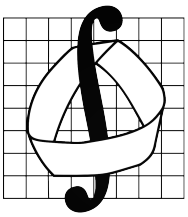
Работа защищена с оценкой ___

Ученый секретарь: доцент В. Д. Валединский

Московский государственный университет имени М.В. Ломоносова
Механико-математический факультет
Кафедра вычислительной математики



Курсовая работа

Сравнение методов решения задачи структура-свойство на основе графовых нейронных сетей

Comparison of methods for solving the structure-property problem based on graph neural networks

Ермолаев Никита Дмитриевич, группа 510
Научный руководитель: д. ф.-м н. Кумсков Михаил Иванович

2022
Москва

Аннотация

В данной работе рассматриваются различные варианты прогнозирования свойств биологический соединений. Помимо описания классических методов, таких как ComFA и моделей на топологических дескрипторах, в работе рассмотрены нейросетевые подходы способные работать с неевклидовыми структурами данных. Эти методы могут быть удобными при работе со сложноструктурированными геометрическими объектами и показывать state-of-the-art результаты на большом классе задач. В работе представлена общая схема построение графовых нейронных сетей (GNN), а также рассмотрены несколько конкретных архитектур. Рассмотрен общий фреймворк построения графовых нейронных сетей - Massage Passing Neural Network (MPNN), который обобщает широкий класс архитектур. Исследуются способы построения и обучения моделей GNN для задачи прогнозирования молекулярных свойств. В практической части проведено сравнения моделей на основе графовых нейронных сетей с классическими моделями, обученными на топологических дескрипторах молекулярных графах. Проведенное сравнение результатов вычислительных экспериментов свидетельствует о перспективности использования моделей на основе GNN.

Оглавление

Введение

В работе рассматривается одна из важных задач биоинформатики - прогнозирования свойств молекулярных соединений. В настоящее время перед специалистами из расзличных областей, таких как: химия, биология и биоинженерия - стоит задача определения различных важных свойств молекулярных соединений. определение подобных свойств в рамках лабораторных экспериментов требует значительных временных и денежных ресурсов. Поэтому использование компьютерных технологий, в том числе алгоритмов машинного обучения, может позволить существенное упростить решение данной задачи.

Одним из вариантов представления химического соединения является малекулярный граф. С молекулярным графом связаны различные физико–химические и биологические свойства, которыми обладает молекула. Поэтому представить сложный структурный объект в виде векторного описания, которое требуется для многих классических алгоритмов машинного обучения - не тривиальная задача. Именно поэтому в последнее время набирают популярность методы основанные на глубоком обучении. Преимуществом таких подходов является то, что модель способна обучать векторные представления различных объектов, например молекул, на основе конкретной задачи. Было предложено много нейросетевых архитектур способных обрабатывать сложные геометрические объекты, в том числе и графы. При использовании таких архитектур, итоговое векторное представление объектов содержит общую топологическую информацию, которая теряется при рассмотрения молекулярного графа как множества независимых дескрипторов в классических подходах решения данной задачи.

Целью данной работы является сравнения методов прогнозирования свойств молекулярных соединений на основе графовых неронных сетей с классическими методами на основе топологических дескрипторов.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Исследовать подходы к задаче прогнозирования свойств молекулярных соединений
2. Исследовать архитектры графовых нейронных сетей

3. Исследовать варианты применения архитектур GNN к задаче прогнозирования химических свойств молекулрных соединений
4. Обучить графовые нейронный модели на множестве молекулрных выборок и провести сравнительный анализ с классическими метода

Актуальность работы подтверждается растущем числом задач прогнозирование свойств молекулярных соединений в самых различных областях. Число доступных данных каждый год многократно увеличивается, в том числе размеры молекулярных выборов увеличиваются на порядки. Поэтому без использования компьютерных технологий обработка такого количества соединений не представляется возможным. Существующие исследования на тему сравнения этих двух подходов не дают четкого ответа. Результаты из разных статей бывают противоречивыми. Периодически оба подхода выдают новые эталонные результаты.

Глава 1. Прогнозирование биологических свойств химических соединений. Методы и подходы.

Модели которые позволяют по структурам химических соединений прогнозировать их разнообразные свойства называют QSAR [1, 2, 3, 4, 5, 6, 7, 8]. QSAR моделирование основано на математической статистики и машинном обучении. При прогнозировании свойств на качественном уровне (например, будет ли данное химическое соединение обладать данным видом биологической активности) говорят о решении классификационной задачи, тогда как при прогнозировании числовых значений свойств говорят о решении регрессионной задачи. Описание структур химических соединений для этих целей может быть векторным либо невекторным (графовым). QSAR модели позволяют прогнозировать биологическую активность новых химических соединений на основе закономерностей (касающихся структурных, физико-химических и конформационных свойств молекул), выявленных из предыдущих опытов [9].

Есть несколько вариантов построения QSAR - моделей. Они существенно зависят от вариантов представления молекулярных соединений.

Как видно на рисунке 1.1 варианты представления могут сильно разнится по структуре и средствам представления.

a) Двумерное (2D) изображение (структура Кекуле).
b) Молекулярный граф (2D), состоящий из вершин (атомов) и ребра (связи).
c) Строка SMILES [ссылка на smiles], в которой тип атома, тип связи задаются буквенно-цифровыми символами.
d) Трехмерный (3D) граф, состоящий из вершин (атомы), их положение (координаты x, y, z) в трехмерном пространстве, и ребра (связи).
e) Молекулярная поверхность представлена в виде сетки, окрашенной в соответствии с типам атомов.

Все способы представить молекулярные соединения не тривиальны с точки зрения применения к ним математических методов прогнозирования. Поэтому одним из способов решения QSAR задачи является построение векторного описания соединения. Данное описание строится из молекулярных дескрипторов [8, 10, 11] инвариантов молекулярного графа.

Молекулярные дескрипторы могут извлекаться из трехмерной модели структуры, то есть на данные о пространственном расположении атомов. Дру-

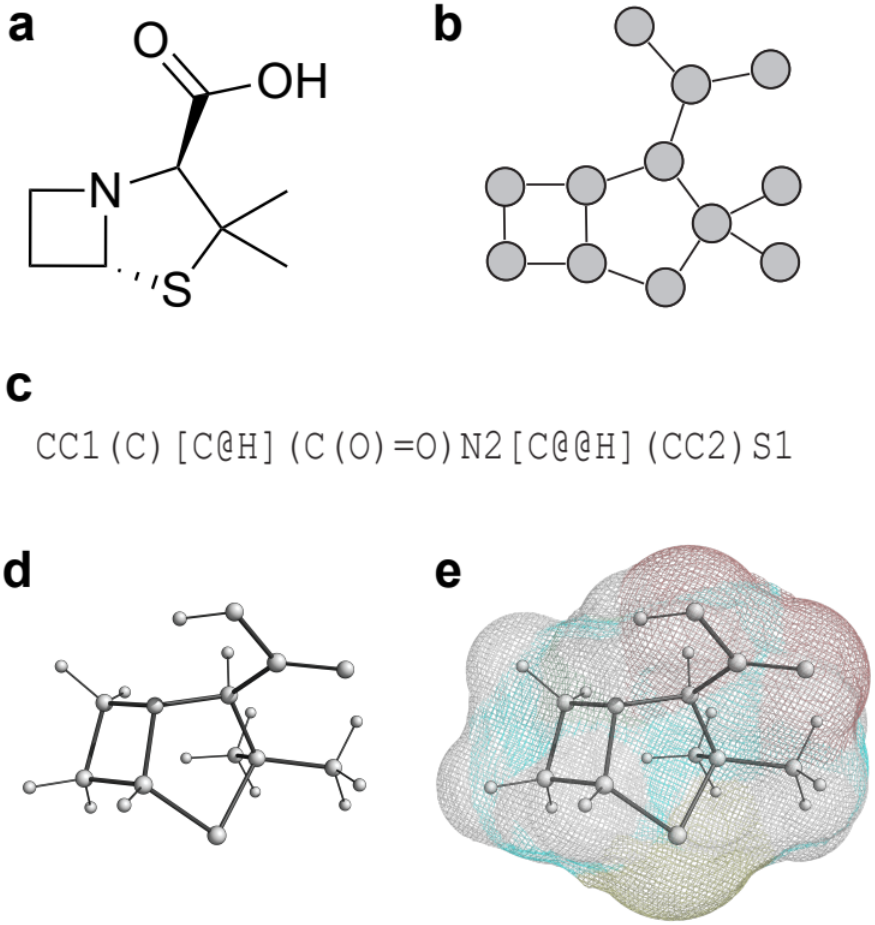


Рисунок 1.1 — Варианты представления молекулярных соединений для конкретной молекулы

гая вариант — топологические дескрипторы [12] — извлекают из структурной формулы вещества, дающей информацию о типах атомов и связях между ними, причем как для атомов, так и для связей дополнительно могут задаваться локальные стереохимические и физико-химические характеристики.

Топологические дескрипторы подразделяют на топоструктурные, содержащие информацию о смежности и топологических расстояниях между атомами, и топохимические, которые, кроме этого, указывают на элементную принадлежность атомов и гибридизацию. Среди топологических дескрипторов особую роль играют фрагментные дескрипторы, которые показывают наличие или отсутствие тех или иных фрагментов в структуре молекулы [13]).

1.1 QSAR - топологические дескрипторы

Многие топологические QSAR-методы так или иначе используют для построения модели локальные молекулярные характеристики (особенно для серий

соединений достаточно близкой структуры). Например, это относится к методу Ганча (Hanch) [14](константы заместителей в определенных положениях) и методу Фри-Уилсона [15](индикаторные переменные, описывающие присутствие в определенных положениях заданных заместителей).

В качестве молекулярных дескрипторов можно взять, например, фрагментные дескрипторы [13].

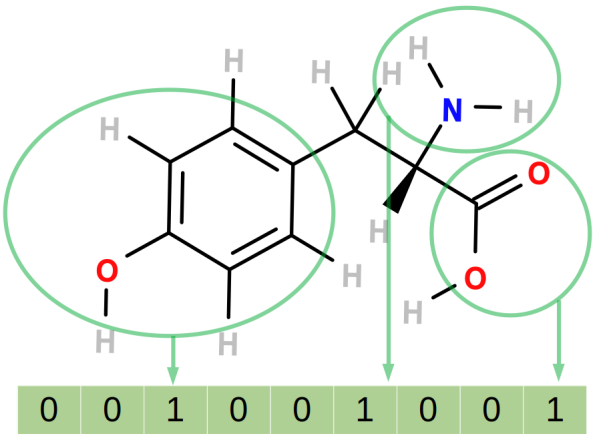


Рисунок 1.2 — Пример построения описания с фрагментными дескрипторами

Для построения фрагментных дескрипторов берется вся выборка молекул. Выбирается параметр k - длина цепочек. Чтобы определить, что такое цепочка, промаркируем атом каждой молекулы некоторой символьной меткой. В качестве меток, можно взять, например, название соответствующего атома.

Следующий шаг заключается в выборе всех связных ациклических подграфов молекулярного графа длины k. Каждый такой фрагмент обозначим последовательностью маркеров вершин атомов этой цепочки. Чтобы метка цепочки не зависела от обхода графа, запишем ее от минимального символа к максимального (в лексикографическом порядке).

В итоге каждая молекула будет представлять собой набор цепочек и количество вхождений каждой из цепочки.

Чтобы получить QSAR - матрицу, необходимо объединить все наборы цепочек, получив из них вектор. И для каждой молекулы, заполнить полученный вектор числом вхождений каждой из цепочек.

На основе полученного векторного описания, можно обучать все классические методы машинного обучения. Исследования по этой теме можно посмотреть в [38, 39]

Классическими линейными методами статистического анализа, традиционно используемым для целей QSAR/QSPR, являются множественная линейная регрессия (Multiple Linear Regression, MLR), метод частичных наименьших квадратов (Partial Least Squares, PLS), регрессия на главных компонентах (Principal Components Regression, PCR), гребневая регрессия (Ridge Regression, RR) [16].

К нелинейным методам относят искусственные нейронные сети (ANN) [17, 18], метод ближайших соседей (kNN) [19] и ряд других.

Однако, подобные методы имеют ряд недостатков. Данные методы рассматривают молекулу как набор изолированных элементов. При таком подходе теряется информация о взаимном расположении структурных элементов молекулы.

1.2 3D-QSAR

Поскольку практически все свойства химических соединений, обусловленные образованием межмолекулярных комплексов, зависят от их пространственного строения, методы 3D QSAR [20] являются ведущими при поиске новых биологически активных соединений.

К стандартными методам 3D QSAR можно отнести подходы, в основе которых лежит предположение о том, что биологическая активность лигандов обусловлена нековалентным взаимодействием с биологическими мишенями посредством молекулярных полей. В рамках этих методов для описания таких полей вычисляют энергию взаимодействия между атомами совмещенных в пространстве (выравненных) молекул и пробными атомами, помещенными в узлы воображаемой трёхмерной решётки. Такие энергии взаимодействия рассматривают как потенциалы молекулярных полей. На основе результатов расчёта формируют матрицу, каждая строка которой отвечает молекуле лиганда, а каждая колонка - энергии взаимодействия, рассчитанной на определенном узле решётки. Количественные соотношения между значениями энергий взаимодействия и значениями биологической активности получают с помощью статистического анализа на базе методов машинных обучения.

