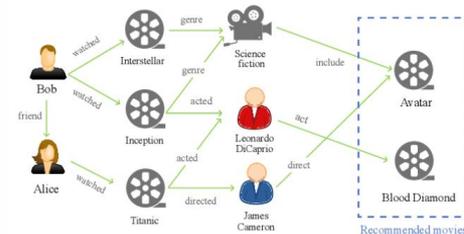
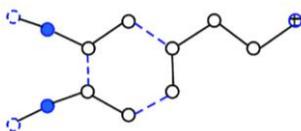
## Графовые нейронные сети:



Dopamine



Molecular graph



# Фреймворки для графовых нейронных сетей **ITMO**

```
# Загрузка данных
dataset = Planetoid(root='/tmp/Coza', name='Coza')
data = dataset[0]

# Подготовка данных
train_mask = data.train_mask
test_mask = data.test_mask

# Определение архитектуры сети
class Net(torch.nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = GCNConv(dataset.num_node_features, 16)
        self.conv2 = GCNConv(16, dataset.num_classes)

    def forward(self, data):
        x, edge_index = data.x, data.edge_index
        x = F.relu(self.conv1(x, edge_index))
        x = F.dropout(x, training=self.training)
        x = self.conv2(x, edge_index)
        return F.log_softmax(x, dim=1)

# Инициализация сети и оптимизатора
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model = Net().to(device)
data = data.to(device)
optimizer = optim.Adam(model.parameters(), lr=0.01, weight_decay=5e-4)

# Обучение сети
model.train()
for epoch in range(200):
    optimizer.zero_grad()
    out = model(data)
    loss = F.nll_loss(out[train_mask], data.y[train_mask])
    loss.backward()
    optimizer.step()

# Тестирование сети
model.eval()
_, pred = model(data).max(dim=1)
correct = pred[test_mask].eq(data.y[test_mask]).sum().item()
accuracy = correct / test_mask.sum().item()

print("Accuracy: {:.4f}".format(accuracy))
```

StableGNN (automation)

```
class Graph()
```

```
class TrainModelNC()
```

```
class Explain()
```



backend

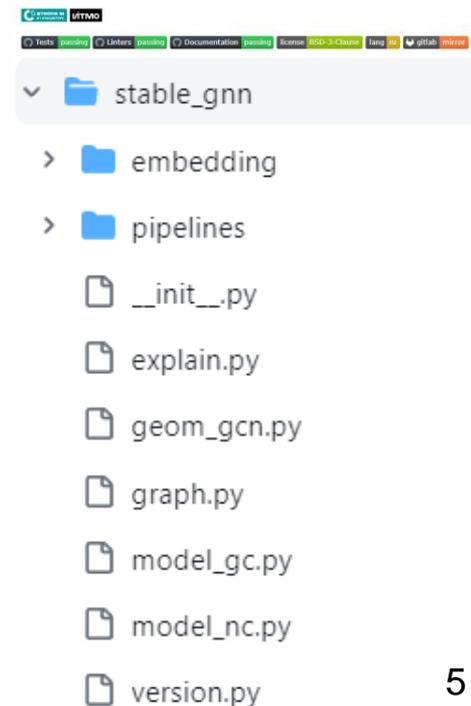
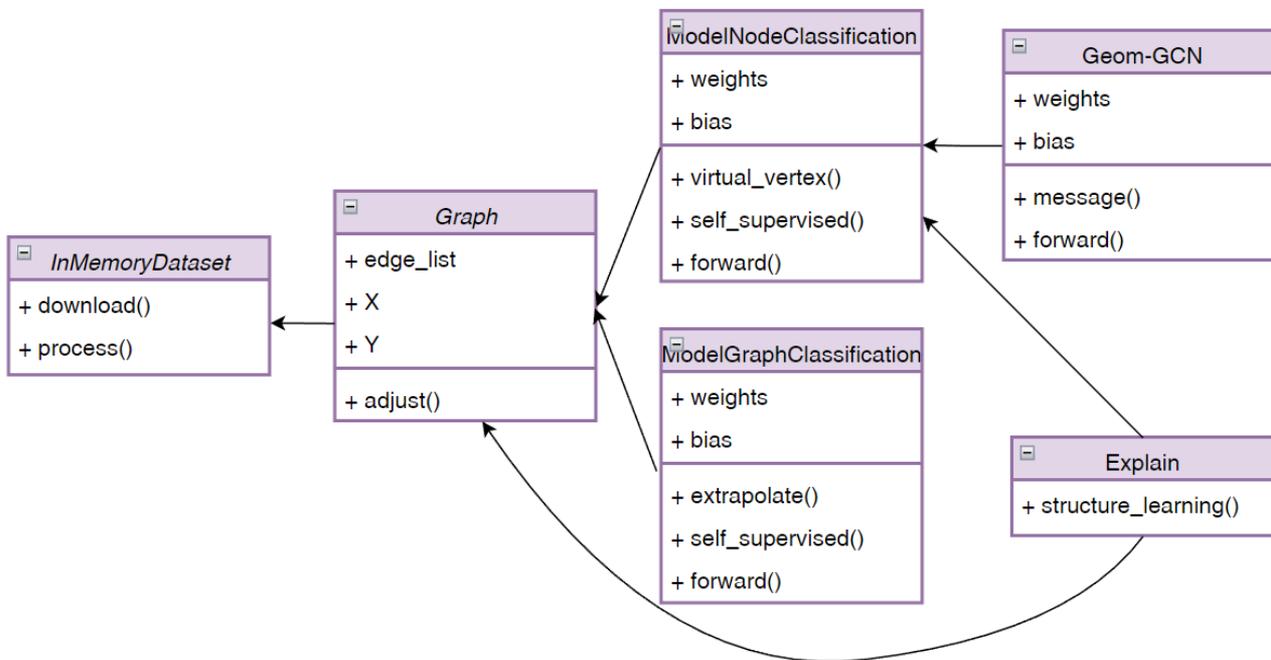


