Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина»
Институт радиоэлектроники и информационных технологий-РТФ
Кафедра информационных технологий и систем управления

ДОПУСТИТЬ К ЗАЩИТЕ ПЕРЕД ГЭК
Заведующий кафедрой ИТиСУ
_____ Е. В. Кислицын
«30» мая 2025 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

РАЗРАБОТКА И АПРОБАЦИЯ МОДЕЛИ ГЕНЕРАЦИИ РАСПИСАНИЯ ЗАНЯТИЙ С ИСПОЛЬЗОВАНИЕМ ГРАФОВОЙ НЕЙРОННОЙ СЕТИ И СИСТЕМЫ МУЛЬТИАГЕНТНОГО ВЗАИМОДЕЙСТВИЯ

Научный руководитель: Сысков Алексей Мстиславович
к.т.н., доцент

подпись

Нормоконтролер: Огуренко Егор Владимирович

подпись

Студент группы: РИМ-230963 Зиннер Рон Александрович

подпись

Екатеринбург
2025

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
**«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»**

Институт радиоэлектроники и информационных технологий – РТФ
Кафедра информационных технологий и систем управления
Направление подготовки 09.04.01 Информатика и вычислительная техника
Образовательная программа 09.04.01/33.03 Инженерия машинного обучения

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студента Зиннер Рона Александровича **группы** РИМ-230963
(фамилия, имя, отчество)

1. Тема выпускной квалификационной работы
Разработка и апробация модели генерации расписания занятий с использованием графовой нейронной сети и системы мультиагентного взаимодействия
Утверждена распоряжением по институту от «02» декабря 2024 г. № 33.02-05/334

2. Научный руководитель
Сысков Алексей Мстиславович, к.т.н., доцент

3. Исходные данные к работе
Нормативная, учебная, методическая литература по теме магистерской диссертации, материалы, полученные в ходе преддипломной практики, техническое задание по разработке продукта, научная статья

4. Перечень демонстрационных материалов
Презентация, архитектура модели генерации расписания, приложение

5. Календарный план

№ п/п	Наименование этапов выполнения работы	Срок выполнения этапов работы	Отметка о выполнении
1.	1 раздел (глава)	до 24.03.2025 г.	Выполнено
2.	2 раздел (глава)	до 28.04.2025 г.	Выполнено
3.	3–4 раздел (глава)	до 19.05.2025 г.	Выполнено
4.	ВКР в целом	до 23.05.2025 г.	Выполнено

Научный руководитель Сысков Алексей Мстиславович _____
Ф.И.О. (подпись)

Студент задание принял к исполнению 10.02.2025 г. _____
дата (подпись)

6. Допустить Зиннер Рона Александровича к защите выпускной квалификационной работы в экзаменационной комиссии

Заведующий кафедрой ИТиСУ _____ Е. В. Кислицын
(подпись) Ф.И.О.

РЕФЕРАТ

Выпускная квалификационная работа содержит 78 страниц, 5 рисунков, 2 таблицы и 52 источника.

Ключевые слова: МАШИННОЕ ОБУЧЕНИЕ, ГРАФОВАЯ НЕЙРОННАЯ СЕТЬ, МУЛЬТИАГЕНТНАЯ СИСТЕМА, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ, ОПТИМИЗАЦИЯ РАСПИСАНИЯ, РАСПИСАНИЕ ЗАНЯТИЙ

Цель выпускной квалификационной работы – Разработка и апробация модели генерации расписания на основе графовой нейросети и мультиагентной системы для автоматизации распределения преподавателей с учётом реальных ограничений.

Объект ВКР – процесс планирования занятий и распределения преподавателей по учебным группам в образовательной организации.

Предмет ВКР – модели и алгоритмы интеллектуального составления расписания с использованием графовых нейросетей и мультиагентного взаимодействия, а также механизмы учёта ограничений без их явного программного описания.

Методы исследования: Применены графовое машинное обучение, мультиагентное моделирование, анализ ограничений и методы симуляции конфликтов.

Результаты работы: Создана и протестирована интеллектуальная система, способная формировать расписание в условиях неполных и изменяющихся данных, обеспечивая устойчивость, точность назначения и равномерную нагрузку на преподавателей.

Выпускная квалификационная работа выполнена в текстовом редакторе и представлена в электронном виде.

СОДЕРЖАНИЕ

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

1. Графовая нейронная сеть (GNN) – Модель машинного обучения, предназначенная для работы с данными, представленными в виде графов. Позволяет учитывать не только свойства отдельных объектов, но и их взаимосвязи.

2. Мультиагентная система (MAS) – Архитектура программной системы, состоящая из независимых агентов, каждый из которых действует в соответствии с собственными целями, правилами и знаниями, взаимодействуя с другими агентами для достижения общей цели.

3. Агент – Программный объект в составе MAS, обладающий автономностью, локальной информацией и возможностью принимать решения на основе заданных правил.

4. CRM-система – Система управления взаимоотношениями с клиентами (в данном контексте – с учениками), в которой хранятся данные о группах, занятиях, преподавателях и истории обучения.

5. Топ-k Accuracy – Метрика качества, показывающая, попал ли правильный вариант в список из k лучших предсказаний модели.

6. Precision@k – Метрика оценки качества ранжирования: доля релевантных результатов в топ–k

7. Recall@k – Метрика оценки качества ранжирования: доля покрытых релевантных объектов в топ-k.

8. Вершина графа (Node) – Объект в графе (например, преподаватель, тема, группа), обладающий своими признаками.

9. Ребро графа (Edge) – Связь между двумя вершинами, отражающая их взаимодействие или зависимость.

10. Признак узла (Node Feature) – Числовое или категориальное описание сущности в графе (например, набор компетенций преподавателя).

ВВЕДЕНИЕ

Составление расписания занятий – одна из основных организационных задач в сфере образования, напрямую влияющая на эффективность учебного процесса, а также на рациональное использования ресурсов и удовлетворённость как учеников, так и преподавателей. Особую сложность имеет процесс организации занятий в сфере дополнительного образования для детей в области информационных технологий, где необходимо учитывать возрастные особенности обучающихся, индивидуальные образовательные траектории, ограниченность ресурсов и разнообразие изучаемых курсов.

Современная практика ИТ-обучения подразумевает изучение различных инструментов и технологий, преподаваемых преподавателями различного уровня компетентности. При этом образовательный процесс должен оставаться гибким, адаптивным и устойчивым к внеплановым изменениям, таким как болезни преподавателей, технические сбои или изменения в расписании занятий.

В традиционных системах планирования расписаний эта задача решается вручную или с применением простых шаблонных подходов, что приводит к риску перегрузки преподавателей, конфликта по времени и местоположению, а также снижению качества образовательного взаимодействия. В условиях масштабируемого обучения, охватывающего сотни групп и преподавателей, такие подходы становятся неэффективными.

Современные технологии искусственного интеллекта позволяют автоматизировать процесс составления расписания, учитывая многослойные зависимости между участниками процесса: группами учеников, преподавателями, темами занятий, временем и местом проведения занятий. Особенно перспективным представляется использование графовых нейронных сетей (GNN), которые эффективно моделируют взаимосвязи между сущностями, а также мультиагентных систем (MAS), обеспечивающих

гибкое взаимодействие между участниками учебного процесса.

Целью данной работы является разработка и апробация модели автоматизированной генерации расписаний на основе гибридной архитектуры GNN + MAS, предназначенной для применения в системе очного обучения детей IT-дисциплинам. Разработанная система должна обеспечивать интеллектуальное распределение преподавателей по группам с учётом множества параметров и ограничений.

Предметом исследования выступает возможность построения модели, способной учитывать ограничения расписания без их явного кодирования в коде. Также изучается потенциал мультиагентной логики как компенсирующего механизма, повышающего устойчивость решений в условиях неопределённости.

Актуальность темы обусловлена растущим спросом на масштабируемые и интеллектуальные решения в сфере образовательного планирования, а также необходимостью оптимизации существующих ресурсов при обеспечении высокого качества обучения. Практическая значимость заключается в возможности применения разработанной модели в реальной среде образовательной организации с последующим масштабированием и адаптацией под другие форматы обучения.

В отличие от классических решений составления расписаний, где ограничения кодируются вручную и решаются через оптимизацию, предложенная модель реализует рекомендательный подход [36], [37]. Графовая нейросеть генерирует множество кандидатных решений, а мультиагентная система фильтрует и корректирует их с учётом текущих условий. Такая архитектура делает систему более адаптивной, понятной для пользователей и масштабируемой при изменении требований. Также мультиагентная система в данной архитектуре не только проверяет ограничения, но и обеспечивает балансировку преподавательской нагрузки. Это реализуется на уровне логики агентов и не требует изменения модели.

Такое поведение обеспечивает устойчивость расписания и снижает вероятность перегрузки отдельных преподавателей, что также можно рассматривать как элемент архитектурной новизны системы.

ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ОБОСНОВАНИЕ АКТУАЛЬНОСТИ ТЕМЫ

1.1 Предметная область

Современная система дополнительного образования в сфере информационных технологий играет важную роль в развитии цифровых навыков у школьников и подростков. В условиях стремительного роста технологических отраслей и цифровизации различных сфер экономики, обучение детей программированию, робототехнике, разработке игр, а также другим направлениям IT становится всё более востребованным.

Дополнительное IT-образование – это не только возможность углубить знания за рамками школьной программы, но и один из инструментов ранней профориентации. Курсы, рассчитанные на детей в возрасте от 6 до 14 лет, проводятся в очном формате, преимущественно в виде коротких, интенсивных программ, разделённых на тематические модули.

Организация такого обучения имеет ряд особенностей:

— дети обучаются в малых группах (до 12 человек);

— занятия проходят по заранее утверждённому расписанию, как правило, один раз в неделю по два академических часа;

— образовательный процесс осуществляется на специально оборудованных локациях с компьютерами, интернетом и специализированным ПО;

— используются готовые учебные программы, разработанные с учётом возраста и уровня подготовки учащихся.

Также важно учитывать, что состав групп остаётся фиксированным на протяжении модуля. Учебная нагрузка преподавателя, доступность аудитории, особенности расписаний учеников, логистика между филиалами – всё это создаёт множество ограничений, которые необходимо учитывать при планировании.