CoMFA. Исторически первым и до сих пор одним из наиболее распространённых методов 3D QSAR является CoMFA (Comparative Molecular Field

Analysis, метод сравнительного анализа молекулярных полей), разработанный в 1988 г. Крамером [21]. В рамках этого метода в качестве дескрипторов используются потенциалы электростатического и стерического полей, рассчитанные на узлах гипотетической трёхмерной решётки 1.3.

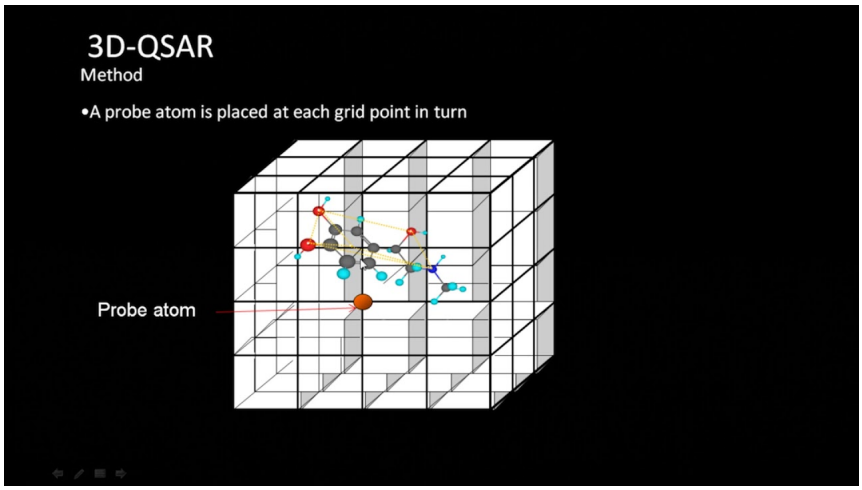


Рисунок 1.3 — CoMFa

Электростатические и стерические поля обычно считаются достаточными для описания нековалентных взаимодействий между лигандом и биологической мишенью. Для расчёта потенциалов электростатического поля в узлы решётки помещают пробные атомы водорода с зарядом +1 (протон), а для расчёта потенциалов стерического поля - атомы углерода в sp3-гибридизации. Электростатические потенциалы рассчитывается по закону Кулона, а стерические - с использованием потенциала Леннард-Джонса 6-12. В качестве стандартного метода статистического анализа используется метод частичных наименьших квадратов PLS (Partial Least Squares).

GRID. Программа GRID (Graphic Retrieval and Information Display) [22] применяется в качестве альтернативы методу CoMFA [23]. Программа GRID также рассчитывает взаимодействие между молекулой и пробными атомами, расположенными в узлах трёхмерной решётки, но имеет ряд преимуществ перед CoMFA: во-первых, вместо потенциалов Леннард-Джонса 6-12 используются более гладкие функции типа 6-4; во-вторых, для описания большого разнообразия типов межмолекулярного взаимодействия в методе GRID используют значительно большее число различных пробных атомов и даже групп атомов. В частности, в дополнение к электростатическим и стерическим потенциалам, программа вычисляет потенциалы водородной связи и гидрофобный потенциал.

CoMSIA. Другой метод, широко используемый в 3D QSAR, Метод сравнительного анализа индексов молекулярного подобия (Comparative Molecular Similarity Indices Analysis, CoMSIA) [24] был разработан как развитие метода CoMFA. Подробное описание этого метода и его модификаций приведено в работах [25, 26]. В рамках этого подхода рассчитываются индексы молекулярного подобия, которые используются в качестве дескрипторов. Расчёт проводят путём сравнения каждой молекулы базы с пробными атомами радиуса 1Å с зарядом +1 и гидрофобностью +1, помещёнными в узлах решётки. Наиболее часто с помощью индексов молекулярного подобия описывают электростатические, стерические, гидрофобные поля, а также поля водородных связей. В отличие от CoMFA, для описания потенциалов в этом методе используются функции Гауссова типа, что позволяет избежать резких изменений при переходе из одной ячейки к другой и не требует введения ограничительных значений для потенциалов сверху. Кроме того, модели, полученные методом CoMSIA, легче интерпретировать визуально.

Однако данные методы являются вычислительно дорогостоящими. Это может сделать их неприменимыми для больших выборок.

1.3 Нейросетвые подходы решения QSAR задачи

Варианты моделей, описанные раннее, строили вектор дескрипторов для дальнейшего обучения моделей. Однако, на фоне рассвета нейросетевых подходов в машинном обучении возникла область называемая - геометрическим глубоким обучением (GDL)[40]. Термин геометрическое глубокое обучение был придуман в 2017 году. Хотя GDL первоначально использовался для методов применительно к неевклидовым данным [41], теперь он распространяется на все методы глубокого обучения, включающие информацию о структуре и симметричности интересующей системы [42].

Данные методы способны обрабатывать различные типы геометрических данных, в том числе есть подходы, способные работать напрямую с каждым конкретным представлением молекул изображенным на рисунке 1.1. Заметим, что данные модели могут не только решать исходную задачу, но и строить векторное описание молекул, которое в дальнейшем будет использовано в других моделях машинного обучения [31, 32] . На рисунке 1.4 изображено идейное

различие описанных подходов. В этой работе мы расммотрим подмножество этих методов, которые работают с графовыми структурами.

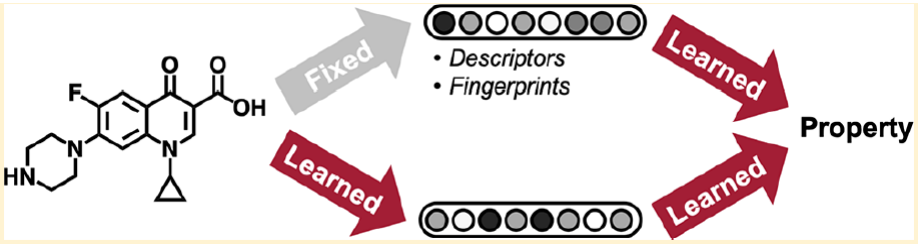


Рисунок 1.4 — Идеи подходов решения QSAR задачи

Графовые нейронные сети (GNN)[27]- нейронные сети со специальной архитектурой, которая работает непосредственно с матрицей смежности молекулярных графов, а также матрицами признаков вершин и ребер.

В качестве примера таких сетей, можно привести:

1. GCN [28]
2. ChemProp [29]
3. MoleculaNet [30]

Примеры использования GNN:

1. Предсказание химической реакции - Graph Transformation Policy Network [33] кодирует входные молекулы и генерирует промежуточный граф с сетью предсказания пары узлов и сетью стратегии.

2. Прогнозирование белкового интерфейса. Белки взаимодействуют друг с другом с помощью интерфейса, который образован аминокислотными остатками из каждого участвующего белка. Задача прогнозирования интерфейса белка состоит в том, чтобы определить - являются ли определенные остатки частью белка. Обычно, предсказание для одного остатка зависит от других соседних остатков. Позволяя остаткам быть узлами, белки могут быть представлены в виде графиков, которые могут использовать алгоритмы машинного обучения на основе GNN. [34] предлагают основанный на GCN метод для изучения представления остатков лиганда и рецепторного белка и объединения их для парной классификации. MR-GNN [35] вводит подход с несколькими разрешениями (multi-scale) для извлечения и суммирования локальных и глобальных особенностей для лучшего прогнозирования.

1.4 Вывод

В данной главе проведен обзор вариантов построения QSAR - моделей. Варианты существенно зависят от способа представления молекулярных соединений. Есть 2 основных подхода к построению QSAR моделей. Первые строят вектор описания химического соединения из молекулярных дескрипторов - инвариантов молекулярного графа. На основе построенного описания обучаются модели машинного обучения - классификаторы или регрессоры (в зависимости от целевого свойства). Другой подход заключается в применении современных нейросетевых архитектур к сложным геометрическим вариантам представления молекулярных соединений для построения прогнозирующей модели. Одним из типов таких моделей являются графовые нейронные сети - нейросетевые архитектуры способные работать с плоскими графами, представленными матрицей смежности и векторным описанием вершин и ребер графа. Преимущество этих методов в том, что они способны самостоятельно извлекать признаки из молекул и строить оптимальное векторное представление объектов с точки зрения конкретной задачи.

Глава 2. Графовые нейронные сети

Граф - структура данных, которая представляется наборов вершин (nodes) и их отношений (edges). Исследователи уделяет все больше внимания анализу графов с помощью методов машинного обучения, потому что они обладают большой выразительность во многих задачах, таких как: анализ социальных сетей, анализ физических систем, QSAR задачах, графах знаний и тд.

Проблема анализа графов заключается в том, что это не евклидовая структура, т.е для нее нет готового векторного представления в некотором линейном пространстве. Это значит, что в исходном виде методы машинного обучения к ним неприменимы. Чтобы решить эту проблему, исследователям приходилось самим придумывать для графов некоторые векторные представления на основе отдельных признаков и структурных особенностей. Однако с достаточной полнотой и выразительностью придумать описание такой сложной структуры как граф очень сложно. С таким подходом нет никаких гарантий, что предложенное описание и используемые в нем признаки будут оптимальными для каждой конкретной задачи.

После прорыва в области компьютерного зрения, который совершили сверточные нейронные сети, исследователи решили придумать обобщение этого подхода на структуру графа. Это позволило бы нейронной сети самой извлекать нужные признаки из графа и составлять для нее лучшее векторное представление для каждой отдельной задачи. Таким образом появились графовые нейронные сети [47].

Графовые нейронные сети (GNN[27]) - методы глубокого обучение, которые работают непосредственно со структурой графа. Они получилили широкое распространение в последнее время.

2.1 Определения и обозначения

$G = (V, E)$ - граф, где $|V| = N$ - число вершин в графе, а $|E| = N_e$ - число ребер.

$A \in R^{N \times N}$

h_v - векторное представление вершины v, скрытое состояние вершины.

h_v^t - скрытое состояние вершины в момент t (на t-м слое).

e_{vw} - векторное представление ребра vw.

$\odot$ - поэлементное умножение

I_N - единичная матрица размерности N

σ- функция сигмоиды

N_v - множество вершин, смежных с v.

$|x|$ - мощность множества x

$||$ - операция конкатенации

2.2 Постановка задачи

Пусть дана обучающая выборка:

$$X = \{(G_1, y_1), (G_2, y_2), ..., (G_N, y_N)\} \in (\hat{G} \times Y)^N$$

G_i - молекулярные граф представленный набором вершим и ребер (V_i, E_i). Из данного представления можно получить матрицу смежности графа - A_i. Каждая вершино и ребро графа имеют векторно описанные признаки: e_{vw} - векторное представление ребра vw, где $v, w \in V$ и x_v - признаки вершины $v \in V$. Y - целевая переменная, которая в общем случае представляет из себя вещественное число. Но в случае классификации это могут быть метки конечного числа классов.

Требуется построить модель $a : \hat{G} \to Y$, апроксимирующий целевую зависимость на множестве элементов X.

В качестве модели возьмем одну из нейросетевых архитектур, которая параметризуется набором параметров W. Возьмем функцию потерь L, которая подбирается под конкретный тип целевой переменной Y. С помощью классического алгоритма обучения нейронных сетей - градиентного спуска обучим параметры нашей сети минимизируя функционал:

$$\hat{L}(W, X) = \frac{1}{N} \sum_{i}^{N} L(a(W, G_i), y_i) \to min$$

.

2.3 Процесс разработки GNN

Рассмотрим общий процесс разработки GNN [48] для некоторой задачи:

1) Необходимо определить структуру графа.

Обычно существует два сценария: структурные сценарии и неструктурные сценарии. В структурных сценариях структура графа явно указана в задаче. Это относится к молекулам, физическим системам, графам знаний и так далее. В неструктурных сценариях графы не указаны, поэтому мы должны сначала построить граф на основе задачи, например, построить «словесный» графа для текста или построить граф сцены для изображения.

2) Необходимо определить тип графа и его размер.

Графы обычно классифицируют на ориентированные и неориентированные, гомогенные и гетерогенные, статические и динамические. В зависимости от типа графа может менятся выбор архитектуры сети и подходы к обучению.

Для размера графа нет четкой классификации на “маленькие” и “большие”. Это зависит от имеющихся вычислительных ресурсов (например, объема памяти GPU). Будем считать, что граф “большой”, если его нельзя обработать на 1 GPU.

3) Выбор функции потерь. Он зависит от типа задачи и подхода к обучению.

Задачи на графах обычно делятся на:

- Node-level задачи, которые фокусируются на вершинах. Например, задача классификации вершины.
- Edge-level задачи, которые включают, например, классификацию ребер или предсказание связи.
- Graph-level задачи включают классификацию графов, регрессионные задачи и тд.