*С какими проблемами мы можем столкнуться, если хотим делать «универсальный пайплайн»?*

- Шумные данные => уточнение структуры графа (с учетом априорных свойств графа);
- Малый объем размеченных данных => self-supervised функции потерь;
- Низкая интерпретация результатов => объяснять решения;
- Ограниченные / неизвестные распределения данных => Обобщаемость / возможность экстраполяции;
- Графы с низкой ассортативностью => метод GeomGCN.

# Библиотека

<https://github.com/aimclub/StableGNN>

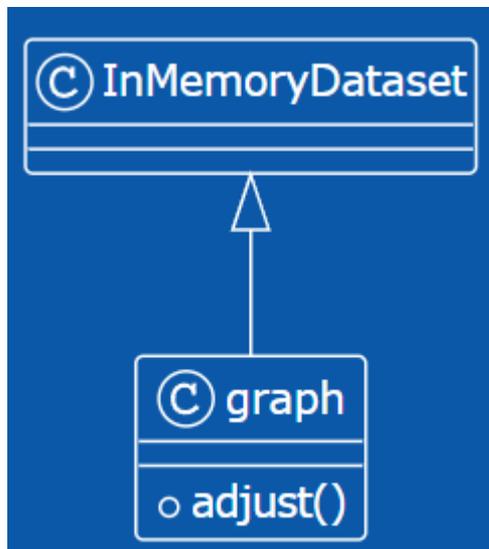


# Основные компоненты StableGNN

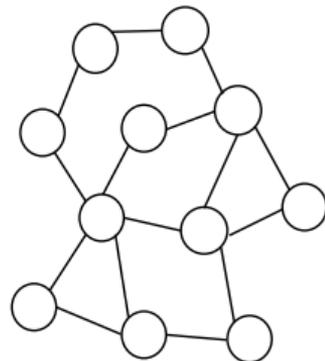
- Модуль `graph`: уточнение графа с учетом свойства ассортативности
- Модуль `model_nc`: self-supervised (предсказание степени вершины), свертка `geom_gcn` (нет ее реализации в `torch_geometric`)
- Модуль `model_gc`: экстраполяция, self-supervised (предсказание степени вершины)
- Модуль `explain`: PGM-Explainer (сейчас только для `model_nc`)

