УДК 004.021
DOI: 10.26102/2310-6018/2025.48.1.038

Использование графовых нейронных сетей для решения задачи Штейнера

Д.А. Пиминов✉, В.В. Печенкин, М.С. Королёв

Саратовский государственный технический университет имени Гагарина Ю.А., Саратов, Российская Федерация

Резюме. Теория дискретной оптимизации играет важную роль в решении задач теории графов, таких как задача Штейнера. Она используется в транспортной инфраструктуре, логистике и проектировании коммуникационных сетей. Задача является NP-трудной, что требует применения эвристических методов, таких как генетические алгоритмы и искусственные нейронные сети. Для решения задачи Штейнера была выбрана графовая нейронная сеть (GNN). Архитектура GNN включает итеративное обновление признаков с использованием информации о смежных вершинах, что позволяет моделировать сложные зависимости в графах. Для агрегации информации применяется механизм передачи сообщений (MPNN), который обновляет состояние вершины, учитывая данные смежных вершин и ребер. Обучение модели осуществляется с использованием графов, сгенерированных с помощью эвристического алгоритма Мельхорна. В эксперименте показано, что GNN хорошо работает на графах, схожих с обучающими, но при увеличении размера входных графов метрики точности и полноты результатов значительно снижаются. Это снижение, скорее всего, связано с ограничениями механизма MPNN, который агрегирует информацию о соседних вершинах на ограниченном расстоянии. Графовые нейронные сети показывают высокий потенциал для задач на графах небольшой и средней размерности, особенно в анализе сложных систем, таких как беспроводные сети, где важны взаимосвязи между вершинами. Однако при увеличении размерности наблюдается снижение качества, что требует улучшений в алгоритмах агрегации и оптимизации.

Ключевые слова: задача Штейнера, графовые нейронные сети, теория графов, искусственные нейронные сети, алгоритм Мельхорна.

Для цитирования: Пиминов Д.А., Печенкин В.В., Королёв М.С. Использование графовых нейронных сетей для решения задачи Штейнера. *Моделирование, оптимизация и информационные технологии.* 2025;13(1). URL: https://moitvivt.ru/ru/journal/pdf?id=1828 DOI: 10.26102/2310-6018/2025.48.1.038

Using graph neural networks for solving the Steiner tree problem

D.A. Piminov✉, V.V. Pechenkin, M.S. Korolev

Yuri Gagarin State Technical University of Saratov, Saratov, the Russian Federation

Abstract. The theory of discrete optimization plays a crucial role in solving graph theory problems, such as the Steiner tree problem. It is widely applied in transportation infrastructure, logistics, and communication network design. Since the problem is NP-hard, heuristic methods such as genetic algorithms and artificial neural networks are often required. To solve the Steiner tree problem, a graph neural network (GNN) was selected. The GNN architecture involves iterative feature updates using information from neighboring nodes, allowing it to model complex dependencies in graphs. A message-passing neural network (MPNN) mechanism is employed for information aggregation, updating node states based on data from adjacent nodes and edges. The model is trained on graphs generated using the Melhorn heuristic algorithm. Experiments show that GNN performs well on graphs similar to the training data but experiences a significant drop in precision and recall metrics as the input graph size increases. This decline is likely due to the limitations of the MPNN mechanism, which aggregates

information only from neighboring nodes within a limited range. Graph neural networks demonstrate strong potential for small- and medium-scale graph problems, particularly in analyzing complex systems such as wireless networks, where node interconnections are critical. However, as graph size increases, performance deteriorates, highlighting the need for improvements in aggregation and optimization algorithms.

Keywords: Steiner tree problem, graph neural networks, graph theory, artificial neural networks, Mehlhorn algorithm.

For citation: Piminov D.A., Pechenkin V.V., Korolev M.S. Using graph neural networks for solving the Steiner tree problem. 2025;13(1). (In Russ.). *Modeling, Optimization and Information Technology* URL: https://moitvivt.ru/ru/journal/pdf?id=1828 DOI: 10.26102/2310-6018/2025.48.1.038

Введение

Теория дискретной оптимизации играет ключевую роль в решении задач теории графов и сетей, поскольку многие реальные задачи могут быть представлены в виде дискретных структур, таких как графы. Дискретная оптимизация на графах охватывает широкий спектр задач, направленных на поиск эффективных решений в дискретных структурах. Например, задача Штейнера требует нахождения минимального дерева, соединяющего только заданное подмножество терминальных вершин, что делает ее NP-трудной [1]. Данную задачу можно считать одной из ключевых задач в теории графов и оптимизации, которая заключается в поиске минимальной сети, соединяющей заданное множество точек, называемых терминальными, с возможностью добавления новых точек, которые приводят к сокращению общей стоимости сети и называются точками Штейнера [2].