1.1.1 Проблематика ручного составления расписания

Традиционно в подобных образовательных организациях расписание формируется вручную менеджерами или административным персоналом. При небольшом количестве групп такой подход работает удовлетворительно, однако при масштабировании он становится неэффективным. Например, при наличии нескольких десятков тысяч учеников в разных городах, тысяч преподавателей и сотен учебных локаций, возникает высокая вероятность ошибок: пересечений по времени, перегрузки отдельных преподавателей, назначения неподходящих педагогов и т.п.

Кроме того, ручное планирование вызывает трудности при учёте важных дополнительных параметров: рейтинг и опыт преподавателя, историю занятий с конкретной группой, персональные ограничения (отпуска, болезни), особенности пробных или индивидуальных занятий. Всё это приводит к потере качества обучения и неудовлетворённости со стороны родителей и детей.

1.1.2 Цифровизация и подход к автоматизации

Решение данной задачи возможно за счёт применения современных цифровых и интеллектуальных технологий. В рамках данной работы предлагается разработка модели, которая будет автоматически формировать расписание занятий на основе множества входных параметров. Для этого необходимо формализовать предметную область и определить ключевые сущности: преподаватели, группы, модули, локации, временные слоты и правила взаимодействия между ними.

Особенность образовательного процесса в данной сфере заключается в модульности и регулярности: темы для обучения заранее определены, длительность модулей известна, логика их смены предсказуема. При этом каждый преподаватель может вести занятия по разным темам, а каждая группа имеет свою историю и предпочтения. Это делает задачу расписания похожей на задачу распределения ресурсов в условиях ограничений, которую можно

решать с применением графовых структур и систем, моделирующих поведение агентов (участников процесса) [40].

1.1.3 Роль информационных систем

Во многих центрах IT-обучения уже используются системы управления учебным процессом (CRM-системы), в которых хранятся данные о группах, темах, преподавателях и расписаниях. Однако их основной функционал – отображение и хранение информации, а не интеллектуальное планирование и отслеживание выполнения задач. Автоматизация распределения преподавателей по занятиям на основе этих данных требует внедрения модели, способной:

— учитывать исторические данные и ограничения,

— обеспечивать адаптивность к изменениям,

— предсказывать оптимальное назначение преподавателя на занятие.

Такая система может быть реализована с использованием гибридного подхода, сочетающего графовые нейронные сети (GNN) – для обработки структуры данных, и мультиагентную систему (MAS) – для согласования решений на основе правил и предпочтений агентов.

1.2 Ограничения: локации, преподаватели, модули, группы

Организация учебного процесса в системе дополнительного образования по информационным технологиям сопряжена с множеством ограничений, связанных как с инфраструктурой, так и с человеческими и содержательными ресурсами. Эти ограничения необходимо строго учитывать при построении расписания, поскольку их нарушение может привести к сбоям в работе всей системы, недовольству со стороны родителей и перегрузке преподавателей.

В рамках данной работы ограничения можно условно разделить на четыре основные категории: локационные, преподавательские, тематические

(модульные) и групповые.

1.2.1 Локации

Каждая учебная группа закреплена за определённой локацией (филиалом), на которой проходят все занятия этой группы. Перемещение групп между локациями не допускается. Это обусловлено как территориальной логистикой (родители выбирают ближайший к дому филиал), так и внутренней организацией учебного процесса.

Ограничения, связанные с локациями, включают:

— Ограниченное число аудиторий и временных слотов: на локации одновременно могут обучаться не более 1–3 групп, в зависимости от технической оснащённости и пропускной способности;

— Требования к квалификации преподавателя: для некоторых локаций установлены минимальные рейтинги преподавателей, что связано с престижностью, историей работы филиала или особенностями целевой аудитории;

— Физическая доступность: преподаватель, назначенный на занятие, должен иметь возможность добраться до локации, что ограничивает количество возможных назначений в пределах одного дня.

1.2.2 Преподаватели

Кадровый ресурс является одним из наиболее критичных в системе IT-обучения. При составлении расписания необходимо учитывать ряд ограничений, связанных с графиком, компетенциями и рабочей нагрузкой преподавателя:

— Компетенции: каждый преподаватель обладает набором навыков, ограниченным перечнем тем, которые он может преподавать. Эти данные представлены в виде связей между преподавателями и модулями;

— Загруженность: преподавателю устанавливается недельный лимит часов (например, 12, 16 или 20 часов), который нельзя превышать. Кроме того, необходимо соблюдать равномерность распределения нагрузки в течение

недели;

— Непересекаемость: преподаватель не может вести два занятия одновременно – система должна учитывать все уже запланированные занятия [6];

— История взаимодействия: приоритет отдается тем преподавателям, которые уже вели занятия в данной группе ранее. Это повышает стабильность восприятия информации детьми и снижает время на адаптацию;

— Недоступные дни: каждый преподаватель имеет выходные дни недели, в которые он не работает.

1.2.3 Модули

Учебные темы реализуются в виде модулей – законченных блоков программы продолжительностью от 4 до 12 занятий. Каждая группа обучается только по одному модулю в конкретный период времени. При формировании расписания занятий следует учитывать следующие ограничения:

— Последовательность: некоторые модули имеют зависимости и не могут быть изучены раньше других (например, «Python: углублённый» – только после «Python: основы»);

— Сложность: модули имеют разные уровни сложности, и группа должна быть готова к его прохождению, иначе требуется либо дополнительная подготовка, либо подбор другого модуля;

— Фиксированное расписание: модульные занятия не могут быть перемещены внутри недели – они проходят раз в неделю в одно и то же время;

— Наличие преподавателя: модуль может быть назначен группе только при наличии преподавателя, обладающего соответствующими компетенциями и доступного в нужный день и время.

1.2.4 Группы

Каждая группа – это фиксированный состав учащихся, подобранный по возрасту и уровню подготовки. В рамках создания расписания необходимо учитывать:

— Фиксированное время занятий: каждая группа посещает занятия в строго определённый день и временной слот, который сохраняется на протяжении всего учебного года;

— Фиксированная локация: как указано выше, перемещение между филиалами не допускается [3];

— Пробные и индивидуальные занятия: некоторые группы могут принимать новых учеников через пробные занятия, которые должны быть встроены в расписание без пересечений с основными модулями;

— Приоритет преподавателя: если группа долгое время обучалась с одним преподавателем, желательно сохранить преемственность, если это возможно.

Таким образом, каждая категория ограничений вносит вклад в сложность задачи составления расписания. Их совокупность требует применения моделей, способных одновременно учитывать структурные связи и правила взаимодействия между сущностями. Использование графовой нейросети позволяет формализовать структуру ограничений, а мультиагентная система – обеспечить их гибкое и контекстное соблюдение при генерации расписания.

1.3. Обзор существующих подходов к генерации расписания

Задача автоматического составления расписания имеет долгую историю и активно исследуется как в академической среде [4], так и в сфере промышленного программного обеспечения. Несмотря на кажущуюся прикладную направленность, она относится к числу NP-трудных задач [3], поскольку предполагает необходимость удовлетворения большого числа ограничений [35] – как жёстких (например, пересечения по времени), так и мягких (например, предпочтения преподавателей или стабильность учеников в группах). В контексте дополнительного образования по ИТ дисциплинам эта

задача дополнительно усложняется нестандартным графиком, разнородными ресурсами и динамикой модульной структуры.

Существующие подходы к решению задачи можно условно разделить на три большие группы: классические алгоритмические методы, эвристические/метаэвристические подходы [10] и методы, основанные на машинном обучении и нейросетевых архитектурах.

1.3.1 Классические алгоритмические методы

К этой категории относятся методы, основанные на точных алгоритмах и строго определённых правилах. Наиболее известными являются:

— Метод целочисленного линейного программирования (ILP) – позволяет формализовать задачу составления расписания в виде системы ограничений и целевых функций. Однако при большом количестве переменных (например, сотни преподавателей и групп) ILP становится вычислительно неэффективным;

— Графовые алгоритмы – например, построение двудольных графов и раскраска графа, где вершины представляют преподавателей и группы, а рёбра – возможные занятия. Эти методы позволяют частично оптимизировать распределение, но плохо справляются с мягкими ограничениями и динамическими изменениями;

— Жадные алгоритмы – строят расписание поэтапно, исходя из текущего наилучшего выбора. Хорошо работают в небольших системах, но склонны к локальным минимумам и не обеспечивают глобальной оптимальности.

1.3.2 Эвристические и метаэвристические методы

Также существуют эвристические методы:

— Генетические алгоритмы (GA) – имитируют процесс естественного отбора, генерируя и эволюционируя множество возможных расписаний. Эффективны в задачах с большим числом параметров, но чувствительны к настройкам и требуют длительного времени на подбор оптимальных решений;

— Алгоритмы имитации отжига (SA) [31] – используют случайные перестановки с поэтапным "охлаждением" параметров, стремясь к глобальному минимуму функции потерь. Часто используются в системах, где важен компромисс между скоростью и точностью;

— Муравьиные алгоритмы [22], алгоритмы табу-поиска [25] и пр. – направлены на обход пространства решений с учётом истории и ограничений. Их главная проблема – высокая вычислительная стоимость и необходимость тщательной настройки [8].

Эти методы часто применяются в крупных вузах, при составлении расписаний конференций, а также в логистике. Однако в контексте IT-обучения детей они требуют значительной адаптации и регулярного ручного вмешательства.

1.3.3 Методы на основе машинного обучения и ИИ

Современные подходы предлагают использовать модели машинного обучения, в частности графовые нейронные сети (GNN) [39] и мультиагентные системы (MAS) [11], для более гибкого и масштабируемого решения задачи [51]. Преимущества таких методов заключаются в следующем:

— Автоматическое извлечение закономерностей из исторических данных – GNN может обучаться на прошедших расписаниях, выявляя шаблоны, которые невозможно явно запрограммировать;

— Устойчивость к неполным данным – модель может делать обоснованные предсказания даже при отсутствии полной информации;

— Гибкая архитектура – MAS позволяет смоделировать поведение участников процесса (преподаватель, группа, локация) как агентов с собственными целями и ограничениями.

Среди практических реализаций можно отметить коммерческие решения для вузов, такие как ASU TP (Автоматизированная система управления учебным процессом), UniTime, aSc Timetables, а также open-source системы (например, Kronos, OptaPlanner), часть которых уже включает модули

с ML/AI.