Подходы к обучению разделяются на классические подходы машинного обучения: supervised, unsupervised и semi-supervised. Примеры различных графов можно увидеть на рисунке ниже.

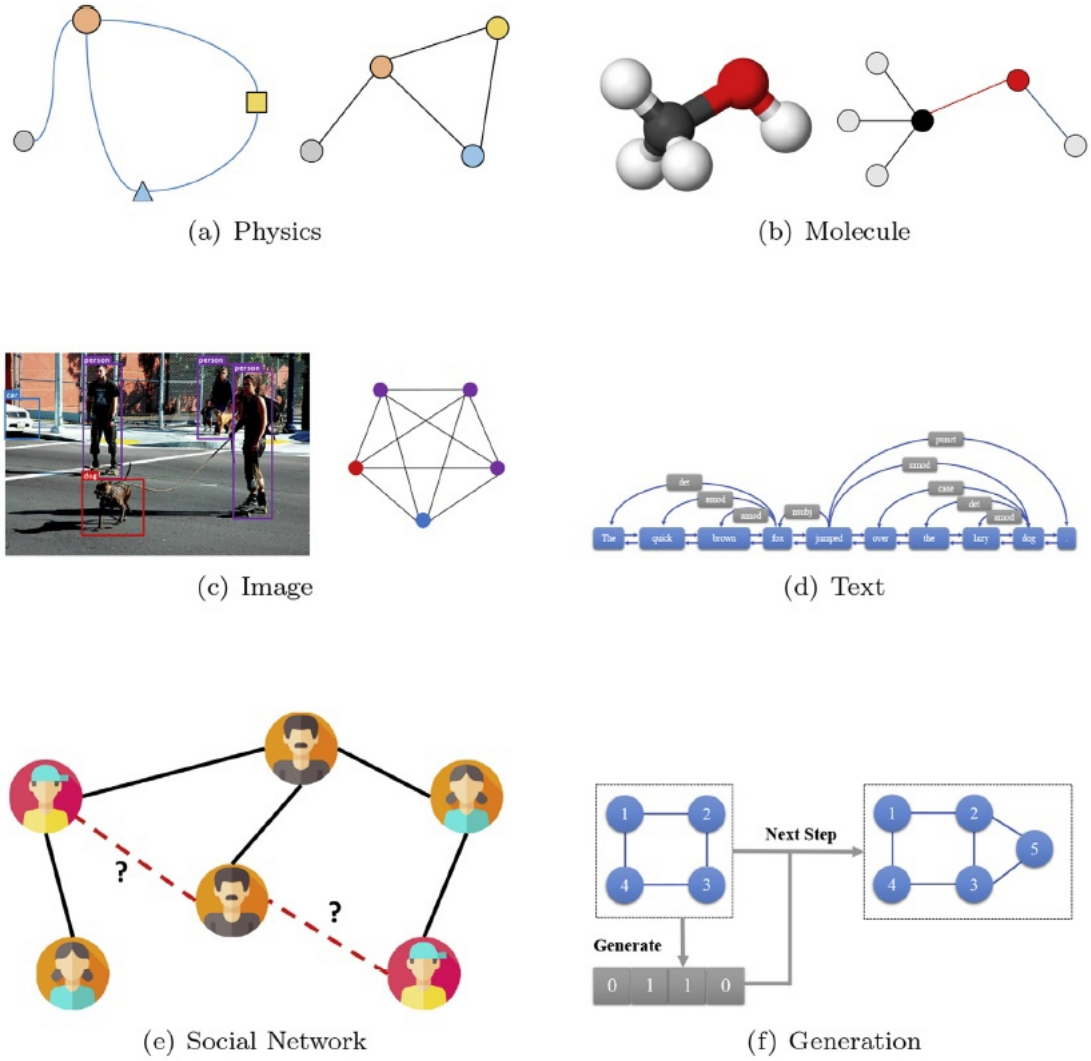


Рисунок 2.1 — Примеры графов в разных задачах

4) Последний шаг - выбор архитектуры. Он происходит на основе 3х основных вычислительных компонент из которых состоят слои графовой нейронной сети.

– Propagation Module
– Sampling Module
– Pooling Module

Подробнее про архитектуры GNN мы поговорим ниже.

2.4 Общая архитектура GNN

Общая архитектура GNN представляет из себя некоторое количество GNN слоев (GNN Layer). Эти слои принимают на вход признаки вершин графа (матрицу X размера $N \times M$, где N - число вершин, M - размерность описания

каждой вершины), признаки ребер графа (матрицу E размера N^e , где N^e - число ребер, M^e - размерность описания каждого ребра) и матрицу смежности A размера $N \times N$. На картинке 2.2 можно увидеть пример архитектры вариационного графового автокодировщика [ссылка] с аналогично описанными входными данными.

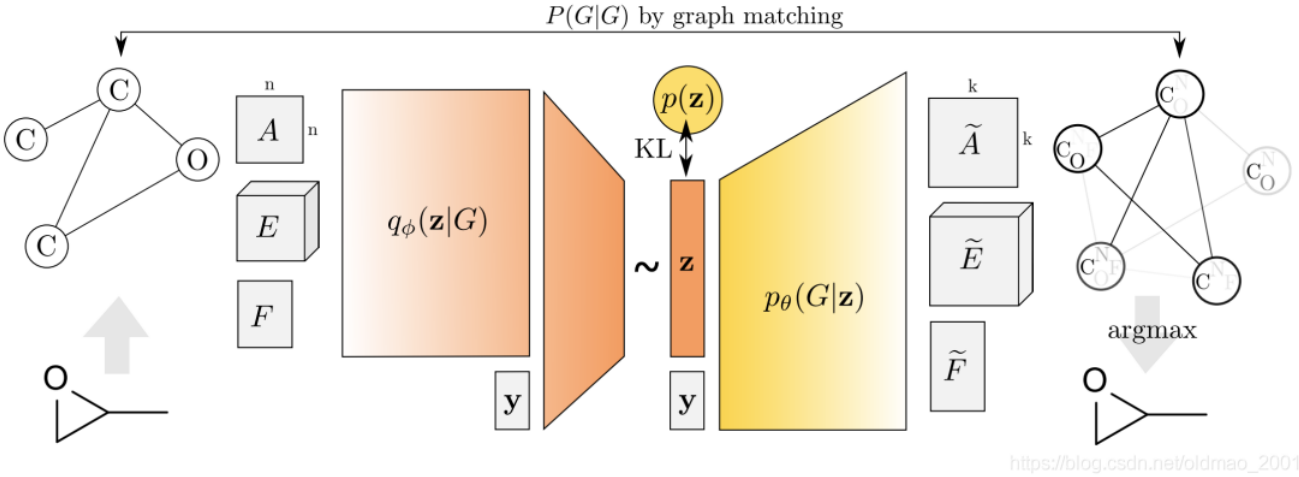


Рисунок 2.2 — Пример архитектуры вариационного графового автокодировщика

Каждый слой GNN, как показано на рисунке 2.3, состоит из 3х блоков, указанных в 4 пункте процесса разработки.

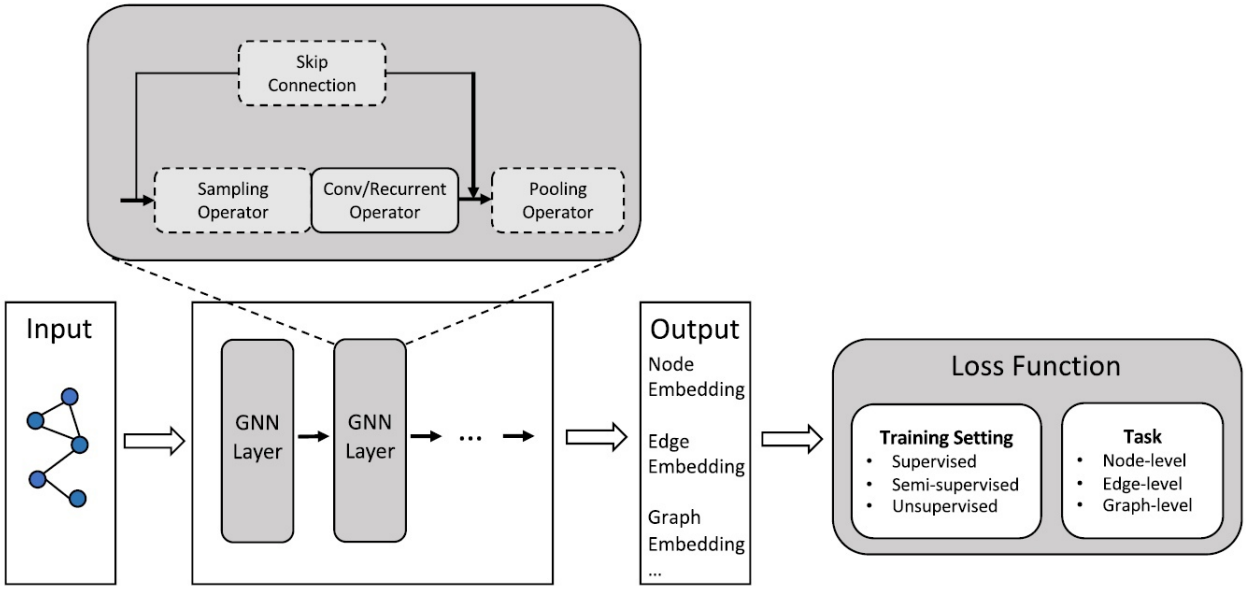


Рисунок 2.3 — Общая архитектура GNN. Входной граф проходит через несколько слоев GNN, затем полученные выходные представления отправляются в функцию потерь.

Propagation Module - модуль распространения, который используется для распространения информации между узлами, чтобы агрегировать информацию как из признаков, так и топологическую информацию графа. В этом модуле

обычно используется оператор свертки (convolution operator) или рекуррентный оператор (recurrent operator) для агрегирования информации от соседей.

Skip Connection используется для учета более ранней информации из вершин и решения проблемы затухания градиента.

Sampling Module отвечает за поведение propagation module в случае, если граф большой. Например, это может быть семплирование некоторого подграфа, к которому будет применяться оператор свертки.

Pooling Module используется когда нам нужны представления высокого уровня: подграфа или графа. Модули объединения необходимы для извлечения информации из вершин и построения итогового представления. Например, это может быть суммирования векторов вершин, которое будет представлять нам описание всего графа.

После t - ого GNN слоя мы получаем матрицу H^t , которая является обновленным вариантом матрицы X, т.е новым описанием наших вершин. Ее строки мы будем называть скрытыми состояниями вершин. После прохода последнего слоя мы получим итоговые векторные представления для вершин, ребер и при необходимости для всего графа (с помощью pooling module на последнем слое). Эти вектора мы используем для решения исходной задачи. Например, можно отправить такие вектора в MLP, который выдаст нам класс графа или вещественное число. Предсказанное значение отправляется в функцию потерь (подобранную под задачу) и методом градиентного спуска происходит обучение внутренних слоев GNN сети.

Теперь подробнее рассмотрим варианты **propagation module.** Имеется 2 типа модулей: сверточные и рекуррентные. Мы в основном будем говорить про сверточные модули, т.к они чаще всего используются в GNN сетях. Сверточные модули подразделяются на спектральные (spectral) и пространственные (spatial) подходы.

Спектральные подходы работают со спектральным представлением графов. Эти методы теоретически основаны на графовой обработке сигналов [46] и определяют оператор свертки в спектральном пространстве.

В спектральных методах сигнал графа x сначала преобразуется в спектральное пространство преобразованием Фурье F на графе, затем выполняется свертка. После свертки результирующий сигнал преобразуется обратно с помощью обратного преобразования Фурье F^{-1} на графе. Преобразование Фурье графа имеет вид: $F(x) = U^T x$, $F^{-1}(x) = Ux$, где U - матрица собственных

векторов нормализованного Лапласиана графа $L = I_N - D^{-1/2}AD^{-1/2}$. D - диагональная матрица степеней вершин.

В итоге оператор свертки матрицы g с сигналом x будет иметь вид:

$g \star x = F^{-1}(F(g) \odot F(x)) = U(U^T \text{g} \odot U^T x)$,

что в случае диагональной матрицы g_w имеет вид $g_w \star x = UgU^T$x. g_w - это и есть фильтр или веса, которые будут обучаться в процессе обучения самой сети.

Рассмотрим несколько конкретных архитектур:

1) **ChebNet**[45]. Идея этой архитектуры в том, чтобы аппроксимировать операцию свертки многочленами Чебышева до K-й степени включительно. Операция свертки принимает вид: $g_w \star \text{x} \approx \Sigma_{k=0}^{K} w_k T_k()x$, где $= \frac{2}{\lambda_{\max}} L - I_N$

($\lambda_{\max}$ - максимально собственное значение L.)