Задачу Штейнера можно сформулировать как на графах, так и на евклидовой плоскости. В графовой версии задачи задан граф $G = (V,E)$, где V – множество вершин, E – множество ребер с положительными весами. Также задано подмножество терминальных вершин $T \subseteq V$, которые необходимо соединить. Требуется найти подграф $G' = (V',E')$, минимизирующий суммарный вес ребер, соединяющий терминальные вершины, в котором могут использоваться дополнительные вершины из $V \setminus T$ для минимизации веса, при этом подграф G' должен быть деревом [3].

В евклидовой постановке задачи задано конечное множество точек T на плоскости. Требуется построить сеть минимальной длины, соединяющую все точки из T. Для минимизации длины сети разрешается добавлять новые точки, называемые точками Штейнера, при этом также как и в задаче на графах, полученный граф должен являться деревом. В этой задаче используются геометрические свойства, такие как углы в 120 градусов, что позволяет найти оптимальное решение [4].

Решение задачи Штейнера используется при оптимизации транспортной инфраструктуры – одной из ключевых проблем современного градостроительства, логистики и планирования территорий. В условиях роста городов, увеличения транспортного потока и необходимости минимизации затрат, применение математических методов становится особенно важным. Одним из таких методов является задача Штейнера, которая позволяет построить сеть соединений минимальной суммарной длины, соединяющую заданные точки (населенные пункты, промышленные объекты и т. д.) с возможностью добавления промежуточных вершин [5].

В настоящее время разработка моделей и алгоритмов оптимизации коммуникационных сетей становится приоритетной задачей. Сети передачи данных, интернета и телефонии требуют минимизации затрат на строительство и эксплуатацию, одновременно обеспечивая высокую производительность и надежность. Задача Штейнера играет важную роль в проектировании таких сетей, позволяя строить

топологии, которые соединяют заданные узлы с минимальной общей длиной соединений и с возможностью добавления промежуточных точек. В коммуникационных сетях терминальные точки (например, узлы сети или абоненты) должны быть соединены таким образом, чтобы минимизировать затраты на прокладку кабелей или радиосигналы. Задача Штейнера позволяет определить оптимальные места для установки промежуточных узлов, таких как ретрансляторы, серверы или распределительные точки [6]. Это особенно актуально при проектировании сетей в условиях ограничений, таких как гористая местность или высокая плотность застройки.

Задача Штейнера, как говорилось ранее, является NP-трудной, что делает использование точных методов на практике затруднительным для больших графов [1]. Для решения этой задачи на практике часто применяются эвристические алгоритмы, которые позволяют найти приближенные, но достаточно хорошие решения за разумное время.

Метод ветвей и границ – это универсальный подход для решения задач дискретной и глобальной оптимизации, используемый в задачах, где полный перебор вариантов слишком трудоемок, например, в задачах коммивояжера, задаче Штейнера и других [7]. Он основан на рекурсивном разбиении множества решений на подмножества и вычислении границ оптимального значения целевой функции, что позволяет исключать заведомо неоптимальные подмножества [8]. Эффективность сильно зависит от качества используемых оценок и принципиально не изменяет оценки сложности получения решения.

Одним из вариантов решения труднорешаемых задач являются генетические алгоритмы, они предлагают гибкие приближенные решения и эффективны для задач, таких как кластерное дерево Штейнера, где вершины графа разделены на кластеры, и дерево должно учитывать их соединение [9].

Использование искусственных нейронных сетей (ИНС) для решения задачи Штейнера является относительно новым направлением, эффективно работающим с задачами, требующими поиска решения в больших структурах данных. Среди различных архитектур ИНС особый интерес представляют самоорганизующиеся нейронные сети, способные адаптироваться к структуре входных данных и находить оптимальные решения без внешнего контроля. Особенностью данного типа ИНС является способность выявлять скрытые закономерности, кластеризовать данные и эффективно визуализировать многомерные пространства, что делает их полезными для задач, связанных с оптимизацией и анализом графов. Основная идея применения самоорганизующейся сети заключается в том, чтобы адаптировать сеть к оптимальной конфигурации, основываясь на результатах целевой функции, связанной с длиной дерева Штейнера [10]. К недостаткам данного типа нейронных сетей можно отнести то, что точность начинает снижаться с увеличением размерности графа.

В последние годы применение методов глубокого обучения для решения задач комбинаторной оптимизации демонстрирует возрастающую эффективность. Особое место занимают гибридные подходы, объединяющие несколько различных видов нейронных сетей [11]. Используя графовую нейронную сеть и обучение с подкреплением, для решения задачи направленного дерева Штейнера, можно получать хорошие результаты имеющие значительные преимущества в скорости и не уступающие по качеству на средних и больших графах. Для решения более частного случая задачи построения минимального прямоугольного дерева Штайнера на практике могут использоваться различные комбинированные подходы, объединяющие до 5 нейронных сетей. Такой подход может накладывать различные ограничения на размерность входящих графов, но при этом имеет линейную зависимость времени выполнения от количества вершин [12].

Целью данной работы является исследование применимости графовых нейронных сетей для решения задачи Штейнера, включая оценку результатов на графах с различной размерностью и топологией.