Однако большинство таких систем заточены под классическое школьное или вузовское расписание с жёсткими рамками [32]. Дополнительное IT-образование требует большей гибкости, поддержки модульной структуры и учёта множества динамических параметров – от рейтингов преподавателей до ограничений по пробным занятиям.

В отличие от большинства подобных решений, в которых ограничения задаются вручную или в коде, модель не требует жёсткого кодирования всех ограничений вручную.

Также при сравнении с другими вузовскими решениями, такими как OptaPlanner, где ограничения описываются в виде правил и явно задаются в коде, предлагаемая модель обучается на примерах ранее созданного расписания, корректных и некорректных назначений преподавателей. Это позволяет учитывать сложные ограничения без необходимости их программного описания, что снижает требования к разработке и облегчает адаптацию модели под новые условия.

Таким образом, несмотря на широкий спектр решений, задача генерации расписаний в контексте IT-обучения детей остаётся слабо формализованной в существующих инструментах. Это обуславливает необходимость в разработке специализированных интеллектуальных моделей, сочетающих структурное представление данных (через GNN) и поведенческую логику (через MAS), что и является предметом настоящего исследования.

1.4. Проблемы в существующих системах

Несмотря на наличие большого числа программных решений для автоматизированного составления расписаний, большинство из них ориентировано на задачи классического школьного или вузовского образования. В контексте дополнительного IT-обучения детей такие системы

оказываются малоэффективными или требуют значительной адаптации. Анализ существующих продуктов и подходов позволяет выделить ряд характерных проблем, ограничивающих их применимость в данной предметной области.

1.4.1 Отсутствие гибкости в структуре расписания

Многие системы ориентированы на фиксированный учебный план, где расписание определяется раз и навсегда на семестр или учебный год. В дополнительном образовании по IT-дисциплинам, напротив, присутствует модульная структура, предполагающая регулярную смену тем, преподавателей и состава учебных блоков. Такие изменения требуют пересмотра расписания чуть ли не еженедельно. Статические системы плохо справляются с подобной динамикой.

Кроме того, в типовых решениях не предусмотрено автоматическое продление модулей, адаптация по сложности или индивидуальные образовательные траектории учащихся, что делает невозможным автоматическое продолжение обучения после завершения темы.

1.4.2 Ограниченное управление человеческими ресурсами

Существенной проблемой является ограниченность моделей управления преподавателями. Во многих решениях отсутствует учёт:

— индивидуальных расписаний преподавателей;

— коэффициентов загруженности и предпочтений;

— приоритета на основе истории взаимодействия с группой;

— ограничений по квалификации и рейтингу.

Такие аспекты особенно важны в обучении детей: педагог, который уже знаком с группой, обеспечивает более стабильный контакт и лучше контролирует прогресс. Кроме того, некоторые локации требуют преподавателей с высоким рейтингом, для обеспечения наивысшего качества оказываемых услуг, чего типовые системы также не учитывают.

1.4.3 Недостаточная адаптивность к реальным условиям

Большинство существующих решений плохо справляются с нештатными ситуациями – будь то отмена занятия, болезнь преподавателя или изменение условий аренды локации. Такие сбои приходится вручную компенсировать, в том числе изменением расписания, что требует постоянного вмешательства менеджеров.

Также существует проблема слабой интеграции с внешними источниками данных (например, CRM-системами, в которых хранятся сведения о группах, преподавателях и истории занятий). Большинство решений работают сами по себе, без гибкой поддержки API или импорта пользовательских ограничений.

1.4.4 Игнорирование пробных и индивидуальных занятий

Дополнительное образование часто включает пробные занятия для новых учеников, а также индивидуальные сессии (в том числе как элемент подготовки к олимпиадам или доработки темы). Подобные занятия имеют собственные требования к преподавателю (обычно более опытный), не входят в основное расписание, а также назначаются с учётом специфики ребёнка и родительских предпочтений.

В большинстве систем такие занятия оформляются как "исключения", вносимые вручную, что резко снижает эффективность автоматизации.

1.4.5 Отсутствие интеллектуального анализа данных

Традиционные решения создания расписаний не используют исторические данные и не обладают механизмами самообучения. Это означает:

— отсутствие оптимизации на основе прошлых ошибок (например, системных конфликтов);

— невозможность построения рекомендаций при отсутствии очевидного решения;

— отсутствие предиктивной логики распределения преподавателей по

группам.

Таким образом, существующие системы работают по принципу "текущего момента", не используя накопленную информацию о процессе обучения и поведении агентов.

1.5. Обоснование выбора GNN и MAS

В предыдущих разделах были рассмотрены особенности организации дополнительного образования в сфере информационных технологий, ограничения, с которыми сталкиваются организаторы, а также проблемы существующих решений по автоматизации составления расписаний. Учитывая сложность предметной области и высокую вариативность входных данных, становится очевидно, что эффективное решение должно учитывать как структурную взаимосвязь сущностей, так и контекстное поведение участников образовательного процесса. В этом контексте применение графовых нейронных сетей (GNN) и мультиагентных систем (MAS) представляется обоснованным и целесообразным.

1.5.1 Графовые нейронные сети (GNN): работа со связями и структурой

В задаче генерации расписания для IT-обучения участников процесса объединяет множество взаимосвязей:

— Преподаватель – Тема (компетенции);

— Преподаватель – Группа (история занятий);

— Группа – Модуль (актуальная тема обучения);

— Группа – Локация (место проведения);

— Преподаватель – Локация (расписание и ограничения);

— Модуль – Зависимости (очерёдность и зависимости).

Такие связи удобно представить в виде графа, где вершинами являются сущности (ученики, преподаватели, темы, локации и т. д.), а рёбра отражают отношения и ограничения между ними. Классические алгоритмы плохо

работают с подобными структурами, поскольку не умеют эффективно учитывать контекст и скрытые зависимости.

GNN, в отличие от линейных моделей, способны учиться на графе и выявлять закономерности распределения преподавателей по группам, используя исторические данные [49]. Это позволяет:

— учитывать контекст назначения преподавателя не изолированно, а в общей структуре связей;

— делать обоснованные предсказания даже в случае частично отсутствующих данных (например, новых групп или преподавателей);

— предлагать оптимальные назначения не на основе фиксированных правил, а на основе опыта предыдущих успешных назначений.

Таким образом, GNN выступает как аналитическая часть системы, способная предложить кандидатов на занятие, основываясь на сложной совокупности факторов.

1.5.2 Мультиагентная система (MAS): поведенческая логика и согласование

GNN позволяет предсказывать оптимальные назначения, но она не может в полной мере учитывать правила, исключения и мягкие ограничения, присущие реальному процессу. Для этого необходим слой логики, способный принимать решения на основе интересов и ограничений всех участников. Эту роль выполняет мультиагентная система [23], [29].

В MAS каждая сущность (преподаватель, группа, локация, занятие) представляется в виде агента, обладающего:

— собственным состоянием и параметрами (расписание, приоритеты, рейтинг);

— правилами поведения (например, отказ преподавателя от занятий по средам);

— возможностью взаимодействовать с другими агентами и договариваться (в рамках планирования слота).

Такой подход позволяет реализовать мягкие ограничения, например, избегать перегрузки преподавателей или учитывать их предпочтения, а также моделировать приоритеты, то есть преподаватель с более высоким рейтингом получает доступ к сложным модулям и ключевым локациям, и обеспечить гибкое разрешение конфликтов, при возникновении накладок MAS может выбирать альтернативы без вмешательства человека.

MAS выполняет роль контролёра и адаптера, отвечая за корректное распределение ролей и соблюдение логики взаимодействия, на основе рекомендаций, предоставленных GNN.

1.5.3 Объединение GNN и MAS

Сочетание GNN и MAS позволяет объединить обучаемую модель, способную выявлять закономерности из исторических данных, с контекстно управляемой системой, учитывающей индивидуальные предпочтения, ресурсы и ограничения в режиме реального времени.

Такое гибридное решение может масштабироваться под задачи разной сложности, допускает расширение и внедрение новых типов агентов (например, в случае онлайн-обучения), и обеспечивает прозрачность и интерпретируемость решений – каждое назначение можно обосновать через взаимодействие агентов и веса модели.

Таким образом, выбор GNN и MAS для решения задачи автоматизированного формирования расписания является обоснованным как с теоретической, так и с практической точки зрения. Эти технологии позволяют совместить структурный анализ данных с логикой поведения участников, что особенно важно в условиях живой, меняющейся образовательной среды.

1.6. Роль искусственного интеллекта в образовательных технологиях

Искусственный интеллект постепенно становится неотъемлемой частью современного образования, как в его содержательной, так и в административной составляющей. На фоне стремительного роста числа обучающихся, разнообразия форматов и образовательных программ, именно ИИ позволяет обеспечить гибкость, масштабируемость и индивидуализацию учебного процесса [7].

Сегодня интеллектуальные алгоритмы применяются в самых разных аспектах [50] – от адаптивных образовательных платформ до автоматической проверки знаний. Однако особое значение они приобретают в системах управления учебной деятельностью. Здесь ИИ способен взять на себя не только рутинные операции, но и функции, требующие анализа большого количества параметров и принятия решений на их основе. Одним из таких направлений является генерация расписания занятий.

В организациях дополнительного образования, особенно в таких сферах как IT, администрирование процесса обучения требует учёта множества факторов: наличия свободных преподавателей с нужной квалификацией, особенностей графика учеников, истории предыдущих занятий, текущих тем и ограничений по локациям. При ручном планировании соблюдение всех этих условий становится практически невозможным, особенно в условиях быстрого роста количества групп и модулей.

Применение искусственного интеллекта позволяет моделировать и анализировать эти взаимосвязи гораздо глубже. В отличие от простых алгоритмов или шаблонов, интеллектуальные модели способны учитывать контекст, обрабатывать исторические данные и адаптироваться к новым условиям. Например, система может автоматически подобрать преподавателя, основываясь не только на его компетенциях, но и на том, насколько часто он ранее работал с конкретной группой, насколько комфортен для него предложенный временной слот и соответствует ли он политике распределения нагрузки на преподавателя.

Важно и то, что интеллектуальные модели способны не просто выполнять жёстко заданную логику, а учиться на накопленном опыте, выявляя успешные паттерны назначения преподавателей и формируя предсказания на этой основе. Это делает такие решения особенно перспективными в условиях динамики, свойственной дополнительному образованию, где расписания часто меняются, а группы переходят от одного модуля к другому без длительных пауз.