1) **GCN**[28]. (Graph Convolution Network). Архитектура совпадает с предыдущей при выборе параметра $K = 1$. Это позволяет бороться с переобучение и ускорить процесс обучения. Операция свертки принимает вид $g_w \star \text{x} \approx w(I_N - D^{-1/2}AD^{-1/2})x$. Для решения проблемы затухающих / взрывающихся градиентов был придуман трюк перенормировки. В предыдущей формуле делается замена $I_N - D^{-1/2}AD^{-1/2} \rightarrow {}^{-1/2}\,{}^{-1/2}$, где $= A + I_N$ и $_{\text{ij}} = \Sigma_j\,{}_{\text{ij}}$.

В итоге формула обновления скрытых состояний принимает вид:

$H = {}^{-1/2}\,{}^{-1/2}\text{X W}$,

где $X \in \Re^{N \times M}$ - входная матрица признаков вершин (т.е N - векторов размерности M), $W \in \Re^{M \times M'}$ - матрица обучаемых параметров, $H \in \Re^{N \times M'}$ - новая матрица скрытых состояний.

Пространственные подходы (spatial) определяют свертки непосредственно на графе и основываются на его топологии. Основная проблема пространственных подходов - определение операции свертки с окрестностями разного размера (с переменным числом соседей у вершины) и поддержание локальной инвариантности как у CNN (т.е чтобы выход свертки не зависел от расположения вершины, к которой применяется свертка).

Рассмотрим несколько примеров конкретных архитектур:

1) **Neural FPs** [31]**.** Эта архитектура используем различные матрицы весов для вершин с различным числом соседей. Формула обновления скрытых состояний принимает вид:

$$t = {h_v}^t + \Sigma_{\mathrm{u} \in \aleph_v} {h_u}^t$$
$${h_v}^{t+1} = \sigma(t\, {W^{t+1}}_{|\aleph_v|}),$$

где ${W^{t+1}}_{|\aleph_v|}$-матрица весов для вершин степени $|\aleph_v|$.

1) **GraphSAGE** [44]. Общих подход применяющий семплирование и агрегацию информации из соседей для данной вершины. Формулы обновления имеют вид:

$${h_{\aleph_v}}^{t+1} = AGG_{t+1}\left(\{{h^t}_u, \vee \mathrm{u} \in \aleph_v\}\right),$$
$${h_v}^{t+1} = \sigma(W^{t+1} * [{h^t}_v \,||\, {h_{\aleph_v}}^{t+1}]).$$

Этот метод используется не все множество соседей вершины, а семплирует некоторое фиксированное количество из равномерного распределения. AGG_{t+1} - функция агрегации, которая может быть, например, средним или LSTM.

1) **GAT. (Graph Attention Network)**[43]**.** Метод добавляет механизм внимания в propagation module. Т.е сначала определяется “важность” соседей, а затем обновляются скрытые состояния на основе взвешивании соседей:

$${h_v}^{t+1} = \rho(\Sigma_{u \in \aleph_v}\, \alpha_{\mathrm{vu}} W {h^t}_u)$$
$$\alpha_{\mathrm{vu}} = \frac{exp(LeakyRely(a^T[Wh_v||Wh_u]))}{\Sigma_{k \in \aleph_v} exp(LeakyRely(a^T[Wh_v||Wh_k]))}$$

где W - это матрица весов, a - вектор весов.

2.5 Massage Passing Neural Network

Это некоторые конкретные методы, которые относятся к spatial подходу. Однако их всех объединяет одна и та же идея: обновление информации внутри вершины, происходит на основе информации, которая содержится в соседних

вершинах. Поэтому был предложен общий фреймворк для описания подобных архитектур, который называется MPNN - Message Passing Neural Network[36].

Такие модели содержат 2 фазы: фаза передачи сообщения и фаза считывания. В первую фазу используется функция M_t, которая агрегирует информацию от соседей и делает из него "сообщение" $m^t{}_v$. Затем используется функция U_t, которая обновляет скрытое состояние вершины $h_v{}^t$:

$$m_v^{t+1} = \Sigma_{u \in \aleph_v} M_t(h^t{}_v, h_u{}^t, e_{vu})$$
$$h_v^{t+1} = U_t(h^t{}_v, m_v{}^{t+1}).$$

На фазе считывания применяется функция R, которая вычисляет итоговое значение, требующееся в задаче (это может быть класс самого графа или, например, вещественное число связанное с каждой отдельной вершиной):

$$\widehat{y} = R(\{h_v^t \mid v \in G\}).$$

Для определения архитектуры разработчику необходимо лишь определить функции M_t, U_t, R. Функции M_t, U_t, R - обучаемые дифференцируемые функции. Функция R - работает с множеством состояний вершин, т.е должна быть инварианта к перестановке аргументов, чтобы сеть получилась инвариантой к изоморфизму графов.

Общая идея spatial подходов изображена на рисунке 2.4.

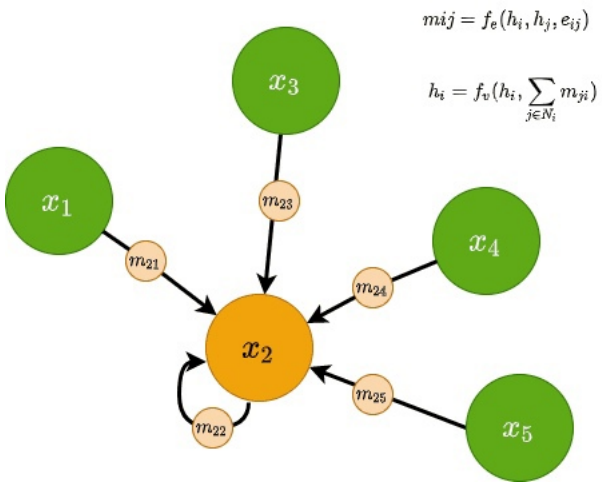


Рисунок 2.4 — Передача сообщений между вершинами. Вершины x_1, x_3, x_4, x_5 передают информацию вершине x_2.

2.6 Вывод

В этой главе был описан общий подход к построению архитектуры графовой сети, рассмотрены различные варианты построения ее внутренних слоев. Обучение таких сетей происходит за счет классического алгоритма обучения нейронных сетей - градиентного спуска. Графовые нейронные сети - это множество архитектур, которые появились сравнительно недавно. Они все чаще успешно используются исследователями, которые работают с графовыми структурами, включая молекулярные графы. За последнее время с помощью них было получено много интересных результатов[1], но при этом все равно остается множество направлений для исследования в этой теме.

[1] Список статей по этой теме можно найти по ссылке: *https://github.com/thunlp/GNNPapers*

Глава 3. Графовые нейронные сети в задаче предсказания свойств молекулярных графов

В прошлой главе был рассмотрен общий подход к построению графовых нейронных сетей, а также некоторые конкретные архитектуры. Однако эти архитектуры имеют общую направленность и не все из них созданы специально для решения задачи предсказания молекулярных свойств. Графовые представления молекулярных соединений имеют свои особенности, связанные с разными типами связей, наличием циклов, вариантами описания атомов и связей и тд. Поэтому в этой главе будут рассмотрены архитектуры и подходы созданные специально для решения задачи предсказания свойств молекул.

3.1 ChemProp

Еще одной интересной архитектурой является - ChemProp [29]. Она предлагает измененный вариант MPNN архитектуры - Directed-MPNN. Ее основная идея состоит в передаче информации не между вершинами, а между ребрами графа. Т.е на каждой итерации обучаются скрытые представления не вершин, а ребер. Также граф рассматривается как ориентированный, что дает нам возможность распространять информацию по направлению ребер. Пример работы архитеткуры приведен на рисунке 3.1. Мотивацией к такому переходу служит зацикливание информации при стандартном атомарном способе распространения сообщений. Т.к через один шаг информация дойдет от вершины к ее соседям, а на следующий шаг информация вернется обратно в текущую вершину. Это вносит шум в представление графа и мешает распространению информации на длинное расстояние.

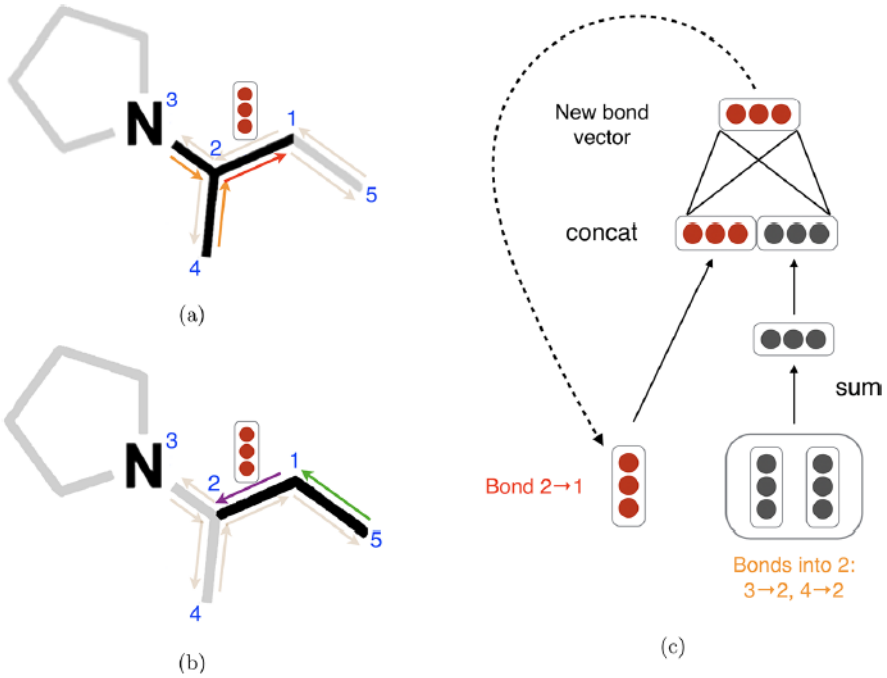


Рисунок 3.1 — Пример обновления скрытого состояния в ChemProp. (a) Информация из оранжевых связей используются для обновления скрытого состояния красной направленной связи. Напротив, в традиционной MPNN сообщения передаются от атомов к атомам (например, атомы 1, 3 и 4 к атому 2), а не от связей к связям. (b) Аналогично, информация от зеленой связи используется для обновления скрытого состояния фиолетовой направленной связи. (c) Иллюстрация функции обновления скрытого представления красной направленной связи на диаграмме (a).

Архитектуры имеет несколько основных шагов.

Сначала инициализируются скрытые состояния ребер:

$$h_{vw}^0 = ReLU(W_i concat(x_v, e_{vw}))$$

, W_i - обучаемая матрица $\in R^{h \times h_i}$, $concat(x_v, e_{vw}) \in R^{h_i}$ - конкатенация признаков x_v для вершины v и признаков e_{vw} для связи vw.

Затем обновляются их скрытые состояния:

$$m_{vw}^{t+1} = \sum_{k \in \{N(v) \setminus w\}} h_{kv}^t$$

$$h_{vw}^{t+1} = ReLU(h_{vw}^0 + W_m m_{vw}^{t+1}), t \in 1,..,T$$

Затем определяются скрытые состояния вершин:

$$m_v = \sum_{w \in N(v)} h_{vw}^T$$

$$h_v = ReLU(W_a concat(x_v, m_v))$$

Readout слой представляет собой простое суммирования:

$$h = \sum_{v \in G} h_v$$

W_i, W_m, W_a - обучаемые параметры сети, которые используются на всех шагах распространения информации. Это значит, что даную архитектуру можно отнести к подмножеству рекурентных графовых нейронных сетей. Параметр T отвечается за количество шагов распространения информации.

Финальный шаг для предсказания целевого свойства выглядит так:

$$\hat{y} = f(h)$$

, где f - полносвязная нейронная сеть.

Исходные признаки атомов и связей описаны в таблицах 3.1 и 3.2 соответственно. Все атомарные признаки закодированы one-hot кодированием, за исключением атомной массы, которая представляет собой действительное число, масштабированное для того же порядка величины.

Данная архитектура показывает лучшие результаты для задач классификации среди чисто MPNN архитетуру [49]. Реализация сети представлена в фреймворке [51]. Он позволяет быстро обучить данную архитектуру на произвольных данных с различным типом задач.

3.2 GROVER

Сущестует две основные проблемы препятствующие использованию GNN в реальных сценариях: (1) недостаточность размеченных молекул для контролируемого обучения; (2) плохая обобщающая способность к новосинтезированным молекулам.

Для решения этих проблем, авторы статьи [49] предложили архитектуру $GROVER$. С помощью разработанных задач без учителя на уровне узла, ребра и графа, $GROVER$ может извлекать богатую структурную и семантическую информацию о молекулах из огромного количества немаркированных молекулярных данных. Чтобы кодировать такую сложную информацию, $GROVER$

Таблица 3.1 — Признаки атомов.

Признак	Описание	Размерность
тип атома	тип атома (например, C, N, O); по атомному номеру	100
# bonds	количество связей, в которых участвует атом	6
формальный заряд	целочисленный электронный заряд, присвоенный атому	5
хиральность	не указана, тетраэдрическая CW/CCW или другая	4
# Hs	число связанных атомов водорода	5
гибридизация	sp, sp2, sp3, sp3d или sp3d2	5
ароматичность	является ли этот атом частью ароматической системы	1
атомная масса	масса атома, деленная на 100	1

Таблица 3.2 — Признаки связей.