Задачи исследования включают: разработку и обучение модели графовой нейронной сети; проведение вычислительного эксперимента для оценки применимости графовых нейронных сетей на графах с различной размерностью и топологией; определение и анализ точности предсказаний разработанной модели нейронной сети на различных графах.

Материалы и методы

Для проведения исследования были выбраны графовые нейронные сети (Graph Neural Networks, GNN). Основным критерием выбора является способность GNN учитывать топологию графа и адаптивно обновлять признаки вершин на основе их связей. Архитектура GNN основана на представлении данных в виде графов $G = (V,E)$, где V – множество вершин, E – множество ребер. Вершины и ребра описывают объекты и их взаимосвязи. Основная идея GNN заключается в итеративном обновлении признаков вершин графа, используя информацию об их смежных вершинах [13].

Каждая анализируемая вершина графа имеет начальное представление, которое инициализируется в виде вектора, хранящего категориальный признак и минимальный вес инцидентного ребра. На каждом слое сети вершины обновляют свои представления, агрегируя информацию от вершин и ребер, находящихся в окрестности данной вершины. Начальную матрицу признаков графа можно представить в следующем виде:

$$\mathrm{H}^{(0)} = \{h_1^{(0)}, h_1^{(0)}, \dots, h_N^{(0)}\}, h_i^{(0)} \in \mathbb{R}^d, \quad (1)$$

где $\mathrm{H}^{(0)}$ – матрица признаков вершин графа на начальном этапе, она включает в себя множество векторов начальных признаков вершины $h_i^{(0)}$, d – количество признаков, N – общее количество вершин в графе.

Обновление признаков для каждой вершины происходит рекуррентно и можно представить как:

$$h_i^{(k+1)} = f_{\text{update}}\left(h_i^{(k)}, \text{AGGREGATE}\left(\{h_i^{(k)} \mid j \in N(i)\}\right)\right), \quad (2)$$

где $h_i^{(k)}$ – признаки вершины i на слое k, $N(i)$ – множество вершин в окрестности вершины i, AGGREGATE – функция, собирающая информацию от смежных вершин, f_{update} – функция обновления признаков вершины.

Процесс вычисления включает два ключевых шага: агрегацию сообщений от смежных вершин и обновление состояния вершины с использованием полученных сообщений. Ребра, если они имеют весовые или категориальные атрибуты, также могут участвовать в этом процессе, влияя на агрегацию.

Для агрегации информации о смежных вершинах используется механизм передачи сообщений Message Passing Neural Networks (MPNN) [14]. MPNN состоит из двух ключевых этапов: передача сообщений и обновление состояния. В фазе передачи сообщений каждая вершина графа обновляет свое состояние, получая информацию от смежных вершин через итерационный процесс. Этот механизм можно рассматривать как обмен сообщениями между вершинами графа, в котором они отправляют и получают информацию, чтобы постепенно улучшать свое представление о локальной и глобальной структуре графа. Сообщение от смежных вершин можно получить следующим образом:

$$m_i^{(k+1)} = \text{AGGREGATE}\left(\left\{\mathrm{M}\left(h_i^{(k)}, h_j^{(k)}, e_{ij}\right) \mid j \in N(i)\right\}\right), \quad (3)$$

где $h_i^{(k)}$ и $h_j^{(k)}$ – признаки вершин i и j на слое k, e_{ij} – признаки между вершинами i и j, M – функция генерации сообщения на основе признаков, AGGREGATE – функция агрегации, объединяющая сообщения от смежных вершин.

В зависимости от задач используют различные функции агрегации, но в данном эксперименте выбрана функция минимизации, которая позволяет выделять ключевые вершины и ребра, обладающие минимальным вкладом в общий вес дерева:

$$\text{AGGREGATE} = \min_{j \in N(i)} \text{M}\left(h_i^{(k)}, h_j^{(k)}, e_{ij}\right). \quad (4)$$

Функция передачи сообщений позволяет передать информацию между двумя вершинами, она включает в себя признаки вершин и ребра, соединяющие данные вершины:

$$\text{M}\left(h_i^{(k)}, h_j^{(k)}, e_{ij}\right) = w_{ij} + W \cdot \left[h_i^{(k)}; h_j^{(k)}\right] + b, \quad (5)$$

где w_{ij} – вес ребра между вершинами i и j, W – матрица весов, которая применяется к конкатенации признаков вершин $h_i^{(k)}$ и $h_j^{(k)}$, $[h_i^{(k)}; h_j^{(k)}]$ – вектор, представляющий собой конкатенацию признаков вершин i и j, b – смещение, которое добавляется после линейной трансформации.