Кроме того, системы на основе ИИ легко интегрируются с существующими CRM-платформами, в которых уже хранятся все необходимые данные. Это открывает возможности для автоматизации всего цикла планирования – от загрузки данных до формирования расписания и его визуализации в интерфейсе преподавателя или администратора. Такое решение позволяет существенно снизить нагрузку на административный персонал и минимизировать человеческий фактор при принятии решений.

Таким образом, использование ИИ в образовательных технологиях выходит за рамки теоретического интереса и становится практическим инструментом повышения эффективности и устойчивости учебного процесса [33]. В контексте очного IT-обучения детей применение искусственного интеллекта позволяет не только оптимизировать ресурсы, но и обеспечить высокое качество образовательных услуг, что становится особенно актуальным в условиях масштабирования и роста числа обучающихся [34].

1.7. Вывод по главе

В первой главе были рассмотрены ключевые особенности и ограничения, характерные для организации учебного процесса в сфере дополнительного IT-образования детей. Были проанализированы структурные и поведенческие аспекты формирования расписаний, включая специфику локаций, ресурсы преподавателей, модульную структуру программ и

характеристики учебных групп. Установлено, что традиционные системы автоматизации недостаточно гибки и плохо адаптированы к условиям динамичного и масштабируемого образовательного процесса, что приводит к необходимости разработки специализированного интеллектуального решения.

Обзор существующих подходов к генерации расписаний показал, что классические и эвристические методы либо слишком ограничены, либо требуют значительного ручного участия и не способны учитывать все параметры в совокупности. Использование искусственного интеллекта, в частности графовых нейронных сетей и мультиагентных систем, позволяет сформировать адаптивную и обучающуюся модель, способную принимать решения на основе структуры данных и логики взаимодействия между участниками образовательного процесса.

Таким образом, в качестве методологической базы в данной работе обоснован выбор гибридного подхода, объединяющего GNN и MAS, что позволит не только повысить точность и устойчивость формирования расписания, но и адаптировать систему под реальные условия.

ГЛАВА 2. ОПИСАНИЕ МАТЕРИАЛОВ И МЕТОДОВ

2.1. Постановка задачи и архитектура решения

На основе анализа предметной области сформулирована цель данной работы – создание интеллектуальной модели, предназначенной для автоматизированного формирования расписания в системе очного дополнительного образования детей в сфере информационных технологий. Модель должна учитывать широкий спектр факторов: компетенции преподавателей, доступность учебных локаций, структуру учебных модулей, а также текущие и исторические связи между участниками процесса.

Задача сводится к корректному распределению преподавателей по заранее запланированным занятиям с учётом разнообразных ограничений. Например, необходимо учитывать, может ли преподаватель вести ту или иную тему, свободен ли он в заданное время, не превышает ли текущая нагрузка установленные лимиты, есть ли у него положительный опыт взаимодействия с группой, соответствует ли его рейтинг требованиям конкретной локации. Помимо этого, требуется поддерживать логическую непрерывность модульного обучения, избегать конфликтов по времени и пространству, а также учитывать мягкие предпочтения, такие как преемственность преподавателя в группе или удобные временные слоты.

Задача решается с использованием гибридной архитектуры, в которой объединены графовая нейронная сеть и мультиагентная система. Первая из них обучается на данных о ранее проведённых занятиях и позволяет выявлять скрытые закономерности в распределении преподавателей. Она рассматривает всю структуру связей между сущностями – от тем, по которым может вести преподаватель, до истории его работы с определённой группой или участия в занятиях на конкретной локации. В результате она формирует вероятностную оценку того, насколько хорошо тот или иной преподаватель подходит для

конкретного занятия.

Однако GNN не учитывает всех контекстных ограничений, особенно тех, которые трудно формализовать в виде признаков. Именно поэтому в системе предусмотрен второй компонент – мультиагентная система. Здесь каждая ключевая сущность представлена в виде агента, обладающего своими параметрами, целями и правилами поведения. Например, преподаватель знает о своей занятости, максимально допустимом количестве часов и нежелательных днях. Группа – о своём привычном времени занятий и преподавателях, с которыми ранее работала. А локация – о допустимом числе одновременно проводимых занятий и минимальном рейтинге педагогов.

Мультиагентная система получает рекомендации от GNN и проверяет их на соответствие актуальным условиям. Если назначение допустимо, оно принимается. Если нет – агентская система инициирует подбор альтернативного варианта, соблюдающего все заданные ограничения. Таким образом, GNN формирует «предпочтительный» вариант, а MAS выполняет роль согласующего механизма, обеспечивая реалистичность и согласованность расписания.

Такое разделение обязанностей позволяет объединить сильные стороны обеих методик: способность нейросети обобщать опыт и находить шаблоны – с логической строгостью и интерпретируемостью агентной модели. В результате получается архитектура, способная решать сложную задачу расписания с учётом большого числа факторов, включая как формальные зависимости, так и поведенческие сценарии.

2.2. Сбор данных

Разработка интеллектуальной модели генерации расписания невозможна без качественного и структурированного набора данных, отражающего реальную работу образовательной организации. В рамках

проекта использовались данные из различных источников, охватывающие все ключевые сущности: преподавателей, группы, учеников, модули, занятия и локации. Несмотря на то, что в текущей реализации проекта данные представлены в виде таблиц, в реальной системе они формируются и хранятся в базе данных PostgreSQL. Структура таблиц полностью соответствует шаблонам, использованным в модели, что обеспечивает совместимость и масштабируемость решений.

Основным источником информации о составе групп и учащихся является внутренняя CRM-система. В ней поддерживается актуальное распределение учеников по группам, фиксируется история переходов между модулями и сохраняются все параметры, связанные с группами: размер, уникальный идентификатор, закреплённая локация и временной слот. Эти данные регулярно синхронизируются с базой данных, где представлены в виде отдельных таблиц. Таким образом, CRM выступает в роли основного центра учёта учебного контингента, а база данных – как слой хранения и интеграции с аналитической моделью.

Информация о преподавателях также хранится в базе данных и поддерживается в актуальном состоянии через личный кабинет менеджера. При трудоустройстве каждого нового педагога создаётся персональная карточка, в которую вносятся ключевые параметры: навыки (в виде идентификаторов учебных тем, которые он может вести), возраст, опыт работы, расписание доступности, история занятий, статус занятости, максимальная нагрузка и другие характеристики. Менеджер может в любое время редактировать или дополнять карточку, а также отмечать временные периоды недоступности преподавателя, например, при уходе в отпуск. Такая структура позволяет не только собирать актуальные данные, но и гибко реагировать на изменения в кадровом составе.

Данные о модулях, в отличие от динамичных сущностей, загружаются в систему вручную и являются предопределёнными. Каждый модуль описывает

конкретную тему, включает длительность в неделях, уровень сложности, список обязательных тем-предшественников и другие параметры. Эта информация вносится единожды при создании учебной программы и может быть дополнена в случае расширения перечня направлений.

Особое внимание уделяется данным о прошедших занятиях, поскольку они лежат в основе обучения графовой нейросети. Исторические сведения за последние два года (примерно 10 000 записей о проведенных занятиях в г.Екатеринбург в период 2023–2024 год) были изначально представлены в виде таблицы Google Sheets, где фиксировались дата занятия, группа, преподаватель, тема, номер по порядку и другие параметры. Эти данные были преобразованы в структурированную таблицу базы данных, обеспечив совместимость с другими источниками. Кроме того, в реальной системе новая информация добавляется автоматически после подтверждения сгенерированного расписания менеджером, что позволяет со временем увеличивать обучающую выборку модели.

Локации, на которых проходят занятия, также создаются через интерфейс личного кабинета. Каждой локации присваивается адрес, уникальный идентификатор, ограничения по максимальному количеству групп, технические характеристики и минимальный допустимый рейтинг преподавателя. Такая детализация необходима для корректной фильтрации вариантов во время распределения преподавателей и обеспечения соответствия условиям проведения занятий.

Таким образом, процесс сбора данных в проекте представляет собой интеграцию множества источников – от автоматизированных систем учёта до исторических таблиц. Все данные проходят унификацию и попадают в централизованную базу, откуда впоследствии извлекаются для построения графа и обучения модели. Это обеспечивает не только полноту картины, но и возможность последующей автоматической актуализации данных при внедрении решения в реальную образовательную среду.

2.3. Структура графа и формирование признаков

Графовая нейронная сеть, лежащая в основе предлагаемого решения, требует предварительного представления всех сущностей предметной области в виде ориентированного графа. Такой подход позволяет отразить сложные взаимосвязи между участниками учебного процесса, а также учесть структурные зависимости и ограничения при генерации расписания. Построение графа выполняется на основе собранных данных, описанных ранее, и включает в себя как выбор типов вершин и рёбер, так и формирование признаков для каждого элемента.

Вершинами графа являются ключевые сущности системы: преподаватели, группы, модули (темы), локации и занятия (прошедшие или запланированные). Каждая вершина сопровождается набором признаков, отражающих её свойства, например:

— Преподаватель – возраст, стаж, рейтинг, список компетенций и предельная нагрузка;

— Группы – размер, текущее местоположение, тема и привязанность к преподавателю;

— Модуль – длительность, сложность, порядковый номер в цепочке;

— Локация – пропускная способность, ограничения по рейтингу и количеству групп;

— Занятие – дата, принадлежность к группе и преподавателю, пробный статус.

Связи между вершинами задаются рёбрами с направлением, отражающим смысл отношения. Например, рёбра между группой и преподавателем могут обозначать факт предыдущего взаимодействия (если преподаватель вёл занятия в данной группе), между преподавателем и модулем – наличие компетенции, между группой и модулем – текущую учебную тему. Каждое ребро также может содержать числовой признак –

например, количество совместных занятий, коэффициент привязанности или силу связи (на основе частоты взаимодействия) [12].

Таким образом, граф включает следующие основные связи:

— преподаватель – группа (на основе истории занятий);

— преподаватель – модуль (по списку навыков);

— группа – модуль (текущая тема обучения);

— группа – локация (место проведения занятий);

— преподаватель – локация (по фильтрации доступных филиалов);

— занятие – группа, занятие – преподаватель, занятие – модуль (прошлые занятия);

— модуль – модуль (если один зависит от другого).