# ОСНОВНЫЕ КОМПОНЕНТЫ: StableGNN.graph

*Уточнение структуры графа*

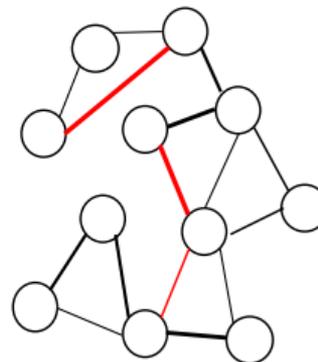


Original graph structure  $A$



Structure modeling

Refined graph structure  $A^*$

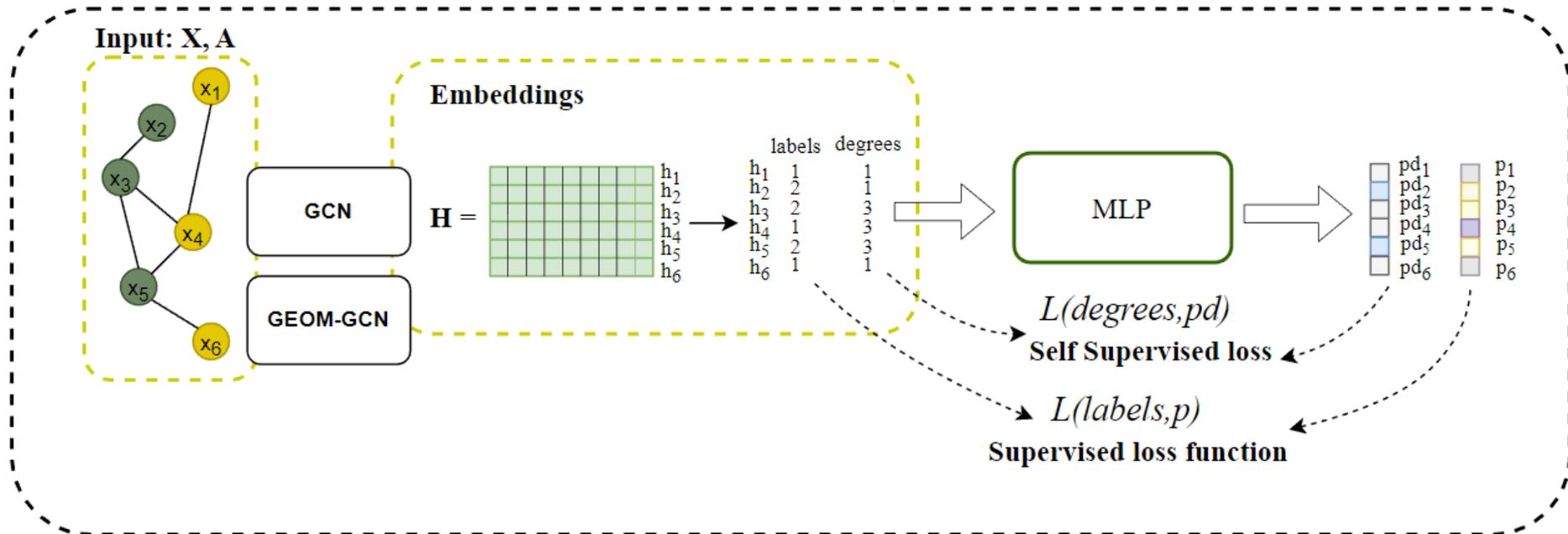


Graph neural networks

Learning parameters of the refined graph structure

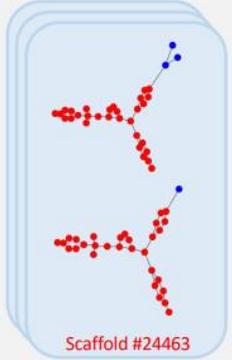
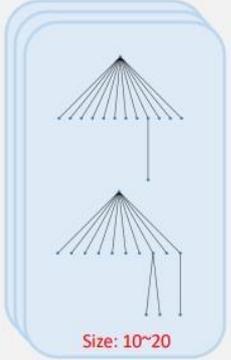
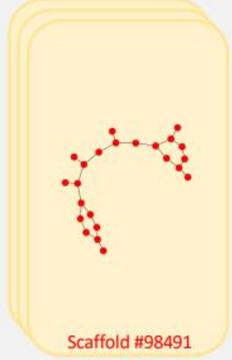
# Основные компоненты: StableGNN.model\_nc

*geom\_gcn, loss\_self\_supervised*



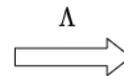
# ОСНОВНЫЕ КОМПОНЕНТЫ: StableGNN.model\_gc **ITMO**

*Extrapolation*

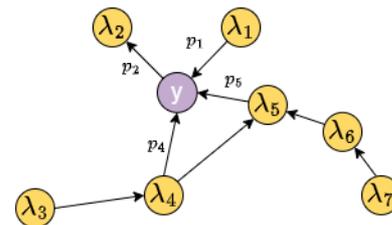
	MolPCBA	GossipCop
Train Set	 <p>Scaffold #24463</p>	 <p>Size: 10~20</p>
Test Set	 <p>Scaffold #98491</p>	 <p>Size: 90~110</p>