Итоговую функцию полученного агрегированного сообщения от вершин из окрестности вершины i на слое k можно представить следующим образом:

$$m_i^{(k+1)} = \min_{j \in N(i)} \left(w_{ij} + W \cdot \left[h_i^{(k)}; h_j^{(k)}\right] + b\right). \quad (6)$$

В процессе работы механизма передачи сообщений каждая вершина создает сообщения для своих смежных вершин, которые зависят от текущих представлений как вершины-отправителя, так и вершины-получателя, а также учитывают свойства ребра между ним. Процесс передачи сообщений выполняется итеративно, что позволяет распространять информацию на все более удаленные области графа: за одну итерацию вершина получает данные от своих смежных вершин, за две – от вершин являющихся смежными с вершинами на первой итерации и так далее. Таким образом, за K итераций узел получает информацию из K-окрестности графа, что позволяет ему учитывать не только локальную, но и глобальную структуру. Эти итерации критически важны для моделирования сложных зависимостей в графе, например, в задачах на поиск оптимальных деревьев, как в задаче Штейнера. Благодаря фазе передачи сообщений каждая вершина в конечном счете обладает достаточной информацией, чтобы принимать решения на основе взаимосвязей с другими частями графа.

Несмотря на различные подходы к решению задачи Штейнера с использованием нейронных сетей одним из ключевых этапов является подготовка обучающего набора данных. В качестве обучающего набора данных могут выступать как готовые наборы данных, так и сгенерированные графы: случайные графы и графы, созданные на реальных данных. Генерация обучающих данных для деревьев Штейнера требует создания разнообразных графов. Для задач машинного обучения, таких как обучение графовых нейронных сетей, необходимо наличие обширного набора примеров с метками, отражающими характеристики и свойства решений задачи. В этом контексте графы, используемые для поиска деревьев Штейнера, должны охватывать как синтетические, так и реальные данные, чтобы обеспечить широкую обобщаемость моделей.

Графы из библиотеки SteinLib предоставляют разнообразные структуры и характеристики, включая реальные и синтетические данные [15]. Библиотека содержит графы различных типов и структур, которые по природе создания делятся на две группы:

созданные на основе реальных задач и синтетические, сгенерированные искусственно. Однако ограниченное количество графов, фиксированные структуры и отсутствие полной информации о большинстве оптимальных деревьев Штейнера (включая длину для многих из них) создают серьезные ограничения для обучения нейронных сетей, особенно при работе с крупными или уникальными графами. Но несмотря на сложность использования библиотеки для обучения нейронных сетей, она используется для оценки отклонения результатов, предсказанных нейронной сетью, от оптимального значения.

Для обучения нейронной сети предпочтительнее генерировать графы самостоятельно, например, с помощью эвристических алгоритмов, таких как метод Мельхорна [16]. Этот метод позволяет создавать разнообразные графы с полной структурной информацией, лучше отражающие прикладные задачи и повышающие универсальность моделей. Метод Мельхорна является одной из наиболее известных эвристик для задачи решения задачи Штейнера и предлагает решения с отклонением от оптимального не более чем в 2 раза, коэффициент отклонения можно вычислить следующим образом:

$$\text{Коэффициент отклонения} = 2 - \frac{1}{k}, \tag{7}$$

где k – количество листьев у полученного дерева Штейнера. Основная идея данного метода заключается в поэтапном добавлении ребер и вершин таким образом, чтобы минимизировать прирост веса на каждом шаге. Работа алгоритма (Рисунок 1) начинается с преобразования исходного графа. Для этого вычисляются кратчайшие пути между всеми терминальными вершинами и на основании этих путей строится полный граф, где вершинами являются терминалы, а веса ребер равны длинам кратчайших путей в исходном графе. Далее, для этого полного графа строится минимальное остовное дерево. Это дерево выступает в роли приближенного решения задачи. На следующем этапе каждое ребро минимального остовного дерева заменяется соответствующим кратчайшим путем в исходном графе. На финальной стадии выполняется оптимизация: удаляются избыточные ребра и вершины, чтобы сохранить связность дерева при минимальном весе. Итоговое дерево Штейнера включает терминальные вершины, обеспечивая приближенное, но эффективное решение задачи.

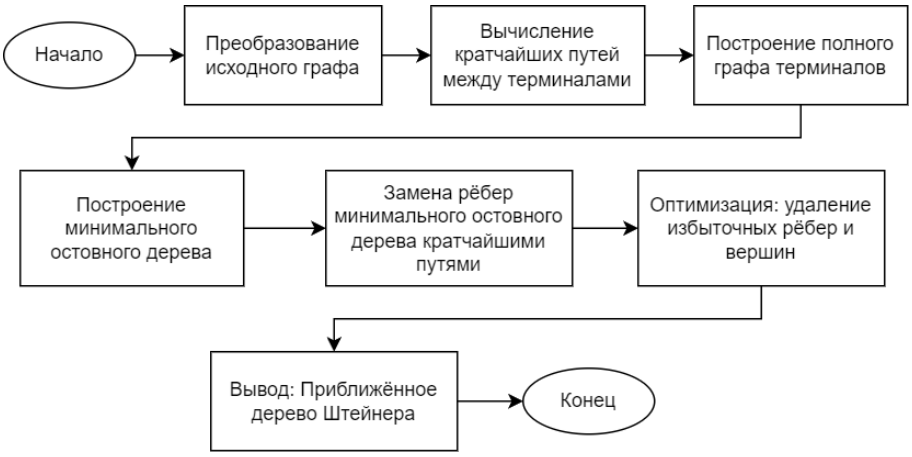

Рисунок 1 – Этапы генерации графа с помощью метода Мельхорна
Figure 1 – Stages of graph generation using the Mehlhorn method