Поскольку GNN использует как признаки вершин, так и структуру связей, важно обеспечить достаточную информативность каждого элемента. В качестве признаков используются числовые, категориальные и бинарные значения, преобразованные в числовой формат. Например, день недели кодируется как число от 0 до 6, булевы параметры (доступность, пробное занятие) – в 0/1, а категориальные признаки (например, имя темы) – через one-hot или embedding-представления.

Признаки нормализуются и объединяются в векторы фиксированной длины, что позволяет подавать их на вход графовой модели. Отдельное внимание уделяется репрезентации истории взаимодействий: она агрегируется в виде суммы, среднего или взвешенного коэффициента, отражающего силу связи между узлами.

В результате получается гетерогенный граф, в котором каждая вершина и каждое ребро содержит осмысленные характеристики, релевантные задаче выбора преподавателя для конкретного занятия. Такая структура делает возможным обучение модели, способной выявлять закономерности в предыдущих назначениях и применять их к новым ситуациям.

2.4. Описание графовой нейронной сети (GNN)

Графовая нейронная сеть (GNN) [15] выступает в качестве центрального аналитического модуля, предназначенного для предсказания наиболее подходящих назначений преподавателей на запланированные занятия. В отличие от традиционных нейросетей, работающих с плоскими таблицами или изображениями, GNN способна учитывать сложную структуру связей между различными объектами в обучающей выборке. Это делает её особенно эффективной в задачах, где сущности взаимодействуют между собой в контексте: как в случае с группами, преподавателями, модулями и локациями.

2.4.1 Общая идея работы GNN

Основной принцип GNN заключается в том, что каждая вершина (узел) графа получает представление (embedding), которое формируется не только на основе её собственных признаков, но и на основе информации от соседних узлов [48]. На каждой итерации (или слое) модель агрегирует признаки соседей и обновляет своё состояние [52]. В результате формируется векторное представление узла, отражающее как его свойства, так и его контекст в структуре графа [24].

Применительно к данной задаче, это означает, что представление группы будет включать информацию о её текущем модуле, привязке к локации и истории работы с преподавателями. Аналогично, embedding преподавателя будет содержать не только данные о его навыках и рейтинге, но и агрегированную информацию о группах, с которыми он ранее работал.

2.4.2 Архитектура модели

В данной работе используется модифицированный вариант GNN, основанный на слоистой архитектуре типа GraphSAGE[26] или GAT (Graph Attention Network). Эти подходы позволяют агрегировать информацию от соседей с учётом важности (веса) каждой связи. Это особенно актуально в условиях, когда связи между узлами имеют различную силу – например,

частота взаимодействия преподавателя с группой или важность темы в учебной последовательности.

Входом в модель является граф, построенный на основе реальных данных: каждая вершина содержит числовой вектор признаков, а рёбра могут содержать дополнительные мета-признаки (например, количество совместных занятий). На выходе модель формирует предсказания по вероятности назначения преподавателя на конкретную группу и модуль. Для этого используется механизм слияния представлений преподавателя и группы с последующей классификацией (например, через многослойный перцептрон) или механизм оценки совместимости (scoring function).

2.4.3 Формат входных и выходных данных

В процессе обучения каждая обучающая пара (запись о проведённом занятии) представляется в виде подграфа, включающего связанные узлы – преподавателя, группу, модуль и локацию. Модель учится на основе метки – идентификатора преподавателя, фактически проведшего занятие. Таким образом, задача сводится к задаче многоклассовой классификации или ранжирования кандидатов по вероятности их выбора в данном контексте.

Выходом модели является либо класс с максимальной вероятностью (ID преподавателя), либо ранжированный список кандидатов, что особенно полезно при интеграции с мультиагентной системой – она может принять или отклонить решение на основе внешних ограничений.

2.4.4 Обучение и оптимизация

Обучение модели проводится на исторических данных, включающих сведения о более чем 10 000 занятиях, проведённых за последние два года. Используется стандартная функция потерь – кросс-энтропия в случае классификации или margin-based loss при ранжировании. Модель оптимизируется с использованием алгоритма Adam или его модификаций, с регуляризацией для предотвращения переобучения [20].

Для повышения качества используются техники:

— подвыборки соседей (neighbor sampling);

— батчинг по подграфам;

— нормализация признаков;

— раннее прекращение обучения (early stopping) по валидационной метрике.

Таким образом, графовая нейронная сеть позволяет сформировать обучающуюся модель, способную учитывать многосвязную структуру предметной области и адаптироваться к различным контекстам. GNN формирует основу для интеллектуального анализа, на которой впоследствии мультиагентная система выполняет контекстную фильтрацию и утверждение решений.

2.5. Реализация мультиагентной системы (MAS)

Мультиагентная система (MAS) в архитектуре предлагаемого решения играет роль контекстного уровня управления, обеспечивающего гибкость, интерпретируемость и соблюдение всех логических и организационных ограничений, характерных для образовательной среды. В отличие от графовой нейронной сети, которая занимается обобщением данных и выявлением закономерностей, MAS оперирует на уровне поведения отдельных сущностей, моделируя их интересы, ограничения и правила взаимодействия.

2.5.1 Агентная модель

Каждая ключевая сущность предметной области – преподаватель, группа, локация, занятие [47] – представлена в системе в виде отдельного агента, обладающего набором характеристик и логикой принятия решений. Агенты могут как инициировать действия (например, занятие "ищет" преподавателя), так и реагировать на предложения других агентов (преподаватель "отклоняет" или "принимает" назначение).

Преподаватель, как агент, хранит информацию о своих навыках,

текущем расписании, максимальной нагрузке, статусе занятости, днях недоступности и истории работы с группами. Группа – о времени занятий, текущем модуле, составе, локации и предпочтительном преподавателе. Локация – о вместимости, минимальном рейтинге преподавателей и загруженности по слотам. А запланированное занятие, как отдельный агент или задача, выполняет координирующую роль – инициирует подбор кандидата и проверяет выполнение всех условий.

2.5.2 Принципы взаимодействия агентов

Работа MAS [13] строится по принципу координации и переговоров [42]. Когда модель GNN формирует список кандидатов для конкретного занятия, мультиагентная система запускает цикл оценки и фильтрации. При этом мультиагентная система реализует компенсирующий механизм: если первичный кандидат, рекомендованный моделью, не проходит проверку по ограничениям (например, из-за пересечения по времени, отсутствия компетенции или перегрузки), система автоматически переходит к следующему варианту, сохраняя корректность назначения. Это делает процесс устойчивым к ложным рекомендациям и позволяет интегрировать рекомендательную природу GNN с логикой MAS. Каждый агент проверяет, соответствует ли предложение его внутренним правилам [43]: преподаватель анализирует занятость и лимиты, группа – привычный состав и расписание, локация – текущую загрузку и требования к рейтингу.

Если назначение удовлетворяет всем условиям, оно принимается и фиксируется. В противном случае MAS либо запрашивает следующего кандидата, либо инициирует перераспределение – например, предлагает другой слот, ищет замену или делегирует занятие другому преподавателю с сопоставимыми характеристиками.

Особенностью MAS является способность работать с мягкими ограничениями. Так, система может учитывать приоритетность преподавателя на основе его истории работы с группой, даже если формально назначение

возможно и без него [5]. Это позволяет сохранять преемственность и повышать удовлетворённость учеников, особенно в младшем возрасте.

2.5.3 Внедрение логики и реализация

С точки зрения реализации, поведение агентов описывается в виде функций или классов с определённой логикой. Каждый агент способен:

— принимать решение о согласии или отказе от предложения;

— обновлять своё состояние после назначения;

— коммуницировать с другими агентами через интерфейс координации.

MAS построена таким образом, что способна работать независимо от внутренней логики GNN. Это означает, что даже при частично или полностью ручном формировании начального списка преподавателей система сможет выполнять проверку и фильтрацию на основе заданных правил.

Реализация взаимодействия агентов может быть как централизованной (через управляющий контроллер), так и распределённой – когда агенты взаимодействуют напрямую через сообщения. В рамках текущей версии проекта реализована централизованная координация, позволяющая проще отслеживать конфликты и проводить отладку логики.

2.5.4 Роль MAS в архитектуре

MAS выполняет важную функцию: она переводит вероятностный выход GNN в конкретное, допустимое и устойчивое расписание. Благодаря этому модель становится практически применимой – она не просто "предсказывает", а предлагает решения, которые можно использовать без дополнительной ручной правки.

Кроме того, мультиагентная система легко масштабируется: с ростом количества групп и преподавателей добавление новых агентов не требует изменения архитектуры, а только подключения новых экземпляров с заданными параметрами.

Таким образом, мультиагентная система обеспечивает контекстно-зависимое, адаптивное принятие решений на основе рекомендаций GNN,

соблюдая при этом все организационные, логистические и педагогические требования образовательной среды. Именно MAS превращает аналитическую модель в полноценный инструмент для генерации реального расписания, пригодного к использованию без дополнительной доработки.

2.6. Интеграция GNN и MAS: логика взаимодействия

Гибридная архитектура, схема которой показана на рисунке 1, лежащая в основе предлагаемого решения, сочетает в себе два комплементарных подхода [41]: обучение на данных с помощью графовой нейронной сети (GNN) и адаптивное принятие решений с участием мультиагентной системы (MAS).