Признак	Описание	Размерность
тип связи	одинарная, двойная, тройная или ароматическая	4
сопряжена	сопряжена ли связь	1
в кольце	независимо от того, является ли связь частью кольца	1
стерео	нет, любой, E/Z или цис/транс	6

интегрирует $MPNN$ сети в архитектуру в стиле Transformer [50], чтобы предоставить класс более выразительных кодировщиков молекул. Гибкость $GROVER$ позволяет эффективно обучать его на крупномасштабном наборе молекулярных данных, не требуя какой-либо разметки. Авторы обучили $GROVER$ со 100 миллионами параметров на 10 миллионах неразмеченных молекул, что на момент выхода статьи являлось самой большой GNN сетью на самом большом наборе обучающих данных.

Ключевой архитектурной идеей является адаптация архитектуры Transformers для графовых данных, которая называется GTransformers. Механизм внимания – это основной строительный блок Трансформера, который представляет из себя взвешеную сумму входных элементов. Один слой внимания принимает набор запросов, ключей, значений (q, k, v) в качестве входных данных. Затем он вычисляет скалярные произведения запрос со всеми ключами и применяет функцию softmax для получения весов значений. Укладывая набор (q, k, v) в матрицы (Q, K, V), он допускает высокооптимизированное матричное умножение. В архитектуре используется несколько параллельных механизмов внимания, которые называются многоголовым вниманием (multi-head attention). В частности, выходы могут быть организованы в виде матрицы

$$\text{Attention}\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right) = \text{softmax}\left(\mathbf{Q}\mathbf{K}^{\top}/\sqrt{d}\right)\mathbf{V},$$

где d это размерность $\mathbf{q}$ and $\mathbf{k}$. Предположим, мы упорядочили k слоев внимания в многоголовом внимание, то его выходная матрица может быть записана как

$$\begin{aligned}\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \text{Concat}\left(\text{ head }_1, \ldots, \text{ head }_k\right)\mathbf{W}^{O} \\ \text{head }_i &= \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^{\mathbf{Q}}, \mathbf{K}\mathbf{W}_i^{\mathbf{K}}, \mathbf{V}\mathbf{W}_i^{\mathbf{V}}\right)\end{aligned}$$

где $\mathbf{W}_i^{\mathbf{Q}}, \mathbf{W}_i^{\mathbf{K}}, \mathbf{W}_i^{\mathbf{V}}$ проекциооные матрицы для i-й головы.

Ключевой компонент слоя GTransformer — это графовый многоголовый механизм внимания, представляющий собой блоки внимания, адаптированные к структурным входным данным. Стандартный блок внимания требует векторизованных входных данных. Поэтому авторы используют динамическую GNN - архитектуру (dyMPN), чтобы извлекать векторы запросов, ключей и значений из узлов графа, а затем передавать их в блок внимания. Эта стратегия проста, но мощна, поскольку позволяет используя очень выразительные модели GNN лучше моделировать структурную информацию в молекулярных данных. Визуализацию архитектуры можно увидеть на рисунке 3.2

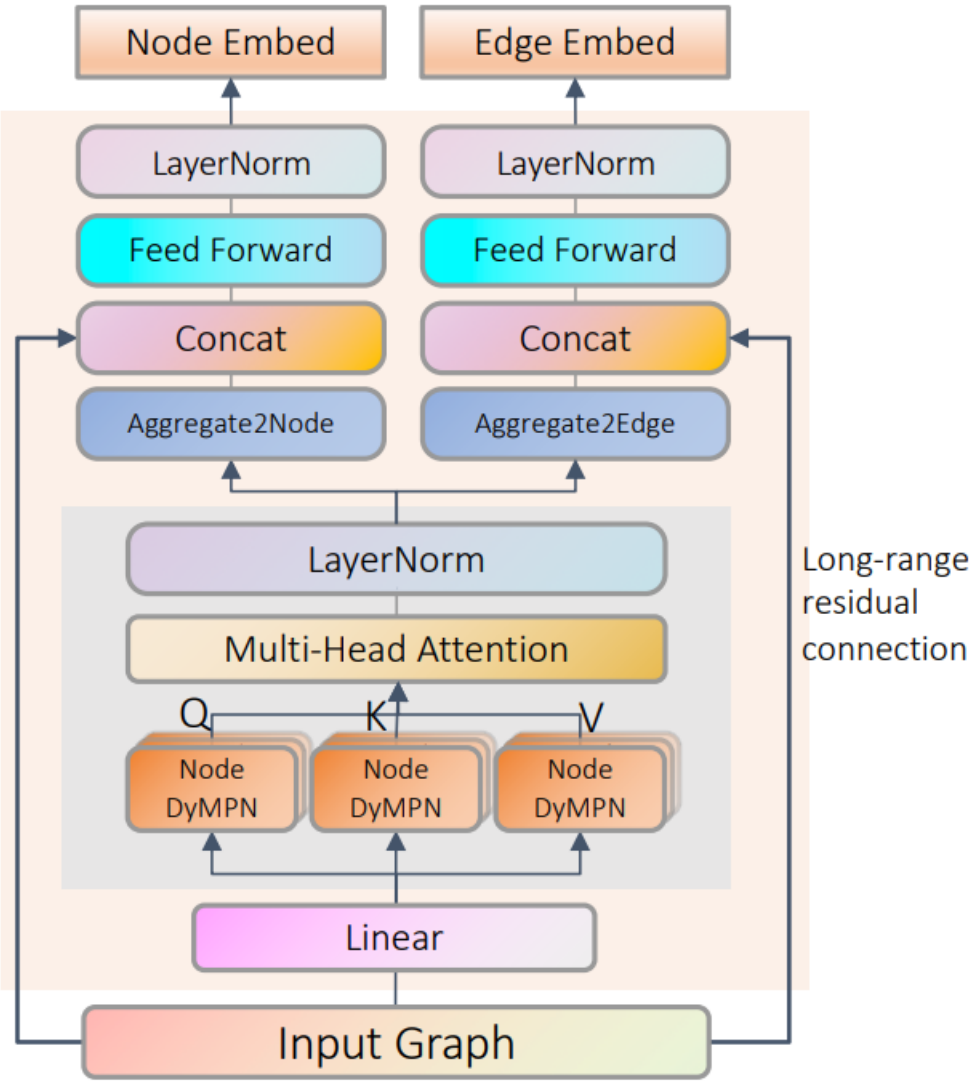


Рисунок 3.2 — Архитектура графового трансформера

Для задачи на уровне узла/ребра вместо предсказания только типа узла/ребра, $GROVER$ случайным образом маскирует локальный подграф целевого узла/ребра и предсказывает это контекстуальные свойства на основе векторных представлений узлов. Для задач на уровне графа, используя знания предметной области, $GROVER$ извлекает семантические мотивы - повторяющиеся подграфы среди входных данных графа, которые преобладают в данных молекулярного графа, и предсказывает возникновение этих мотивов для молекулы из ее векторного представления. Визуализация изображена на рисунке 3.3

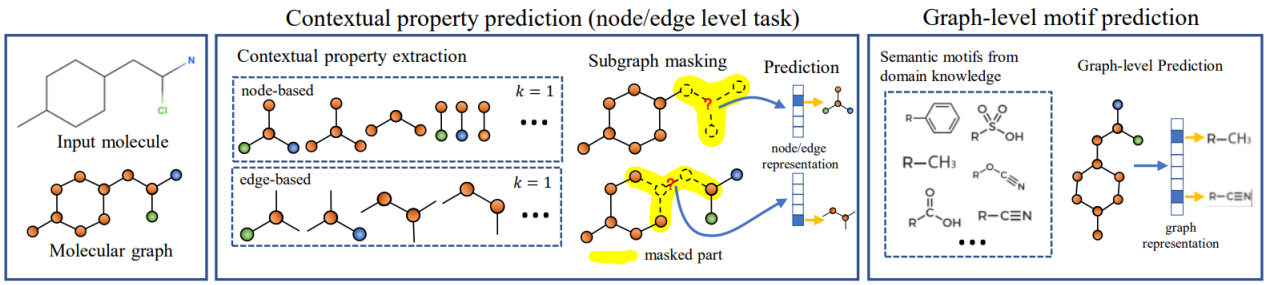


Рисунок 3.3 — Обзор sefl-supervised задач в архитеткуре GROVER

Авторы использовали предварительно обученный $GROVER$ для задачи прогнозирования молекулярных свойств. После дообучения модели под конкретные задачи GROVER добился относительного улучшения на 22,4 % по сравнению [53] и относительное улучшение на 7,4% по сравнению с [52] по задачам классификации. Более того, на момент выхода статьи даже по сравнению с текущими state-of-the art результатами для каждого набора данных наблюдался большой прирост качества от $GROVER$ (в среднем более 6%) по 11 популярным бенчмаркам.

3.3 Варианты улучшения архитектур нейронных сетей

Рассмотрим варианты оптимизации для повышения производительности. Несмотря на то что в идеале $MPNN$ должна быть способна извлекать любую информацию о молекуле, которая может иметь отношение к предсказанию данного свойства, на практике этому могут помешать два ограничения. Первый, многие наборы данных для предсказания свойств очень малы, т.е. порядка сотен или тысяч молекул. С таким небольшим набором данных MPNN архитектуры не могут научиться идентифицировать и извлекать все особенности молекулы, которые могут иметь отношение к предсказываемому свойству. Помимо этого они подвержены негативному влиянию шума в данные. Во-вторых, большинство MPNN используют меньше шагов для передачи сообщений, чем диаметр молекулярного графа, т. е. $T < diam(G)$. Это означает, что атомы, которые находятся на расстоянии большем, чем T друг от друга никогда не будут получать сообщения друг о друге. Это приводит к молекулярному представлению, которое является в основном локальным, а не глобальным по своей природе. Это означает, что MPNN может с трудом предсказать свойства, которые существенно зависят от глобальных свойств.

Одним из вариантов решения этих проблем является добавление в модель глобальных признаков молекул, которые могут быть рассчитаы с использованием специальных химических фреймворков, таких как RDKit. Это дает возможность модели не дублировать уже имеющуюся информацию в выучиваемом представлении и извлекать признаки более устойчиво по отношении к шуму в данных. В ChemProp предлагается добавлять аналогичные признаки на самом последнем шаге, т.е делать конкатенацию представления MPNN сети и посчитанных дескрипторов, которые отправляются в полносвязную сеть для финального предсказания модели. Пусть h_f вектор глобальных признаков, тогда финальный шаг выглядит следующим образом:

$$\hat{y} = f(cat(h, h_f))$$

, где f - полносвязная нейронная сеть.

Однако, можно добавлять не только глобальные признаки молекулы, но и в целом потенциально полезные дескрипторы. В случае если имеется экспертная информация о полезности конкретных дескрипторов или фингерпринтов, то мы можем создать гибридную модель, которая будет дополнять финальное представление полезой информацией и делать модель более выразительной.

Другим приемом в обучении подобных сетей является добавления виртуальной вершины, или master-node. Идея состоит в добавлении фиктивной вершины соединенной со всеми остальными вершинами графа. На каждом шаге данная вершина будет агрегировать информацию со всех вершин графа и распространять ее обратно. Это позволяет распространять информацию между всеми вершинами за одну итерацию, что позволяет решать вторую озвученную проблему. Однако, это может усугубить проблему с зацикливанием информации о которой говорилось в предыдущем пункте.

Чисто технический трюк для улучшения сходимости глубоких сетей - введение skip-connection на шаге обновления скрытых состояний. Это стандартный прием в больший нейросетевых архитектурых компьютерного зрения и обработки естественного языка, который способствует лучшему протеканию градиента.

В итоге получим улучшенную формулу для архитектуры ChemProp:

$$h_{vw}^{t+1} = h_{vw}^{t} + ReLU(h_{vw}^{0} + W_m m_{vw}^{t+1}), t \in 1,..,T$$

вместо

$$h_{vw}^{t+1} = h_{vw}^{t} + ReLU(h_{vw}^{0} + W_m m_{vw}^{t+1}), t \in 1,..,T$$

3.4 Вывод

[**?**] В данной главе были рассмотрены 2 архитектуры графовых нейронных сетей созданных специально для работы с молекулярными соединениями. Первая архитектура является развитием MPNN с идеей перехода от обновления скрытых состояний вершин к обновлению скрытых состояний ребер. Вторая архитектура является адаптации архитектур трансформеров из области обработки естественного языка. Она также использует MPNN фреймворк, но при этом способна обучаться в self-supervised режиме, что дает возможность использовать большие неразмеченные датасеты на этапе предобучения модели. Это повышает обобщающую и выразительную способность модели, что позволило ей достичь на момент выхода статьи самых лучших результатов в задаче предсказания молекулярных свойств на большим числе публичных датасетов. Также исследованы некоторые практические проблемы, которые возникают при обучении подобных сетей и предложены варианты их решения: построения гибридных моделей и введения фиктивных вершин.