Помимо гибкости, использование метода Мельхорна для генерации данных обеспечивает масштабируемость. Например, при решении задач с большими графами, где классические алгоритмы сталкиваются с ограничениями производительности,

сгенерированные наборы данных помогают GNN быстро адаптироваться. Более того, такие графы можно моделировать под реальные задачи, включая сложные инфраструктуры, что делает обучение моделей более релевантным для прикладных приложений. В отличие от графов из SteinLib, где представлена только длина оптимального дерева Штейнера без информации о вершинах и ребрах, образующих это дерево, сгенерированные графы предоставляют всю необходимую информацию, обеспечивая необходимое разнообразие и возможность настройки под конкретные условия. Это делает их более подходящими для создания эффективных и универсальных решений задачи Штейнера.

Результаты

Для проведения вычислительного эксперимента использовался сгенерированный набор из 6000 графов. Более детальную информацию об обучаемом наборе данных можно получить, выделив из всех графов несколько видов в зависимости от их характеристик. По плотности можно выделить разреженный граф, у которого плотность меньше 0,1, граф средней плотности, плотность которого в диапазоне от 0,1 до 0,3, и плотный граф с плотностью больше, чем 0,3. По размерности выделены маленькие графы с количеством вершин меньше 20, средние графы с количеством вершин в диапазоне от 20 до 50, и большие графы с количеством вершин больше 50. Также графы были разделены на два типа в зависимости от кластерного коэффициента: с высоким (0,5 и больше) и низким (меньше 0,5) значением.

Для проверки эффективности предложенной методики был организован вычислительный эксперимент, в котором графовая нейронная сеть обучалась и тестировалась на задаче минимального дерева Штейнера. В качестве тестового набора использовались различные синтетические графы, которые имели разнообразные свойства и особенности (Таблица 1). Эти графы варьировались по количеству связей и размерности, с количеством вершин от 15 до 250. Включались графы с различной плотностью связей: от редких с малым числом ребер до плотных графов с большим числом соединений между вершинами. Также учитывались различные топологии, включая случайно сгенерированные и регулярные решетки, что позволило проверить модель на широком спектре графов с различной структурой и сложности.

Таблица 1 – Виды графов из обучаемого набора
Table 1 – Types of graphs from the training set

Вид графа	Кол-во графов	Среднее количество вершин	Среднее количество рёбер	Средняя плотность	Среднее значение кластерного коэффициента
Разреженный	1998	122,4	444,7	0,03	0,33
Средней плотности	1108	56,6	844,3	0,22	0,71
Плотный	2894	58	1998	0,45	0,78
Маленький	133	17	106,7	0,39	0,71
Средний	1963	35	444,5	0,36	0,73
Большой	3904	103,5	1721,2	0,22	0,56
С высоким кластерным коэффициентом	4723	65,5	1529,1	0,34	0,74
С низким кластерным коэффициентом	1277	129,7	300,8	0,02	0,17

Для оценки качества обучения модели использовались такие метрики как точность и полнота. Точность показывает, какая доля предсказанных как положительные объекты (ребра и вершины дерева Штейнера) действительно являются положительными:

$$\text{Точность} = \frac{TP}{TP+FP}, \quad (8)$$

где TP – истинно положительные предсказания (модель правильно предсказала объекты положительного класса), а FP – ложноположительные предсказания (модель ошибочно предсказала отношение объекта к положительному классу, но на самом деле объект относится к отрицательному классу).

Полнота – доля правильно предсказанных положительных объектов среди всех реальных положительных объектов. Она вычисляется по формуле:

$$\text{Полнота} = \frac{TP}{TP+FN}, \quad (9)$$

где TP – истинно положительные предсказания, а FN – количество ложноотрицательных предсказаний.

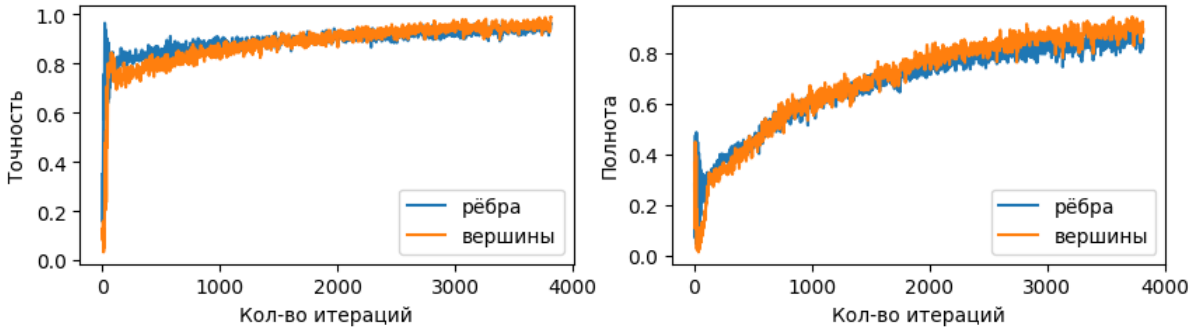

Рисунок 2 – Изменение метрик во время обучения
Figure 2 – Changes in metrics during training