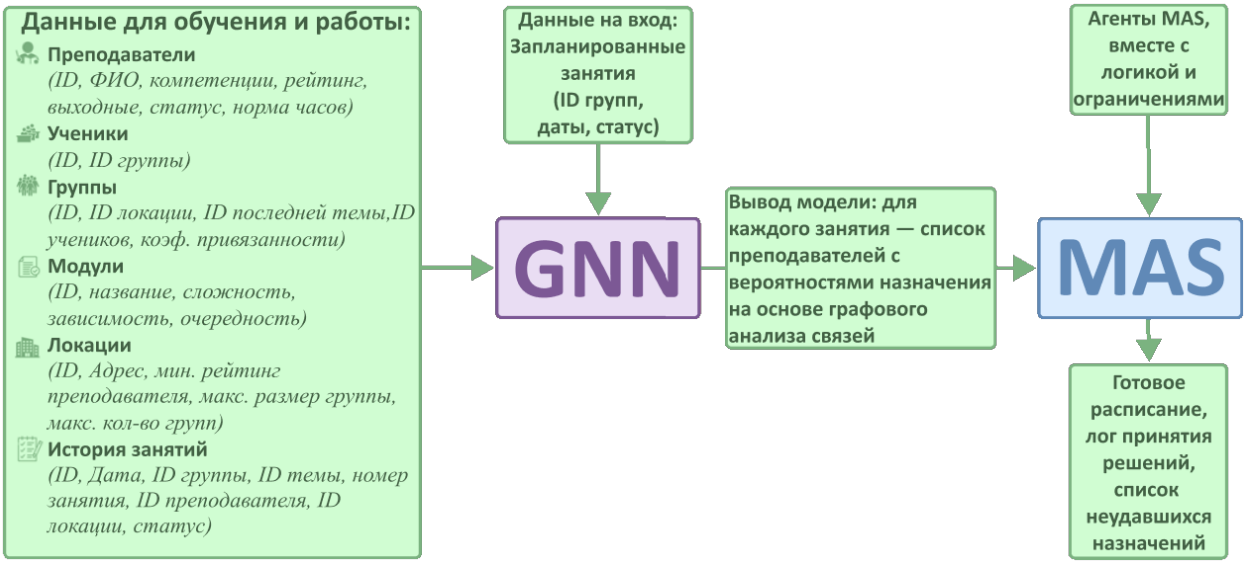

Рисунок 1 – общая архитектура GNN + MAS

Их совместная работа обеспечивает не только высокую точность при выборе преподавателей, но и надёжное соблюдение всех ограничений и бизнес-правил, характерных для образовательной среды.

2.6.1 Роли компонентов в системе

GNN выполняет роль аналитического ядра. Она обучается на исторических данных о занятиях, выявляя закономерности в том, какие преподаватели чаще всего проводили занятия в конкретных группах, по

определённым модулям и в заданных условиях. На основании этого обучения модель формирует ранжированный список кандидатов для каждого нового занятия. Этот список не учитывает всех ограничений напрямую, но отражает вероятностную "близость" преподавателя к контексту занятия – с точки зрения структуры графа, истории и признаков.

MAS, в свою очередь, является координирующим слоем, который получает на вход рекомендации GNN и проверяет их на соответствие реальным условиям. Она моделирует поведение участников образовательного процесса и обеспечивает соблюдение жёстких и мягких ограничений: занятости преподавателя, допустимой нагрузки, преемственности преподавания, ограничений по локациям и времени. MAS может как принять предложение GNN, так и отклонить его, запрашивая альтернативы.

2.6.2 Порядок взаимодействия

Процесс генерации расписания строится по следующей логике:

1. GNN получает граф, сформированный на основе текущего состояния системы: активные группы, преподаватели, модули, локации, история занятий;

2. Для каждого запланированного занятия модель формирует предсказание: список преподавателей, отсортированных по вероятности успешного назначения;

3. Рекомендации передаются в MAS. Сначала рассматривается кандидат с наивысшей оценкой;

4. MAS выполняет проверку по ряду критериев: доступность преподавателя в конкретный день и время, отсутствие конфликтов, соблюдение лимита по часам, соответствие теме, соответствие требованиям локации;

5. Если кандидат удовлетворяет всем условиям, MAS фиксирует назначение, обновляет состояние агентов и переходит к следующему занятию;

6. Если кандидат не подходит, система переходит к следующему в

списке. Если ни один кандидат из списка не прошёл фильтр, MAS может инициировать поиск вручную, предлагать отложить занятие или подключить менеджера к принятию решения.

2.6.3 Преимущества такого подхода

Такое разделение ответственности между компонентами системы позволяет добиться устойчивости и гибкости. GNN обеспечивает адаптацию к данным и выявление скрытых закономерностей, а MAS контролирует корректность и применимость решений [14], [21]. Это особенно важно в условиях динамичного расписания, когда ситуация может быстро изменяться: преподаватель может уйти в отпуск, группа может сменить тему, а локация – временно закрыться [9].

Кроме того, архитектура масштабируема и расширяема. Например, можно внедрить механизм обратной связи, при котором MAS будет возвращать GNN информацию о том, какие предложения были приняты, а какие – отклонены. Это позволит дообучать модель и повышать точность предсказаний на следующих итерациях.

2.7. Обработка расписаний: генерация, фильтрация, утверждение

Одним из ключевых требований к системе генерации расписания является не только интеллектуальное распределение преподавателей, но и возможность внедрения решения в реальный образовательный процесс с учётом всех внутренних регламентов, логистики и человеческого фактора. Поэтому помимо этапов предсказания и фильтрации, в архитектуре модели предусмотрен полноценный контур финальной валидации, включающий подтверждение менеджером. Этот процесс состоит из трёх взаимосвязанных стадий: генерации, фильтрации и утверждения.

2.7.1 Генерация расписания

Первый этап – автоматическое формирование предварительного расписания. На этом этапе модель GNN обрабатывает запланированные занятия, сформированные заранее по параметрам групп, модулей и временных слотов. Для каждого занятия нейросеть предлагает список подходящих преподавателей, опираясь на обученную структуру графа. Эти предложения ранжированы по степени релевантности и вероятности успешного назначения.

На выходе GNN формируется первичная карта назначений: каждый слот получает кандидата, который статистически наилучшим образом подходит по истории взаимодействий, компетенциям, теме и другим признакам.

2.7.2 Фильтрация с помощью MAS

Полученные рекомендации поступают на вход мультиагентной системе, которая выполняет контекстную фильтрацию и проверку всех ограничений. MAS анализирует занятость преподавателей, ограничения по нагрузке, соответствие рейтингов требованиям локаций, предпочтения групп и историю взаимодействий. Если назначение соответствует всем критериям, оно подтверждается системой автоматически. В противном случае MAS обращается к следующему кандидату из списка или передаёт сигнал на дополнительную обработку.

На этом этапе также реализуется механизм учёта мягких предпочтений, таких как сохранение преподавателя в группе, последовательность ведения модулей, временная близость между занятиями одного преподавателя и т. п. Это позволяет получать расписание, не только технически корректное, но и устойчивое с педагогической точки зрения.

2.7.3 Утверждение и ручная корректировка

Несмотря на высокую степень автоматизации, в финальной фазе система передаёт полученное расписание на утверждение менеджеру, выполняющему роль контролёра. Менеджер может просмотреть предложенные назначения через административный интерфейс, внести коррективы вручную или

инициировать повторную генерацию для отдельных групп или временных слотов.

Если расписание подтверждается, оно фиксируется в базе данных, получает статус актуального и становится доступным для всех участников процесса: преподавателей, кураторов, родителей и учеников. Информация также попадает в таблицу истории занятий, которая используется для последующего дообучения модели.

Данный этап особенно важен для интеграции модели в бизнес-процесс компании, поскольку сохраняет возможность управленческого контроля и позволяет учитывать факторы, которые пока не формализованы в виде признаков (например, личные договорённости, внештатные ситуации и пр.).

2.8. Программная реализация и стек технологий

Для реализации системы автоматизированного формирования расписания была выбрана модульная архитектура, предполагающая раздельную разработку компонентов графовой нейронной сети, мультиагентной системы и подсистемы взаимодействия с источниками данных. Такой подход обеспечил гибкость разработки, масштабируемость, а также удобство при интеграции в существующую образовательную инфраструктуру.

Реализация проекта будет на языке программирования Python, как наиболее универсальном и широко поддерживаемом инструменте для разработки систем на основе машинного обучения. Выбор языка обусловлен наличием обширной экосистемы готовых библиотек, поддержкой работы с графами, нейросетями, базами данных и взаимодействием с внешними API.

В качестве библиотеки для построения и обучения графовой нейронной сети использовалась PyTorch Geometric (PyG) – расширение над фреймворком PyTorch, специально ориентированное на работу с графами. Данная

библиотека предоставляет удобный интерфейс для построения гетерогенных графов, управления батчами подграфов, агрегации признаков и обучения моделей, таких как GraphSAGE, GAT и GCN [30].

Для построения и координации логики мультиагентного взаимодействия применялась комбинация объектно-ориентированного программирования и событийной архитектуры. Каждый агент реализован как Python-класс, обладающий внутренним состоянием и методами принятия решений. Логика взаимодействия между агентами реализована через управляющий контроллер, обеспечивающий централизованную фильтрацию и координацию в процессе назначения преподавателей.

Хранение и управление структурированными данными реализовано на базе реляционной СУБД PostgreSQL. Система оперирует множеством взаимосвязанных таблиц: преподаватели, группы, занятия, модули, локации и пр. Все данные поступают из различных источников (внутренняя CRM-система, Google Таблицы, административные панели) и объединяются в единый слой хранения. Для взаимодействия с базой данных использовалась библиотека SQLAlchemy, обеспечивающая удобную ORM-обвязку и поддержку сложных SQL-запросов.

Для организации экспериментов, обучения модели и тестирования применялись такие библиотеки, как:

— NumPy и Pandas – для работы с табличными данными и векторными представлениями;

— Scikit-learn – для базовых метрик, валидации и предобработки;

— Matplotlib – для визуализации результатов обучения и анализа качества модели.

Визуальное отображение связей между сущностями (ER-диаграммы, графы взаимодействия) было реализовано с использованием библиотеки Graphviz, которая позволила создавать наглядные схемы структуры базы данных и логики агентов.

Проектирование системы происходит с учётом возможности расширения: добавление новых типов агентов, расширение структуры графа, подключение новых источников данных, адаптация под другие форматы обучения (включая онлайн). Проект организован в виде Python-пакета с модульной структурой, что обеспечивает удобство при дальнейшем развертывании в среде Docker и сопровождении.

2.9. Подготовка обучающей выборки и схема обучения модели

Одним из ключевых этапов в построении графовой нейросетевой модели является подготовка обучающей выборки, способной адекватно отразить закономерности, существующие в данных. Поскольку модель опирается на исторические данные о проведённых занятиях, особое внимание было уделено структурированию, очистке и приведению этих данных к графовому формату, пригодному для подачи в GNN.