Глава 4. Вычислительные эксперименты

4.1 Используемые выборки

Будем рассматривать тестировать наши модели на 16 общедоступных наборах данных из [30]. Эти наборы данных различаются по размеру. от менее 200 молекул до более 450 000 молекул. Они содержат широкий спектр как регрессионных, так и классификационных задача, в том числе из областей квантовой механики, физической химии, биофизики и физиологии. В таблице 4.1 представлено описание целевых переменных в разных датасетах.

Таблица 4.1 — Описание задач в представленных датасетах

Датасет	Область	Предсказываемое свойство
QM7,QM8,QM9	квантовая механика	компьютерные квантово-механические свойства
ESOL	физическая химия	растворимость в воде
FreeSolv	физическая химия	свободная энергия гидратации в воде
Lipophilicity	физико-химия	коэффициенты распределения октанол/вода
PCBA	биофизика	различные биологические тесты
MUV	биофизика	различные биологические тесты
HIV	биофизика	ингибирование репликации ВИЧ
BACE	биофизика	ингибирование β-секретазы 1 человека
BBBP	физиология	способность проникать через гематоэнцефалический барьер
Tox21	физиология	токсичность
ToxCast	физиология	токсичность
SIDER	физиология	побочное действие лекарств
ClinTox	физиология	токсичность
ChEMBL	физиология	биологические анализы

Для оценки качества моделей используется определенная функция оценки качества, которая называется - метрика. Каждый из датасетов имеет свою

определенную метрику, которая подобрана под тип задачи и распределение целевой переменной. Стоит отметить, что метрика может отличаться от функции потерь использующейся при обучении. Функция потерь должна обладать свойством дифференцируемости, которое часто отсутствует в метриках качества задачи классификации. Поскольку метрики различаются для разных датасетов, мы дополнительно введем одну общую метрику качества для задач регрессии - коэффициент детерминации r2. Некоторые датасеты имеют размерность целевых переменных > 1. Это значит, что нам необходимо строить модели которые будут предсказывать несколько свойств на одном и том же наборе данных. В качестве финального качества полученных моделей будем рассматривать среднюю оценку качества по всем представленным в датасете задачам. Важно понимать, что в случае мультизадачных наборов данных будет происходить существенное различие в обучение дескрипторных и нейросетевых моделей. Поскольку векторное описание в случае дескрипторных моделей детерминировано, то под каждую из задач будет обучаться отдельная модель. В случае нейросетевых подходов векторное описание извлекаемое моделью строится одновременно для всех задач. Это дает возможность извлекать полезную информацию из других целевых переменных для построения более универсального и выразительного представления. В таблице 4.2 уканазно число задач и целевая метрика для каждого из датасетов. Про метрики классификации ROC-AUC, PRC-AUC можно почитать в [54].

4.2 Эксперименты по обучению GNN и дескрипторных моделей

В качестве GNN архитектуры для наших экспериментов возьмем ChemProp, как одну из лучших чисто MPNN архитектур. Обучим для каждого набора данных модель с использованием готового фреймворка [51]. Также, сравним результаты модели $Grover$ предложенные в исходной статье с полученными результатами для дескрипторных. Поскольку основная часть датасетов не велика, будем оценивать качество модели с помощью стратегии кросс-валидации с числом разбиений - 5. Использование GPU ускорителей существенно уменьшает время для обучения сетей. Хотя для небольших датасет (< 1000 элементов) время обучения на CPU также остается малым (порядка нескольких минут).

Таблица 4.2 — Описание датасетов

Датасет	Число задач	Тип задачи	Число соединений	Метрика
QM7	1	regression	6830	MAE
QM8	12	regression	21786	MAE
QM9	12	regression	133885	MAE
ESOL	1	regression	1128	RMSE
FreeSolv	1	regression	642	RMSE
Lipophilicity	1	regression	4200	RMSE
PCBA	128	classification	437929	PRC-AUC
MUV	17	classification	93087	PRC-AUC
HIV	1	classification	41127	ROC-AUC
BACE	1	classification	1513	ROC-AUC
BBBP	1	classification	2039	ROC-AUC
Tox21	12	classification	7831	ROC-AUC
ToxCast	617	classification	8576	ROC-AUC
SIDER	27	classification	1427	ROC-AUC
ClinTox	2	classification	1478	ROC-AUC
ChEMBL	1310	classification	456331	ROC-AUC

В нотации фреймворка Pytorch [55] архитектура модели будет выглядеть следующим образом:

Листинг 4.1: Pytorch version of ChemProp for QM9 Dataset

```
MoleculeModel(
  (encoder): MPN(
    (encoder): ModuleList(
      (0): MPNEncoder(
        (dropout_layer): Dropout(p=0.0, inplace=False)
        (act_func): ReLU()
        (W_i): Linear(in_features=147, out_features=300, bias=
    False)
        (W_h): Linear(in_features=300, out_features=300, bias=
    False)
        (W_o): Linear(in_features=433, out_features=300, bias=
    True)
      )
    )
  )
```

```
  (ffn): Sequential(
    (0): Dropout(p=0.0, inplace=False)
    (1): Linear(in_features=500, out_features=300, bias=True)
    (2): ReLU()
    (3): Dropout(p=0.0, inplace=False)
    (4): Linear(in_features=300, out_features=12, bias=True)
  )
)
```

Суммарное количество параметров моделей насчитывается от 350 - 450 тыс. Их количество зависит от размера скрытого слоя (векторного представления, которое выучивает наша модель), количества целевых переменных, размерности признаковых описаний атомов/связей и молекул.

В случае молекулярных датасетов случайное разбиение объектов на обучающее и тестовое множество может быть некорректным. Поскольку некоторые молекулы имеют сильно схожую структуру, они могут обладать сильно коррелирующими значениями целевых переменных. Нахождение подобных молекул в train и test части датасета приводит к тому, что мы тестируем нашу модель практически на тех же данных, на которых эта модель училась. Такая стратегия приводит к завышенным оценкам качества. Если мы имеем специфическую задачу, с заведомо известным подмножеством молекул (с ограниченным классом структур) случайная стратегия разбиения может быть оправданной. Однако, если мы решаем максимально общую задачу с целью обобщения модели на предельно большой класс молекул, то такая стратегия приведет к ложно завышенным оценкам качества, ведь среди практически бесконечного множества произвольных молекул в нашей тренировочной выборке присутствует лишь маленькая часть возможных структур. Соответсвенно измерять качество нужно аналогично, т.е тестировать модель на структурах, которые модель не видела во время обучения. Для этого можно использовать стратегии разбиения по каркасам (scaffolds spliting). Одним из вариантов является подсчет каркаса Бемиса-Мурко [56], который сохраняет информацию о порядке связей и хиральности. Молекулы с одинаковым каркасом попадают в одну и ту же тренировочную/тестовую группу. Эксперименты с данной стратегией разбиения будут описаны в пункте 4.3.

Для дескрипторных моделей необходимо подготовить посчитанные дескрипторы из которых в дальнейшем будут сформированы обучающие матрицы. Подсчет дескрипторов происходит аналогично процедуре описаной в главе 1.1.

На полученных матрица обучается линейная модель (Ridge-регрессия) с перебором параметра регуляризации C. Качество моделей аналогичным образом оценивается с помощью стратегии кросс-валидации. Разбиение для кросс-валидации сохраняется постоянными для конкретной выборки в рамках разных экспериментов. Это позволяет получать сравнимые оценки качества.

Также, проведем эксперимент по гибридизации модели. Как описывалось в пункте 3.3 сильной стороной GNN является возможность добавления в нее произвольных признаков как на уровне атомов/связей, так и на уровне молекул. Поэтому мы можем добавить к финальному представлению полученному от GNN дескрипторы классических моделей и использовать смешанное представление для финального предсказания. Это может дать понимание полезности разных представлений друг для друга и тому как они взаимодействуют.

4.3 Сравнение моделей

Рассмотрим отдельно эксперименты на регрессионных и классификционных датасетах. В столбцах *Дескр. модели* указаны оценки качества для классических моделей на топологических дескрипторах. В столбце *GNN* указаны значения графовых нейросетевых моделей. В столбце *Разница* указаны разницы полученных метрик (из столбца *Дескр. модели* вычитается столбец *GNN*).

4.3.1 Регрессионные датасеты

В случае регрессионных датасетов, значения метрик являются значениями некоторых функций ошибки. Это значит, что в представленных результатах значение в столбце *Метрика* необходимо минимизировать. В то же время, общая метрика $r2$ является функцией оценки качества, а значит что ее значение необходимо максимизировать.

Рассмотрим сравнение моделей на регрессионных датасетах при случайной стратегии разбиения кросс-валидации.

В таблице 4.3 видно, что на 5 из 6 датасетов качество GNN моделей выше, чем у дескрипторных моделей. Исключением является лишь один датасет $QM7$. При этом для некоторых датасетов качество дескрипторных моделей предельно

Таблица 4.3 — Сравнение моделей (случайное разбиение)

Датасет	Длина	Дескр. модели		GNN		Разница	
		Метрика	r2	Метрика	r2	Метрика	r2
ESOL	1128	0.865	0.826	0.687	0.888	0.18	-0.06
FreeSolv	642	1.385	0.865	1.171	0.9	0.21	-0.04
Lipophilicity	4200	0.953	0.371	0.565	0.781	0.39	-0.41
QM7	6834	59.312	0.753	70.022	0.708	-10.71	0.05
QM8	21786	0.022	0.541	0.011	0.847	0.01	-0.31
QM9	133885	8.656	0.772	3.838	0.925	4.82	-0.15

плохое: для Lipophilicity - 0.371, QM8 - 0.541 значение $r2$. На этих датасетах GNN показывают значительно лучшие результаты: для Lipophilicity - 0.781, QM8 - 0.847 значение $r2$. В случае мультизадачного датасета QM8 дескрипторная модель имеет плохие результаты из-за того, что некоторые задачи решаются предельно плохо. Конкретные цифры можно посмотреть в таблице A.1. В то же время, GNN модель в среднем решает задачи намного лучше. Это следствие мультизадачного подхода к обучению, которое доступно для нейросетевых архитектур. Обучая одну модель сразу на нескольких целевых переменных мы строим более универсальное и устойчивое представление, что позволяет нам получать более высокое качество модели.

Относительно влияния размера выборки можно заметить, что чем больше размер выборки, тем выше разница в качестве между GNN моделями и дескрипторными моделями. Хотя даже на самых маленьких выборках (ESOL, FreeSolv) GNN модели показывают качество выше. Остро стоит вопрос применимости нейросетевых архитектур с большим число параметров к маленьким выборкам, однако эксперименты показывают, что даже на маленьких выборках можно успешно обучать графовые нейросетевые модели с сотнями тысяч параметров.

Рассмотрим как влияет стратегия разбиения на train/test часть для разного класса моделей. Сравним стратегию случайного разбиения и разбиения по каркасам, описанное в прошлом пункте (scaffold splitting). Для GNN моделей результаты приведены в таблице 4.4. Из таблицы видно, что во всех случаях качество модели падает. Это не следствие ухудшения самой модели, это следствие изменения стратегии обучения и тестирования модели. В случае датасета $QM7$ заметно значительное снижение качества. Это значит, что для данного датасета структурные особенности молекул сильно влияют на целевую переменную. При

этом наша модель плохо обобщается на новые каркасы. Также можно заметить, что для более маленьких датасетов падение качества в среднем выше, чем для больших. Это логично объясняется более высокой обобщающей способностью модели обученной на большем количестве данных.

Таблица 4.4 — Сравнение GNN моделей при scaffold разбиении

Датасет	Случайное рабиение		Scaffold разбиение		Разница	
	Метрика	r2	Метрика	r2	Метрика	r2
ESOL	0.687	0.888	0.979	0.755	-0.29	0.13
FreeSolv	1.171	0.9	2.356	0.744	-1.18	0.16
Lipophilicity	0.565	0.781	0.655	0.706	-0.09	0.08
QM7	70.022	0.708	102.844	0.44	-32.82	0.27
QM8	0.011	0.847	0.016	0.76	-0.01	0.09
QM9	3.838	0.925	4.514	0.921	-0.68	0.04

Для дескрипторных моделей результаты приведены в таблице 4.5. В этом случае наблюдаются аналогичные результаты. В том числе отсительно $QM7$ происходит такое же сильное падение качества, что говорит о том, что это в большей степени особенность набора данных, а не конкретной модели.

Таблица 4.5 — Сравнение дескрипторных моделей при scaffold разбиении

Датасет	Длина	Случайное рабиение		Scaffold разбиение		Разница	
		Метрика	r2	Метрика	r2	Метрика	r2
ESOL	1128	0.865	0.826	1.181	0.699	-0.32	0.13
FreeSolv	642	1.385	0.865	7.212	0.67	-5.83	0.19
Lipophilicity	4200	0.953	0.371	0.946	0.357	0.01	0.01
QM7	6834	59.312	0.753	72.332	0.577	-13.02	0.18
QM8	21786	0.022	0.541	0.025	0.529	-0.0	0.01
QM9	133885	8.656	0.772	8.444	0.804	0.21	-0.03

Сравним качество моделей обученных со scaffold стратегией разбиения. Результаты приведены в таблице 4.6 и похожи на базовый эксперимент со случайным разбиением. Однако, при таком способе оценки на 4 из 6 датасетов разница в качестве уменьшается.