По окончании 3780 итераций обучения нейронной сети были получены основные метрики нейронной сети за весь процесс обучения. Стоит также отметить, что точность и полнота предсказания ребер до 1000 итерации была больше, а к концу обучения была на 2–7 % меньше, чем для вершин. На Рисунке 2 показано, что во время всего обучения полнота имеет тенденцию к медленному росту и достигает своих значений в 0,87–0,94 для вершин и в 0,8–0,9 для ребер. Точность предсказания вершин и ребер имеет небольшой разброс в 1–2 % и достигает показателей 0,91–0,97.

Несмотря на высокие показатели качества модели, могут возникать случаи, когда предсказанное дерево Штейнера содержит лишние ребра или вершины. Кроме того, возможно отсутствие некоторых ребер, что может привести к несвязности графа. Данные случаи изображены на Рисунке 3. Чтобы избежать таких ошибок, после предсказания выполняется валидация графа и его последующее исправление. В рамках валидации проверяется связность графа, все терминальные вершины должны быть соединены в единую компоненту. Также проверяется минимальность дерева, исключаются избыточные ребра и вершины, не влияющие на соединение терминалов. После валидации происходит добавление к дереву недостающих ребер и также удаляются лишние элементы, если это необходимо. Эти шаги позволяют улучшить качество предсказанной структуры и привести ее к корректному виду.

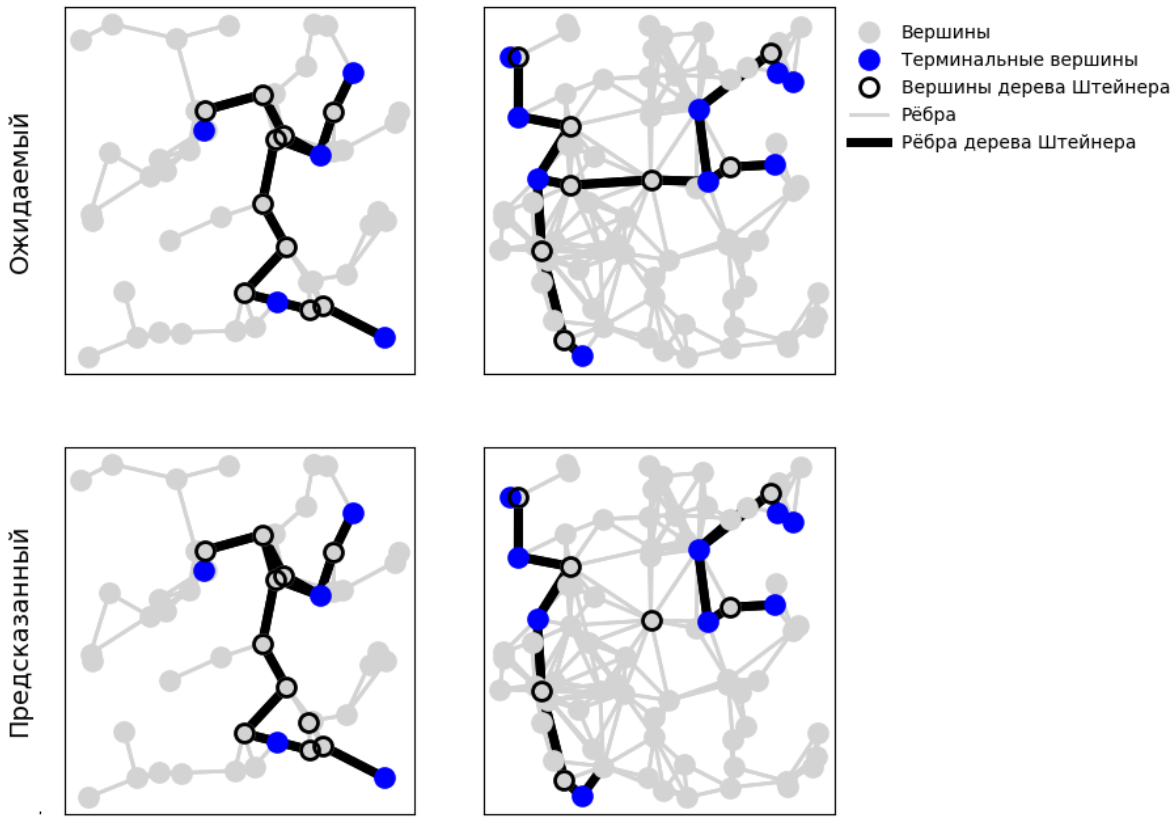

Рисунок 3 – Примеры некорректно предсказанных графов
Figure 3 – Examples of incorrectly predicted graphs

Обученная модель была протестирована на 100 сгенерированных графах средней плотности с разным количеством вершин. Тестирование осуществлялось на графах с количеством вершин 30, 60, 120, 200, 400, где на каждый размер приходится 20 сгенерированных графов. В Таблице 2 представлены полученные средние показатели точности и полноты для вершин и ребер в зависимости от количества вершин исходного графа.