2.9.1 Источник и структура данных

Основой для обучения стали сведения о более чем 10 000 занятиях, проведённых за последние два года в филиалах образовательных центров «KIBERone» в г. Екатеринбург. Эти данные были изначально представлены в виде Google-таблиц, где вручную фиксировались дата занятия, группа, преподаватель, тема, номер занятия, локация и дополнительная информация о пробных занятиях. После первичной очистки и стандартизации, данные были перенесены в PostgreSQL и использованы для формирования связного графа.

Каждая запись о занятии интерпретировалась как положительная обучающая пара «группа–модуль–преподаватель», отражающая успешное назначение. Путём агрегации по ID группы, ID темы (модуля), ID преподавателя и другим параметрам была сформирована обучающая матрица взаимодействий, которая затем использовалась при генерации обучающих подграфов.

2.9.2 Формирование графа для обучения

Для каждого занятия создавался подграф, включающий:

— вершину группы, её признаки и связи (с локацией, текущей темой, историей взаимодействий с преподавателями);

— вершину преподавателя (с признаками и связями с группами, модулями, локациями);

— вершину модуля (тема занятия);

— вершину локации, где проходило занятие;

— ребра, отражающие реальные связи между участниками (например, «преподаватель вёл данную группу 8 раз», «модуль входит в компетенции преподавателя», «преподаватель уже работает на этой локации» и др.).

Негативные примеры (то есть преподаватели, которые не были назначены на занятие, но могли быть потенциально назначены) формировались случайной подстановкой преподавателей с подходящими компетенциями. Это позволило обучать модель в формате задачи бинарной классификации или ранжирования, где она должна отличать корректное назначение от некорректного.

2.9.3 Разделение на выборки

Данные были разделены на три подвыборки:

— обучающая выборка (~70%) – для основного процесса обучения;

— валидационная выборка (~15%) – для оценки качества на каждом этапе обучения и настройки гиперпараметров;

— тестовая выборка (~15%) – для окончательной проверки модели после обучения.

Разделение выполнялось с учётом хронологии – таким образом, данные последнего полугодия были выделены под тестирование, имитируя реальный сценарий применения модели на новых, ранее не виденных примерах.

2.9.4 Обучение и схема работы модели

Обучение модели проводилось в фреймворке PyTorch Geometric, на основе данных, описанных в Приложении А. В качестве архитектуры использовалась модифицированная версия GraphSAGE, позволяющая обрабатывать большие графы с выборкой соседей. Слои модели последовательно агрегировали признаки соседей, а итоговое представление узлов использовалось для вычисления функции совместимости между преподавателем и конкретным занятием.

Использовалась функция потерь cross-entropy в задаче бинарной классификации или margin ranking loss в задаче ранжирования, в зависимости от выбранного сценария. Оптимизация производилась с помощью Adam, с шагом обучения, подбираемым на валидации. Для предотвращения переобучения использовались dropout, нормализация признаков, раннее прекращение (early stopping).

Модель обучалась в несколько эпох до стабилизации качества по валидационной метрике. В качестве основных метрик использовались accuracy, precision@k, top-k accuracy [27], а также частота успешного назначения преподавателя в тестовых сценариях генерации расписания.

2.10. Вывод по главе

Во второй главе была рассмотрена архитектура системы автоматизированного формирования расписаний, основанной на сочетании графовой нейронной сети (GNN) и мультиагентной системы (MAS). Такой гибридный подход позволяет объединить преимущества анализа структурированных исторических данных и гибкости принятия решений в реальном времени с учётом множества ограничений и бизнес-правил.

Были подробно описаны источники и структура данных, используемых в системе: информация о преподавателях, группах, локациях, модулях и уже

проведённых занятиях. Эти данные собираются из CRM-системы, административного интерфейса и внешних таблиц, преобразуются в единый граф, где каждая вершина и каждое ребро содержат осмысленные признаки. Подготовленный граф поступает на вход GNN, которая обучается на исторических примерах назначения преподавателей и формирует вероятностные рекомендации.

Для обеспечения корректности расписаний поверх модели действует мультиагентная система, моделирующая поведение реальных участников процесса – преподавателей, групп и локаций. MAS выполняет фильтрацию и проверку решений GNN на соответствие требованиям доступности, нагрузки, предпочтений и логистических ограничений. Это позволяет избежать конфликтов, обеспечить преемственность и повысить качество образовательного процесса.

Система спроектирована с учётом масштабируемости и интеграции в существующую инфраструктуру образовательной организации.

Полученная архитектура готова к практическому применению и масштабированию. В следующей главе будет рассмотрен процесс обучения модели, проведённые эксперименты, результаты тестирования, а также анализ поведения системы в реальных сценариях планирования.

ГЛАВА 3. РЕАЛИЗАЦИЯ МОДЕЛИ И АРХИТЕКТУРА СИСТЕМЫ

3.1. Средства разработки и структура проекта

Разработка программного решения осуществлялась с использованием языка программирования Python версии 3.11 и открытых библиотек для машинного обучения, построения графов и обработки табличных данных. Выбор Python обусловлен широким распространением, обширной экосистемой научных библиотек, а также поддержкой современных подходов к обучению нейронных сетей, включая графовые архитектуры.

Ключевым элементом системы является модуль графовой нейронной сети, отвечающий за прогнозирование наиболее подходящего преподавателя для каждой учебной группы. Он реализован с использованием библиотеки PyTorch Geometric, обеспечивающей эффективную работу с графовыми структурами и поддержку операций сверточных слоёв на графах. Модель использует трёхслойную архитектуру, включающую графовые свёртки, нормализацию признаков, функции активации ReLU и регуляризацию через Dropout. Такая структура позволяет выявлять скрытые зависимости между сущностями образовательного процесса, включая преподавателей, группы, темы занятий и локации.

Механизм мультиагентного взаимодействия реализован в виде независимых программных компонентов, моделирующих поведение отдельных участников расписания – преподавателей, учебных групп, локаций и учебных модулей. Каждый агент обладает собственным набором правил принятия решений, основанных на доступности, квалификации, истории взаимодействий и других параметрах. Центральный управляющий компонент обеспечивает координацию агентов, обрабатывает входные данные, разрешает конфликты при планировании и формирует финальное недельное расписание.

Программная система построена с учётом принципов модульности, что

облегчает тестирование отдельных компонентов, повторное использование логики и масштабирование проекта. В рамках реализации предусмотрен полный цикл обработки: от импорта данных из информационной системы организации до формирования финального расписания с учётом ограничений и предпочтений.

Обучение и тестирование модели проводилось на локальной рабочей станции, оснащённой графическим процессором NVIDIA RTX 4060 и 16 ГБ оперативной памяти. Такая конфигурация обеспечивала достаточную производительность для обработки графов.

3.2. Реализация графовой нейронной сети

Графовая нейронная сеть (GNN), о играет в разработанной системе центральную роль, формируя интеллектуальную основу для принятия решений о распределении преподавателей по учебным группам. В рамках проекта была реализована модель на основе архитектуры GraphSAGE [26], которая на практике показала устойчивость к разреженным и динамическим графам, характерным для образовательной среды.

3.2.1 Принципы построения GNN-модели

В отличие от классических нейронных сетей, работающих с фиксированной структурой данных, графовая нейросеть способна обрабатывать граф – структуру, пример которой показан на рисунке 2, где узлы представляют сущности (например, преподавателей или группы), а рёбра – связи между ними (например, факт того, что преподаватель проводил занятия в этой группе). Это позволяет учитывать не только признаки конкретного узла, но и контекст, в котором он находится, что особенно важно в задаче генерации расписаний, где большинство решений опираются на взаимодействия между участниками образовательного процесса.

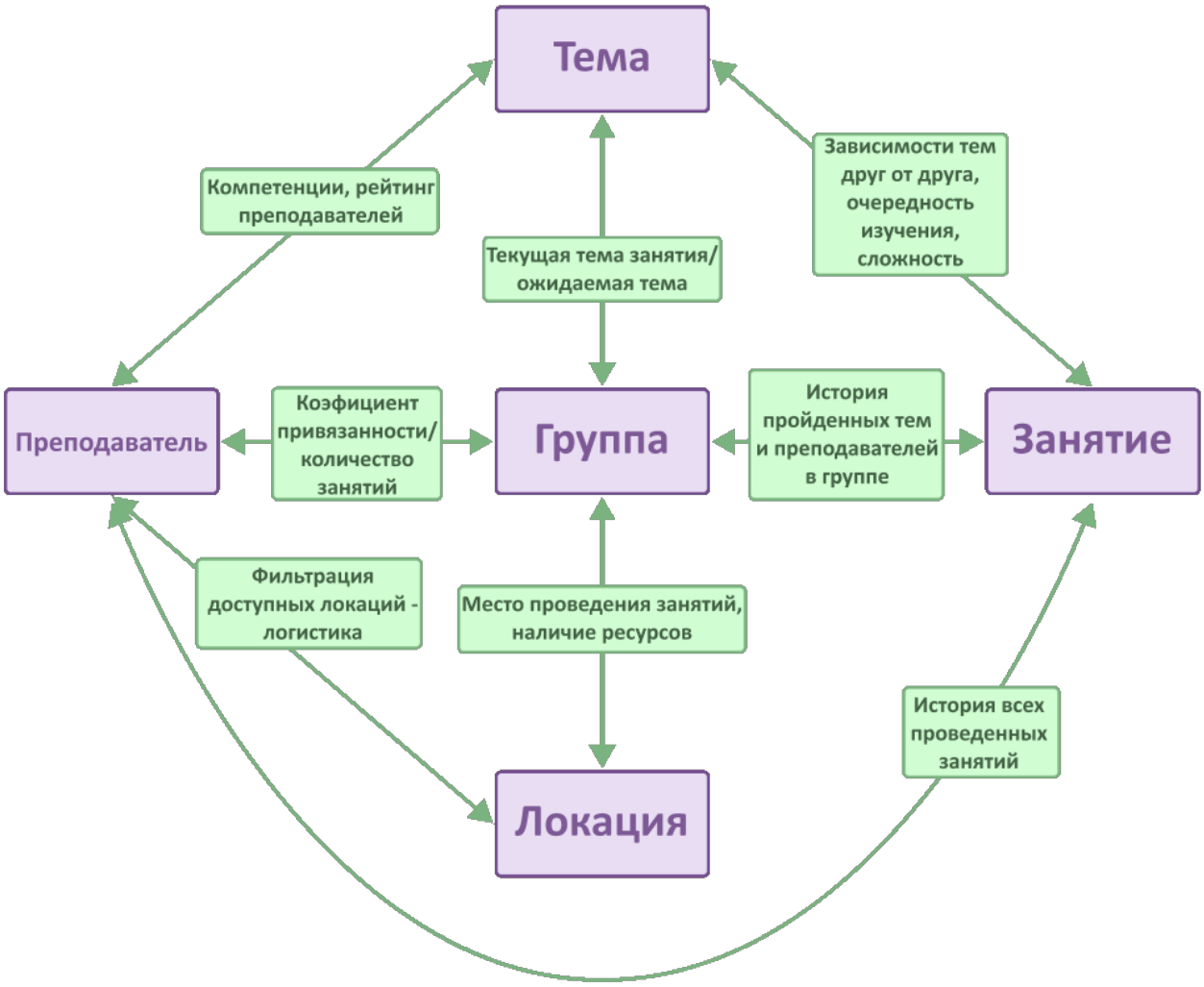

Рисунок 2 – схема графов GNN(GraphSAGE) нейросети

В реализованной модели применена модификация GraphSAGE – подхода, при котором каждый узел агрегирует признаки своих соседей с помощью обучаемой функции. Это позволяет обобщать информацию по соседним вершинам без необходимости перерасчёта всего графа, что особенно удобно при работе с частично обновляющимися данными.