Теперь посмотрим как ведут себя гибридные GNN модели. Проведем эксперименты по добавлению топологических дескрипторов и 200 глобальных

Таблица 4.6 — Сравнение моделей (scaffold разбиение)

Датасет	Дескр. модели		GNN		Разница	
	Метрика	r2	Метрика	r2	Метрика	r2
ESOL	1.181	0.699	0.979	0.755	0.2	-0.06
FreeSolv	7.212	0.67	2.356	0.744	4.86	-0.07
Lipophilicity	0.946	0.357	0.655	0.706	0.29	-0.35
QM7	72.332	0.577	102.844	0.44	-30.51	0.14
QM8	0.025	0.529	0.016	0.76	0.01	-0.23
QM9	8.444	0.804	4.514	0.921	3.93	-0.12

дескрипторов посчитанных с помощью RdKit к GNN. Результаты приведены в таблице 4.7 и представляют собой разностькачества между новыми гибридными моделями и исходной GNN моделью. Из таблицы видно, что добавлению топологических дескрипторов модели ухудшает качества по сравнению с исходной моделью. Это объясняется локальностью топологических дескрипторов. По сути данная информация уже содержится в векторном представлении, которые мы получаем с помощью GNN, т.к $MPNN$ занимаются агрегацией информацией между атомами/связями и финальное представление основано на наборе локальных представлений компонент графа. Добавление топологических дескрипторов вносит шум в представление и не обогощает его. С глобальными дескрипторами наблюдается противоположная ситуация. В этом случае добавление дескрипторов улучшает качество модели. Глобальная дескрипторы - это то, что GNN модель не выучивает явно. Чтобы модель не занималась бесполезной работой по неявному выучиванию каких то известных и доступных глобальных характеристик молекул, мы добавляем их в финальное представление, тем самым подталкивая модель выучивать новые свойства, которые полезны в совокупности с уже существующими.

Таблица 4.7 — Гибридные GNN модели

Датасет	GNN + Тополог. дескр.		GNN + глобал. декср.	
	Метрика	r2	Метрика	r2
ESOL	0.091	-0.031	-0.107	0.031
FreeSolv	0.285	-0.069	-0.17	0.024
Lipophilicity	0.145	-0.126	-0.005	0.004
QM7	-6.931	0.042	-6.284	0.049
QM8	0.001	-0.008	0.0	0.003
QM9	-1.087	-0.049	-0.625	0.011

4.3.2 Классификационные датасеты

В случае классификационных датасетов, значения метрик являются значениями функции оценки качества. Это значит, что в представленных результатах значение в столбце *Метрика* необходимо максимизировать.

Рассмотрим сравнение моделей на классификационных датасетах при случайной стратегии разбиения кросс-валидации. Результаты приведены в таблице 4.8. В случае задачи классификации значительное преимущество наблюдается на всех представленных датасетах. При этом модели на топологических дескрипторах показывают качество превосходящее случайный классификатор только на 2 из 6 датасетах. В остальных случаях качество незначительно превосходит модель со случайными предсказаниями. Это пример того, что фиксированное представление на зафиксировнном наборе дескрипторов может быть не информативно для решения конкретной задачи. Для первых 2х датасетов модель смогла использовать топологические дескрипторы, однако в остальных случаях эта информация бесполезна для предсказания целевой переменной. В случае же GNN моделей на всех датасетах достигается неплохое качество ($>$ 0.8), кроме датасета SIDER. В отличие от дескрипторных моделей, нейросетевые модели смогли извлечь нужные признаки из широкого спектра информации, который мы ей предоставили. В данном случае уверенное превосходство графовых нейросетевых моделей наблюдается на всех датасетах, в том числе на датасетах малого объема.

Таблица 4.8 — Сравнение моделей (случайное разбиение)

Датасет	Длина	Дескр. модели	GNN	Разница
		Метрика	Метрика	Метрика
BACE	1513	0.753	0.88	-0.127
BBBP	2039	0.762	0.902	-0.14
ClinTox	1478	0.535	0.895	-0.36
HIV	41127	0.572	0.827	-0.255
SIDER	1427	0.548	0.638	-0.09
Tox21	7831	0.572	0.843	-0.271

4.4 Вывод

В данной главе были проведены эксперименты для сравнения классических моделей на топологических дескрипторах и современных GNN моделей для задачи предсказания молекулярных свойств. Для этого был собран набор публичных датасетов различных размеров и типов задач. В ходе экспериментов было выявлено существенное превосходства нейросетевых графовых моделей, по сравнению с классическими моделями. В случае регрессионных датасетов, примущество наблюдается на 5 из 6 наборах данных. В случае классификационных датасетов, примущество наблюдается на всех представленных наборах данных. При этом даже на маленьких наборах данных можно обучать графовые модели и получить хорошее качество. Помимо этого было проведено сравнение стратегий разбиение выборки на этапе обучения и тестирования в ходе которого выяснилось, что случайное разбиение выборки на train и test может показывать завышенные оценки качества. Также были рассмотрены гибридные GNN модели, которые могут превосходить исходные GNN при добавлении правильного набора дескрипторов в финальное представление.

Заключение

Задача предсказания молекулярных свойств химических соединений является очень актуальной в области биоинформатике. В данной работе предлагалось сравнить методы основанные на классическом машинном обучении с современными нейросетевыми подходами, а именно графовыми нейронными сетями.

Интерес к графовым нейронным сетям обусловлен тем, что данный тип моделей способен напрямую работать со сложными структурными объектами, такими как молекулярные соединения. Построение векторного представления геометрических объектов может быть сложной задачей, однако использование нейросетевых подходов освобождает нас от этого этапа, т.к подобные модели способны самостоятельно извлекать признаки из объектов и изучать векторное представление оптимальное для каждой конкретной задачи.

Для решение поставленной задачи было сделанно следующее:

1. Были исследованы основные классические подходы QSAR - моделирования, как использующие методы машинного обучения, так и нет. Существенно можно разделить методы на 2 группы: модели использующие фиксированное представление на основе различных дескрипторов (в том числе топологических) и модели строющие векторные описания из исходного представления молекулы. Ко второму типу моделей относятся графовые нейронные сети (GNN).

2. Изучены и описаны архитектуры графовых неронных сетей, а также общий процесс их разработки. В том числе описан общий фреймворк обобщающий широкий класс GNN архитектур - Massage Passing Neural Network (MPNN).

3. Исследованы несколько современных GNN архитектур разработанные специально для задачи предсказания молекулярных свойств химических соединений. Описан вариант обучения графовых нейронных сетей в режиме self-supervised, который позволяет нам использовать большие неразмеченные выборки для обучения моделей и получать state-of-the art результаты. В конце главы рассматриваются проблемы обучения GNN моделей, а также предлгаются варианты их решения, в том числе использование гибридных моделей.

4. Проведены вычислительные эксперименты по обучению GNN моделей и моделей на топологических дескрипторах. Проведен сравнительный анализ полученных моделей, в ходе которого были получены следующие результа-

ты: GNN модели показывает превосходство на большинстве наборов данных разных размеров и типов задач. Также, обучены гибридные варианты GNN моделей, которые могут превосходить исходные GNN модели, при правильном выборе дескрипторов.

Исходя их полученных результатов можно сделать вывод, что использование графовых нейронных сетей в данной задаче является очень перспективным направлением. Они являются очень гибким и выразительным классом моделей, который обладает целом рядом преимуществ. С помощью таких моделей можно:

1. строить векторное представление молекулярных соединений для каждой конкретной задачи, извлекая самые нужные признаки.
2. получать представление молекулярных соединений с информацией о топологии молекулярного графа
3. использовать различные признаки атомов, вершин, самой молекулы для построения описания с возможностью легко масштабировать количество признаков любой из компонент.
4. строить универсальные представления для нескольких целевых переменных на одном датасете.
5. использовать стратегии предобучения для переноса знаний с больших неразмеченных датасетов на другие наборы данных.
6. строить гибридные модели, добавляя в модель любую экспертную информацию из области задачи.
7. использовать обученные векторные представления в других моделях машинного обучения

К недостаткам данных моделей можно отнести большое количество вычислительных ресурсов для обучения с нуля больших моделей. Однако, это касается предобучения моделей на многомиллионных корпусах данных, что часто не является финальной целью обучения. Чаще всего мы хотим использовать хорошую модель, для решения конкретной задачи. Поэтому можно взять обученные модели, которые находятся в открытом доступе и использовать их для дообучения под свою задачу. Данная процедура не такая ресурсоемкая и доступна практически любому исследователю.

В дальнейших работах хотелось бы подробнее рассмотреть другие нейросетевые модели, использующие другие способы представления молекулярных соединений, например, модели использующие SMILES строки, а также варианты их взаимодействия с GNN моделями. Другой областью исследования можеть

быть изучение моделей способных к непосредственной генерации молекулярных соединений, например, GAN или VAE вариаций графовых нейронных сетей.

Список сокращений и условных обозначений

QSAR - Quantitative structure–activity relationship
NN - Neural network
GNN - Graph Neural Network
MPNN - Massege Passing Neural Network
GIN - Graph Isomorphism Network
MLP - Multi-layer perceptron
GCN - Graph Convolution Network
GAT - Graph Attention Network
MAE - Mean average error
MSE - Mean squared error
RMSE - Root mean squared error
AUC - Area under the curve

Список литературы

[1] Le T., Epa V. Ch., Burden F.R., Winkler D.A. Quantative Structure - Property Relationship Modeling of Diverse Materials Properties. // Chemical Reviews. 2012. V. 112. P. 2889-2919.

[2] Leach A.R., Gillet V.J. An Introduction to Chemoinformatics. Springer. 2007. P. 1-255. .

[3] Verma J., Khedkar V.M., Coutinho E.C. 3D-QSAR in Drug Design - A Review. // Current Topics in Medicinal Chemistry. 2010. V.10. P. 95-115. .

[4] Gasteiger J., Engel T. (Eds.). Chemoinformatics. Wiley-VCH. 2006. .

[5] 3D QSAR in Drug Design: Vol. 1. Theory, Methods and Applications (Three-Dimensional Quantitative Structure Activity Relationships). (Ed. by H. Kubinyi). Kluwer/Escom, Dordrecht. 2000. .

[6] 3D QSAR in Drug Design. Vol. 2. Ligand-Protein Complexes and Molecular Similarity. (Ed. by H. Kubinyi, G. Folkers, and Y.C. Martin). Kluwer Academic Publishers, Dordrecht. 2002. .

[7] 3D QSAR in Drug Design. Vol. 3. Recent Advances. (Ed. by H. Kubinyi, G. Folkers, and Y.C. Martin). Kluver Academic Publishers, Dordrecht. 2002.

[8] Katritzky A.R., Kuanar M., Slavov S., Hall C.D. Quantitative Correlation of Physical and Chemical Properties with Chemical Structure: Utility for Prediction // Chem. Rev. 2010. V.110, P.5714-5789.

[9] Ferreira, M. M. C. Multivariate QSAR. J. Braz. Chem. Soc. 2002, 13(6), 742–753

[10] Karelson M. Molecular Descriptors in QSAR/QSPR. John Wiley & Sons: New York. 2000.

[11] Todeschini R., Consonni V. Handbook of molecular descriptors. Wiley-VCH, Weinheim. 2000.

[12] Basak S.C., Gute B.D., Grunwald G.D. Use of topostructural, topochemical, and geometric parameters in the prediction of vapor pressure: A hierarchical QSAR approach. // J. Chem. Inf. Comput. Sci. 1997. V.37. P.651-655.

[13] Baskin I., Varnek A. Fragment Descriptors in SAR/QSAR/QSPR Studies, Molecular Similarity Analysis and in Virtual Screening. In: Chemoinformatic Approaches to Virtual Screening. (Ed. by A. Varnek, A. Tropsha). RCS Publishing. 2008.

[14] Hansch C., Maloney P.P., Fujita T., Muir M. Nature, 1962, v. 194, p.178-180.

[15] Free S.M. Wilson J.M. J. Med. Chem, 1964, v.7, №7, p.395-399

[16] Hoerl A.E., Kennard R.W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. // Technometrics. 1970. V.12 (1). P.55-67.

[17] Гальберштам Н.М., Баскин И.И., Палюлин В.А., Зефиров Н.С. Нейронные сети как метод поиска зависимостей структура - свойство органических соединений // Успехи химии, 2003, Т. 72, №7, 706-727. .

[18] Baskin I.I., Palyulin V.A., Zefirov N.S. Neural networks in building QSAR models. // Methods Mol. Biol. 2008. V.458. P.137-158. .

[19] Cover T.M., Hart P.E. Nearest neighbor pattern classification. // IEEE Transactions on Information Theory. 1967. V.13 (1). P.21-27.

[20] Molecular Interaction Fields: Applications in Drug Discovery and ADME Prediction. (Ed. by G. Cruciani, R. Mannhold, H. Kubinyi, G. Folkers). WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. 2006.