Таблица 2 – Изменение метрик от размерности графа
Table 2 – Changes in metrics depending on graph size

Количество вершин графа	Точность (Рёбра)	Точность (Вершины)	Полнота (Рёбра)	Полнота (Вершины)
30	0,97	0,98	0,95	0,96
60	0,95	0,96	0,92	0,94
120	0,94	0,95	0,92	0,92
200	0,93	0,95	0,89	0,9
400	0,89	0,9	0,82	0,85

Обученная модель также была протестирована на графах, имеющих оптимальное решение, из библиотеки SteinLib на конкретных тестовых наборах I080, I160 и I320. Проводилось сравнение оптимального решения с полученным результатом, в случаях, если предсказанный результат не проходил валидацию, были применены исправления. Усредненные результаты по каждому тестовому набору представлены в Таблице 3.

Таблица 3 – Коэффициенты отклонения от оптимального решения
Table 3 – Deviation coefficients from the optimal solution

Тестовый набор	Коэффициент отклонения от оптимального решения
I080	1,42
I160	1,71
I320	2,34

Обсуждение

В ходе проведенного вычислительного эксперимента было установлено, что графовая нейронная сеть демонстрирует хорошие результаты на графах, схожих с теми, что использовались в процессе обучения. Однако при увеличении размера входных графов наблюдается значительное снижение метрик точности и полноты. Это снижение, скорее всего, связано с ограничениями механизма MPNN, который агрегирует информацию о соседних вершинах. Поскольку количество шагов в MPNN ограничено, это может затруднять сбор полной информации о более сложных графах.

Кроме того, несмотря на то, что сеть показывает достаточно высокую точность на графах из обучающего набора, некоторые графы, на которых были проведены тесты, оказались некорректные. Для их исправления применялись дополнительные меры, такие как добавление новых ребер и вершин, а также удаление избыточных элементов графа. Эти корректировки, безусловно, оказали влияние на итоговые результаты, снижая их точность и полноту, так как такие изменения могут нарушать структуру оптимальных решений.

Важным наблюдением является то, что при тестировании на графах из библиотеки SteinLib, имеющих оптимальные решения, графовая нейронная сеть показала хорошие результаты, особенно на графах с 80 вершинами. Однако при увеличении размера графа до 320 вершин, вес дерева, полученного с помощью нейронной сети, значительно превысил оптимальный вес дерева Штейнера, в некоторых случаях более чем в два раза. Это подтверждает, что текущая модель сталкивается с трудностями при обработке более крупных и сложных графов, что требует дальнейших улучшений в алгоритмах агрегации информации и оптимизации модели для работы с графами большого размера.