Спектральное разложение  
матриц смежности  
каждого графа

$$A_i = U\Lambda_i U^T$$



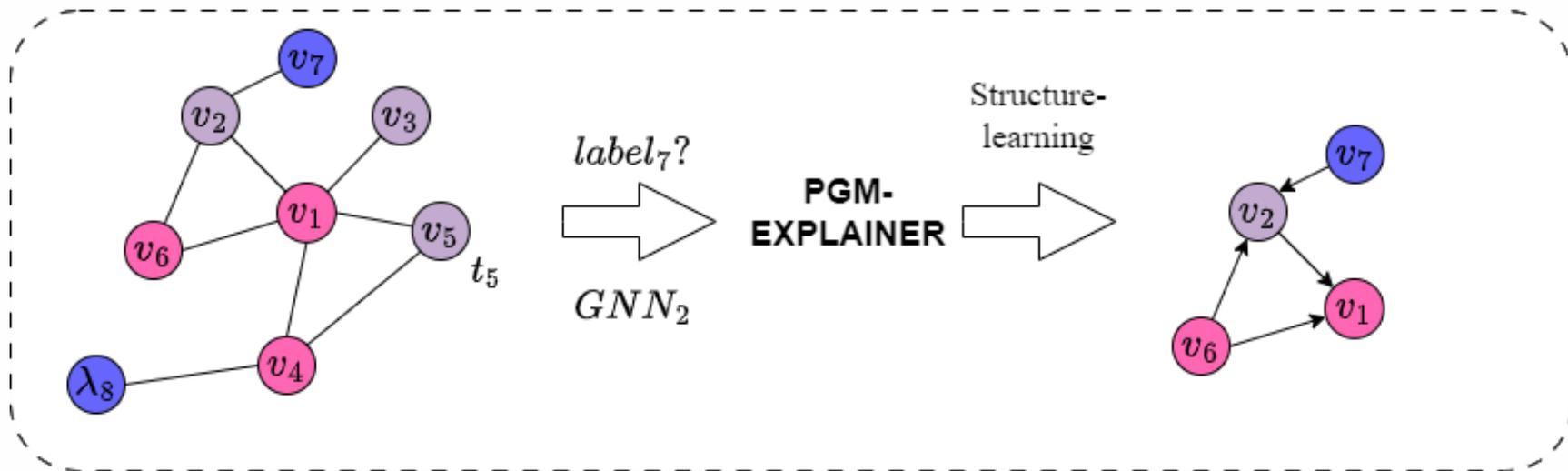
Структурное обучение  
байесовской сети с весами



# Основные компоненты: StableGNN.explain

какой подграф наиболее повлиял на предсказание?

Node-level



# Примеры применения StableGNN

Предсказания валовой прибыли нефтяных месторождений

	adjust_flag = True	adjust_flag = False
ssl_flag = True	<b>0.47</b>	0.42
ssl_flag = False	0.37	0.40

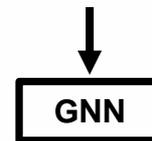
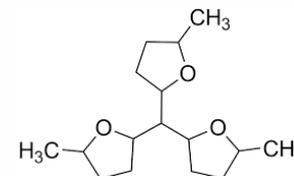
Предсказание свойств горения углеводорода

	F1, extrapolate_flag = True	F1, extrapolate_flag = False
ssl_flag = True	<b>0.8</b>	0.73
ssl_flag = False	0.66	0.74

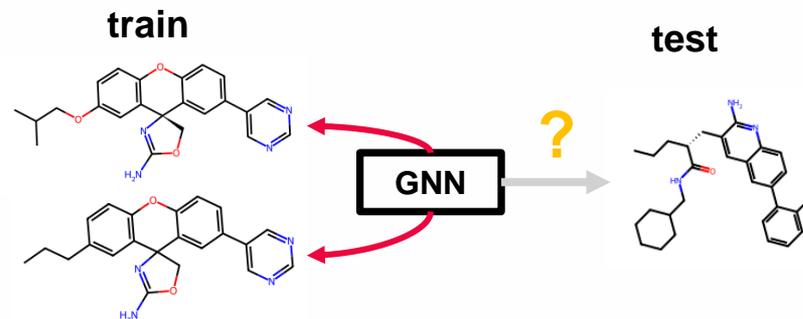
Предсказание качественного связывания набора ингибиторов  $\beta$ -секретазы человека.

**Экстраполяция** GNN на различные типы молекул (scaffold) на примере датасета BACE.

	ROC AUC
UniMol	85.7
<b>StableGNN</b>	<b>90.1</b>



цетановое число  
октановое число



# Возможные направления НИР и ВКР

Исследование и разработка методов объяснения предсказаний графовых нейронных сетей. (`stable_gnn.explain`)

Исследование и разработка методов уточнения структуры графов для обучения графовой нейронной сети (`stable_gnn.graph`)

Исследование и разработка функций потерь самостоятельного обучения графовых нейронных сетей. (`stable_gnn.model_nc/stable_gnn.model_gc`)

Ваши предложения!

## Вопросы и дискуссия!



**THANK YOU  
FOR YOUR TIME!**

**iTMO** *re than a*  
**UNIVERSITY**

Your contact info