[21] Cramer R.D., Patterson D.E., Bunce J.D. Comparative molecular field analysis (CoMFA). 1. Effect of shape on binding of steroids to carrier proteins. // J. Am. Chem. Soc. 1988. V.110. P.5959-5967.

[22] Goodford P.J. A computational procedure for determining energetically favorable binding sites on biologically important macromolecules. // J. Med. Chem. 1985. V.28. P.849-857.

[23] Davis A.M., Gensmantel N.P., Johansson E., Marriott D.P. The use of the grid program in the 3-D QSAR analysis of a series of calcium channel agonists. // J. Med. Chem. 1994. V.37. P.963-972. .

[24] Klebe G., Abraham U., Mietzner T. Molecular similarity indexes in a comparative analysis (CoMSIA) of drug molecules to correlate and predict their biological activity. // J. Med. Chem. 1994. V.37. P.4130-4146. .

[25] Klebe G., Abraham U. Comparative molecular similarity index analysis (CoMSIA) to study hydrogen-bonding properties and to score combinatorial libraries. // Journal of Computer-Aided Molecular Design. 1999. V.13. P.1-10. .

[26] Bohm M., Sturzebecher J., Klebe G. Three-dimensional quantitative structure–activity relationship analyses using comparative molecular field analysis and comparative molecular similarity indices analysis to elucidate selectivity differences of inhibitors binding to trypsin, thrombin, and factor Xa. // J. Med. Chem. 1999. V.42. P.458-477.

[27] Scarselli F., Gori M., Tsoi A., Hagenbuchner M., Monfardini G. The graph neural network model // IEEE Transactions on Neural Networks, т. 20, № 1, 2009. Сс. 61-80. URL: https://persagen.com/files/misc/scarselli2009graph.pdf. (Дата обращения: 05.12.2021).

[28] Kipf T.N., Welling M. Semi-supervised classification with graph convolutional networks, 2017. URL: https://arxiv.org/abs/1609.02907. (Дата обращения: 05.12.2021).

[29] Yang K., Swanson K., Wengong J., Coley C. Analyzing Learned Molecular Representations for Property Prediction // Journal of Chemical Information and Modeling 2019 59 (8), 3370-3388

[30] Wu et al, "MoleculeNet: A Benchmark for Molecular Machine Learning".

[31] Duvenaud D.K., Maclaurin D., Aguileraiparraguirre J., Gomezbombarelli R., Hirzel T.D., Aspuruguzik A., Adams R.P. Convolutional networks on graphs for learning molecular fingerprints, 2015. URL: https://arxiv.org/abs/1509.09292. (Дата обращения: 05.12.2021).

[32] Kearnes, S., McCloskey, K., Berndl, M., Pande, V., Riley, P., 2016. Molecular graph convolutions: moving beyond fingerprints. J. Comput. Aided Mol. Des. 30, 595–608.

[33] Do, K., Tran, T., Venkatesh, S., 2019. Graph transformation policy network for chemical reaction prediction. Proceedings of SIGKDD 750–760.

[34] Fout, A., Byrd, J., Shariat, B., Ben-Hur, A., 2017. Protein interface prediction using graph convolutional networks. In: Proceedings of NIPS, pp. 6533–6542.

[35] Xu, N., Wang, P., Chen, L., Tao, J., Zhao, J., Gnn, M.R.-, 2019d. multi–resolution and dual graph neural network for predicting structured entity interactions. In: Proceedings of IJCAI, pp. 3968–3974.

[36] Gilmer J., Schoenholz S.S., Riley P.F., Vinyals O., Dahl G.E. Neural message passing for quantum chemistry // International conference on machine learning, 2017. Cc. 1263–1272. URL: https://arxiv.org/abs/1704.01212 (Дата обращения: 05.12.2021).

[37] Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks?. ICLR 2019

[38] Кумсков М.И., Рыбакова Е.О., Ермолаев Н.Д., Анализ и предобработка обучающих выборок молекулярных графов для формирования архитектур RBF нейронных сетей. 2020.

[39] Кумсков М.И., Ермолаев Н.Д., Хритова Е.А., Исследование различных вариантов выбора метрики молекулярных графов для RBF нейронной сети в задаче «структура-свойство. 2020.

[40] Atz, K., Grisoni, F., & Schneider, G. (2021). Geometric Deep Learning on Molecular Representations. ArXiv, abs/2107.12375.

[41] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. & Vandergheynst, P. Geometric deep learning: going beyond euclidean data. IEEE Signal Processing Magazine 34, 18–42 (2017).

[42] Bronstein, M. M., Bruna, J., Cohen, T. & Velickovic, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. arXiv:2104.13478 (2021).

[43] Velickovic P., Cucurull G., Casanova A., Romero A., Lio P., Bengio Y. Graph attention networks, 2018. URL: https://arxiv.org/abs/1710.10903. (Дата обращения: 05.12.2021).

[44] Hamilton W.L., Ying Z., Leskovec J. Inductive representation learning on large graphs, 2017. Cc. 1024–1034. URL: https://arxiv.org/pdf/1706.02216.pdf. (Дата обращения: 05.12.2021).

[45] Hammond D.K., Vandergheynst P., Gribonval R. Wavelets on graphs via spectral graph theory, 2009. Cc. 129–150. URL: https://arxiv.org/pdf/0912.3848.pdf. (Дата обращения: 05.12.2021).

[46] Shuman D.I., Narang S.K., Frossard P., Ortega A., Vandergheynst P. The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains // IEEE SPM, 30, 2013. Cc. 83–98. URL: https://arxiv.org/abs/1211.0053. (Дата обращения: 05.12.2021).

[47] Wu Z., Pan S., Chen F., Long G., Zhang C., Yu P.S. A Comprehensive Survey on Graph Neural Networks, 2019. URL: https://arxiv.org/abs/1901.00596. (Дата обращения: 05.12.2021).

[48] Zhou J., Cui G., Zhang Z., Yang C., Liu Z., Sun M. Graph Neural Networks: A Review of Methods and Applications, 2018. URL: https://arxiv.org/abs/1812.08434. (Дата обращения: 05.12.2021).

[49] Rong, Y., Bian, Y., Xu, T., Xie, W., Wei, Y., Huang, W., & Huang, J. (2020). Self-Supervised Graph Transformer on Large-Scale Molecular Data. arXiv: Biomolecules.

[50] Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. ArXiv, abs/1706.03762.

[51] https://github.com/chemprop/chemprop

[52] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Pre-training graph neural networks. arXiv preprint arXiv:1905.12265, 2019.

[53] Shengchao Liu, Mehmet F Demirel, and Yingyu Liang. N-gram graph: Simple unsupervised representation for graphs, with applications to molecules. In Advances in Neural Information Processing Systems, pages 8464–8476, 2019.

[54] https://habr.com/ru/company/ods/blog/328372/

[55] https://pytorch.org/

[56] Bemis, G. W.; Murcko, M. A. J. Med. Chem. 1996, **39**, 2887-2893

Список рисунков

Приложение А

Результаты вычислительных экспериментов

Таблица А.1 — Полное качество дескрипторных моделей (регрессионные датасеты)

Датасет_задача	Размер датасета	Значение метрики	r2	Метрика
QM7_task_0	6834	59.312	0.753	MAE
QM8_task_0	21786	0.015	0.805	MAE
QM8_task_1	21786	0.012	0.776	MAE
QM8_task_2	21786	0.023	0.345	MAE
QM8_task_3	21786	0.042	0.161	MAE
QM8_task_4	21786	0.015	0.831	MAE
QM8_task_5	21786	0.013	0.831	MAE
QM8_task_6	21786	0.021	0.387	MAE
QM8_task_7	21786	0.034	0.139	MAE
QM8_task_8	21786	0.015	0.817	MAE
QM8_task_9	21786	0.012	0.818	MAE
QM8_task_10	21786	0.025	0.404	MAE
QM8_task_11	21786	0.038	0.174	MAE
QM9_task_0	133885	0.221	0.372	MAE
QM9_task_1	133885	0.126	0.386	MAE
QM9_task_2	133885	0.863	0.389	MAE
QM9_task_3	133885	0.98	0.969	MAE
QM9_task_4	133885	0.009	0.708	MAE
QM9_task_5	133885	0.012	0.883	MAE
QM9_task_6	133885	0.016	0.806	MAE
QM9_task_7	133885	92.503	0.782	MAE
QM9_task_8	133885	0.001	0.999	MAE
QM9_task_9	133885	0.016	1.0	MAE
QM9_task_10	133885	0.509	0.971	MAE
QM9_task_11	133885	8.614	0.998	MAE
ESOL_task_0	1128	0.865	0.826	RMSE

FreeSolv_task_0	642	1.385	0.865	RMSE
Lipophilicity_task_0	4200	0.953	0.371	RMSE

Таблица A.2 — Полное качество GNN моделей (регрессионные датасеты)

Датасет	Задача	Метрика	r2
ESOL	target_0	0.687	0.888
FreeSolv	target_0	1.171	0.9
Lipophilicity	target_0	0.565	0.781
QM7	task_0	70.022	0.708
QM8	task_0	0.005	0.968
QM8	task_1	0.006	0.929
QM8	task_2	0.012	0.773
QM8	task_3	0.026	0.619
QM8	task_4	0.005	0.974
QM8	task_5	0.006	0.955
QM8	task_6	0.01	0.82
QM8	task_7	0.02	0.675
QM8	task_8	0.005	0.974
QM8	task_9	0.006	0.953
QM8	task_10	0.011	0.833
QM8	task_11	0.021	0.689
QM9	task_0	0.102	0.742
QM9	task_1	0.063	0.77
QM9	task_2	0.447	0.8
QM9	task_3	0.626	0.976
QM9	task_4	0.004	0.938
QM9	task_5	0.004	0.984
QM9	task_6	0.005	0.972
QM9	task_7	30.099	0.971
QM9	task_8	0.001	0.994
QM9	task_9	2.335	0.984
QM9	task_10	0.292	0.983
QM9	task_11	12.078	0.988

Таблица А.3 — Полное качество дескрипторных моделей (классификационные датасеты)

Датасет	Задача	Метрика
ClinTox	task_0	0.536
ClinTox	task_1	0.534
BACE	task_0	0.753
HIV	task_0	0.572
BBBP	task_0	0.762
Tox21	task_0	0.632
Tox21	task_1	0.656
Tox21	task_2	0.604
Tox21	task_3	0.541
Tox21	task_4	0.521
Tox21	task_5	0.543
Tox21	task_6	0.537
Tox21	task_7	0.566
Tox21	task_8	0.518
Tox21	task_9	0.567
Tox21	task_10	0.652
Tox21	task_11	0.532
SIDER	task_0	0.636
SIDER	task_1	0.568
SIDER	task_2	0.488
SIDER	task_3	0.582
SIDER	task_4	0.522
SIDER	task_5	0.526
SIDER	task_6	0.55
SIDER	task_7	0.503
SIDER	task_8	0.509
SIDER	task_9	0.639
SIDER	task_10	0.569
SIDER	task_11	0.527
SIDER	task_12	0.547

SIDER	task_13	0.524
SIDER	task_14	0.53
SIDER	task_15	0.613
SIDER	task_16	0.535
SIDER	task_17	0.516
SIDER	task_18	0.531
SIDER	task_19	0.535
SIDER	task_20	0.557
SIDER	task_21	0.584
SIDER	task_22	0.498
SIDER	task_23	0.583
SIDER	task_24	0.569
SIDER	task_25	0.517
SIDER	task_26	0.531

Таблица A.4 — Полное качество GNN моделей (классификационные датасеты)

Dataset	Task	metric
ClinTox	task_0	0.905
ClinTox	task_1	0.885
BACE	task_0	0.88
HIV	task_0	0.827
BBBP	task_0	0.902
Tox21	task_0	0.815
Tox21	task_1	0.871
Tox21	task_2	0.894
Tox21	task_3	0.849
Tox21	task_4	0.73
Tox21	task_5	0.829
Tox21	task_6	0.863
Tox21	task_7	0.815
Tox21	task_8	0.869
Tox21	task_9	0.812
Tox21	task_10	0.9

Tox21	task_11	0.863
SIDER	task_0	0.699
SIDER	task_1	0.59
SIDER	task_2	0.456
SIDER	task_3	0.645
SIDER	task_4	0.650
SIDER	task_5	0.62
SIDER	task_6	0.721
SIDER	task_7	0.639
SIDER	task_8	0.617
SIDER	task_9	0.677
SIDER	task_10	0.609
SIDER	task_11	0.657
SIDER	task_12	0.659
SIDER	task_13	0.622
SIDER	task_14	0.627
SIDER	task_15	0.68
SIDER	task_16	0.667
SIDER	task_17	0.568
SIDER	task_18	0.614
SIDER	task_19	0.64
SIDER	task_20	0.650
SIDER	task_21	0.652
SIDER	task_22	0.615
SIDER	task_23	0.624
SIDER	task_24	0.661
SIDER	task_25	0.765
SIDER	task_26	0.592