Заключение

Графовые нейронные сети показывают хорошие результаты на графах, близких к обучающим, но испытывают трудности при увеличении масштаба проблемы, что проявляется в снижении ключевых метрик точности и полноты. Для улучшения точности на больших графах потребуется адаптация текущей архитектуры сети, возможно, с увеличением числа шагов агрегации и улучшением механизма обработки сложных структур. Кроме того, можно исследовать использование более глубоких или сложных моделей, таких как графовые нейронные сети с механизмом самовнимания, которые могут лучше захватывать зависимости между удаленными вершинами. Несмотря на снижение качества при увеличении размерности, графовые нейронные сети демонстрируют высокий потенциал для решения задач на графах небольшой и средней размерности. Способность работать с комплексными структурами данных делает их перспективными для анализа сложных систем, таких как беспроводные сети, где важно учитывать взаимосвязи между вершинами.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Hwang F.K., Richards D.S. Steiner Tree Problems. *Networks.* 1992;22(1):55–89. https://doi.org/10.1002/net.3230220105
2. Ljubić I. Solving Steiner trees: Recent advances, challenges, and perspectives. *Networks.* 2021;77(2):177–204. https://doi.org/10.1002/net.22005
3. Basok M., Cherkashin D., Rastegaev N., Teplitskaya Ya. On Uniqueness in Steiner Problem. *International Mathematics Research Notices.* 2024;2024(10):8819–8838. https://doi.org/10.1093/imrn/rnae025
4. Лотарев Д.Т., Уздемир А.П. Преобразование задачи Штейнера на евклидовой плоскости к задаче Штейнера на графе. *Автоматика и телемеханика.* 2005;(10):80–92.
   Lotarev D.T., Uzdemir A.P. Conversion of the Steiner Problem on the Euclidean Plane to the Steiner Problem on Graph. *Automation and Remote Control.* 2005;66(10):1603–1613. https://doi.org/10.1007/s10513-005-0194-y
5. Grigoreva D.R., Faizullina A.G., Basyrov R.R., Sharipov R.S. Use of Steiner Problem in Solving Practical Problems of Road Construction. *Modern Applied Science.* 2015;9(4):294–302. https://doi.org/10.5539/mas.v9n4p294
6. Winter P. Steiner Problem in Networks: A Survey. *Networks.* 1987;17(2):129–167. https://doi.org/10.1002/net.3230170203
7. Morrison D.R., Jacobson S.H., Sauppe J.J., Sewell E.C. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization.* 2016;19:79–102. https://doi.org/10.1016/j.disopt.2016.01.005
8. Fampa M., Lee J., Melo W. A specialized branch-and-bound algorithm for the Euclidean Steiner tree problem in *n*-space. *Computational Optimization and Applications.* 2016;65(1):47–71. https://doi.org/10.1007/s10589-016-9835-z
9. Do T.A., Ban H.-B., Huynh T.T.B., Le M.T., Nguyen B.L. Genetic algorithm based approach to solve the Clustered Steiner Tree Problem. *Evolutionary Intelligence.* 2023;17:1547–1566. https://doi.org/10.1007/s12065-023-00848-w
10. Jayadeva, Bhaumik B. A neural network for the Steiner minimal tree problem. *Biological Cybernetics.* 1994;70:485–494. https://doi.org/10.1007/BF00203241
11. Yen B.P.-C., Luo Yu. Deep Reinforcement Learning for Solving Directed Steiner Tree Problems. In: *TENCON 2022 – 2022 IEEE Region 10 Conference (TENCON), 01–04 November 2022, Hong Kong, China.* IEEE; 2022. pp. 1–5. https://doi.org/10.1109/TENCON55691.2022.9977539
12. Kahng A.B., Nerem R.R., Wang Yu., Yang Ch.-Yi. NN-Steiner: A Mixed Neural-Algorithmic Approach for the Rectilinear Steiner Minimum Tree Problem. In: *Proceedings of the AAAI Conference on Artificial Intelligence: The Thirty-Eighth AAAI Conference on Artificial Intelligence, 20–27 February 2024, Vancouver, Canada.* Washington: AAAI Press; 2024. pp. 13022–13030. https://doi.org/10.1609/aaai.v38i12.29200
13. Wu Z., Pan Sh., Chen F., Long G., Zhang Ch., Yu Ph.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems.* 2021;32(1):4–24. https://doi.org/10.1109/TNNLS.2020.2978386
14. Gilmer J., Schoenholz S.S., Riley P.F., Vinyals O., Dahl G.E. Message Passing Neural Networks. In: *Machine Learning Meets Quantum Physics.* Cham: Springer; 2020. pp. 199–214. https://doi.org/10.1007/978-3-030-40245-7_10
15. Koch T., Martin A., Voß S. SteinLib: An Updated Library on Steiner Tree Problems in Graphs. In: *Steiner Trees in Industry.* New York: Springer; 2001. pp. 285–325. https://doi.org/10.1007/978-1-4613-0255-1_9

16. Mehlhorn K. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters.* 1988;27(3):125–128. https://doi.org/10.1016/0020-0190(88)90066-X

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Пиминов Дмитрий Алексеевич, старший преподаватель, кафедра «Прикладные информационные технологии», Саратовский государственный технический университет имени Гагарина Ю.А., Саратов, Российская Федерация.
e-mail: dima-piminov@yandex.ru
ORCID: 0000-0002-7234-6874

Dmitriy A. Piminov, Senior Lecturer, Applied Information Technologies Department, Yuri Gagarin State Technical University of Saratov, Saratov, the Russian Federation.

Печенкин Виталий Владимирович, доктор социологических наук, кандидат физико-математических наук, профессор, кафедра «Прикладные информационные технологии», Саратовский государственный технический университет имени Гагарина Ю.А., Саратов, Российская Федерация.
e-mail: pechenkinvv@sstu.ru
ORCID: 0000-0002-5043-1891

Vitaly V. Pechenkin, Doctor of Sociological Sciences, Candidate of Physical and Mathematical Sciences, Professor, Applied Information Technologies Department, Yuri Gagarin State Technical University of Saratov, Saratov, the Russian Federation.

Королёв Михаил Сергеевич, кандидат технических наук, доцент, кафедра «Прикладные информационные технологии», Саратовский государственный технический университет имени Гагарина Ю.А., Саратов, Российская Федерация.
e-mail: koroliow.mikhail@yandex.ru
ORCID: 0000-0002-4901-4468

Mikhail S. Korolev, Candidate of Egineering Sciences, Associate Professor, Applied Information Technologies Department, Yuri Gagarin State Technical University of Saratov, Saratov, the Russian Federation.

Статья поступила в редакцию 18.02.2025; одобрена после рецензирования 17.03.2025; принята к публикации 24.03.2025.

The article was submitted 18.02.2025; approved after reviewing 17.03.2025; accepted for publication 24.03.2025.