3.2.2 Архитектура модели

Финальная архитектура графовой нейронной сети включает в себя три последовательно применяемых слоя агрегации с межслойной нормализацией и активацией ReLU:

— первый слой преобразует входные признаки всех узлов в скрытое пространство заданной размерности и начинает сбор информации о соседях;

— второй слой усиливает локальную агрегацию, позволяя учитывать косвенные связи и обогащать представление узла;

— третий слой завершает процесс агрегации, фиксируя итоговое представление (embedding) каждого узла.

Каждый слой сопровождается механизмом нормализации (BatchNorm), активацией ReLU и Dropout-регуляризацией, что позволяет повысить обобщающую способность модели и устойчивость к переобучению. На выходе сети установлен полносвязный слой, который преобразует эмбеддинг каждого узла в вектор вероятностей по всем возможным кандидатам–преподавателям. Таким образом, модель решает задачу многоклассовой классификации: по узлу типа «запланированное занятие» она должна определить наиболее подходящего преподавателя из заданного множества.

3.2.3 Реализация обучающего процесса

Обучение модели осуществлялось на исторических данных, содержащих информацию о прошедших занятиях, включая ID группы, тему, преподавателя, дату, а также сведения о локации и статуса пробного занятия. Эти данные трансформируются в граф, где рёбра создаются по принципу «преподаватель проводил занятие в группе», «группа изучала тему», «тема преподавалась в локации» и т. д. Такой подход позволяет моделировать реальную структуру образовательного процесса и использовать её как основу для обобщения.

Модель обучается в режиме минибатчей с использованием функции потерь CrossEntropyLoss, а в качестве оптимизатора выбран Adam [45]. Предсказания используются как вероятностное распределение, которое в дальнейшем интерпретируется как ранжированный список преподавателей–кандидатов. Эта информация поступает в мультиагентную систему, где уже осуществляется окончательный выбор с учётом бизнес-правил.

3.2.4 Гибкость и масштабируемость

Архитектура модели спроектирована с учётом потенциального масштабирования: за счёт разделения графа на слои и поддержки батчевой обработки возможно обучение и инференс на графах с десятками тысяч узлов. Также модель легко адаптируется под новые данные: появление новых групп, преподавателей или тем не требует полной перестройки архитектуры.

Таким образом, реализованная GNN-модель представляет собой универсальный компонент интеллектуальной системы планирования, способный выявлять закономерности в сложной связной среде и формировать предсказания, обоснованные структурой данных и их контекстом

3.3. Реализация мультиагентной системы

Мультиагентная система (MAS) в разработанном решении выполняет критически важную функцию – фильтрацию и проверку кандидатов, предложенных графовой нейронной сетью, с последующим принятием решения о назначении преподавателя на конкретное занятие. В то время как GNN формирует ранжированный список подходящих вариантов, именно MAS гарантирует соответствие бизнес-правилам и реальным ограничениям образовательной среды [26].

3.3.1 Архитектурная модель мультиагентной системы

MAS (рис. 3) реализована в виде совокупности агентов, каждый из которых представляет собой самостоятельную сущность с собственной логикой поведения, набором параметров и функцией принятия решений [43], [44]. Агенты взаимодействуют между собой через централизованную среду (координатор), которая управляет порядком симуляции, разрешает конфликты и формирует итоговое расписание [5].

В системе выделяются следующие типы агентов:

— агент преподавателя – содержит данные о компетенциях (темах),

рейтинге, доступности по времени, истории занятий, максимальной нагрузке, а также информацию о текущем статусе (активен/неактивен);

— агент группы – фиксирует ID группы, текущую тему, размер, расписание и историю преподавателей. Также учитываются предпочтения по преподавателю и стабильность состава;

— агент темы (модуля) – хранит данные о длительности темы, сложности, зависимостях и её месте в образовательной последовательности;

— агент локации – содержит параметры физического пространства: вместимость по группам и ученикам, требования к минимальному рейтингу преподавателя, привязку к определённым группам.

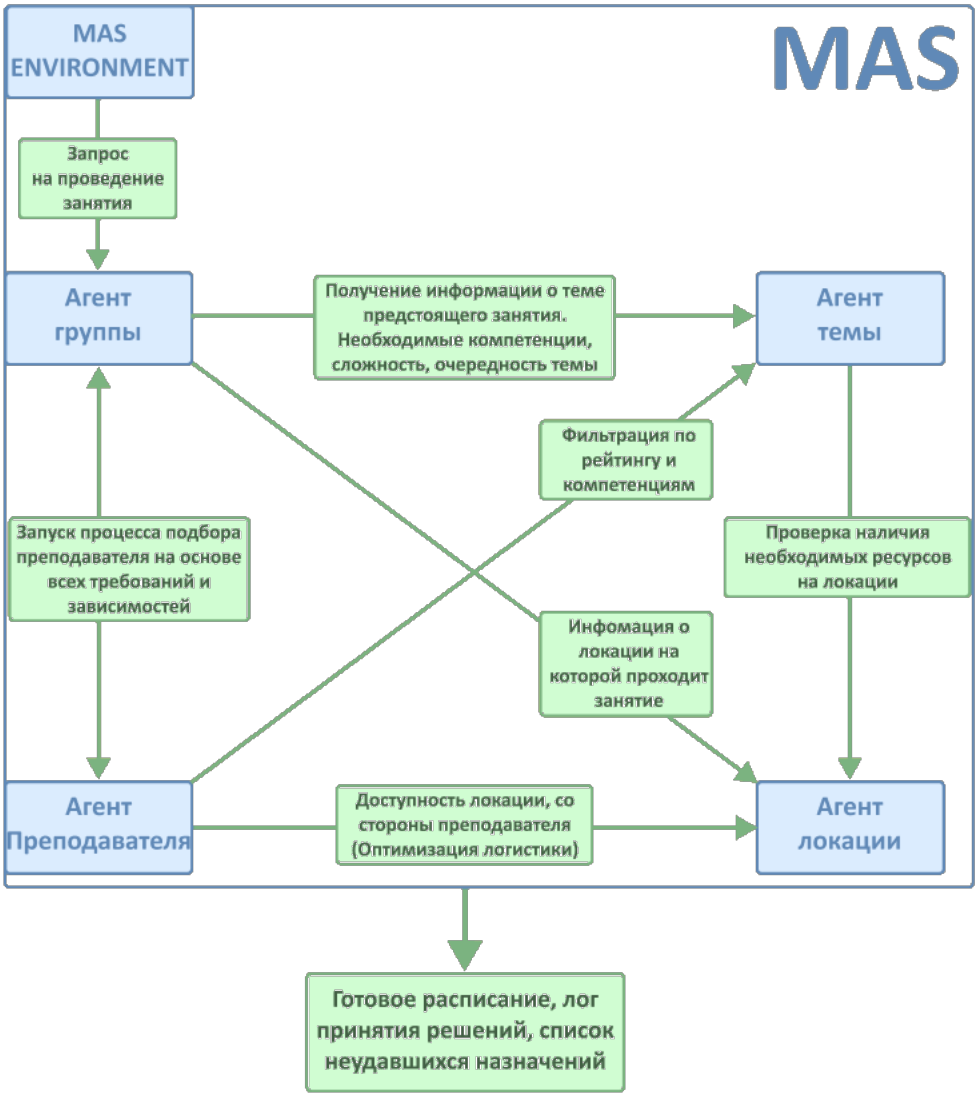

Рисунок 3 – схема работы мультиагентного взаимодействия

Центральный компонент системы – мультиагентная система, которая – управляет симуляцией, получает рекомендации от GNN, инициирует фильтрацию, проверяет соблюдение всех ограничений и принимает окончательное решение о назначении преподавателя.

Все агенты взаимодействуют только через эту среду, что соответствует архитектуре с централизованным контролем.

3.3.2 Логика взаимодействия и принятия решений

Процесс назначения преподавателя на занятие включает в себя следующие шаги:

1. Получение рекомендаций от GNN. Для каждой запланированной пары «группа–тема» GNN формирует список вероятных кандидатов;

2. Формирование контекста. Среда запрашивает у соответствующих агентов информацию об ограничениях: расписание группы и преподавателя, статус локации, компетенции и прочее;

3. Фильтрация кандидатов. Из списка GNN удаляются преподаватели, которые не владеют нужной темой, недоступны в указанный временной слот, не соответствуют требованиям по рейтингу, исчерпали максимальную нагрузку, имеют пересечения в расписании или статус «неактивен»;

4. Разрешение конфликтов. Если несколько кандидатов удовлетворяют условиям, среда применяет дополнительные эвристики (например, преподаватель с наибольшей историей в этой группе или с наименьшей текущей нагрузкой);

5. Фиксация назначения. Выбранный преподаватель закрепляется за группой, обновляется его расписание и история, занятие записывается в итоговое расписание.

Этот процесс повторяется итеративно для всех слотов планируемого периода, обеспечивая целостное расписание без пересечений, перегрузок и нарушений ограничений.

3.3.3 Гибкость и расширяемость MAS

Одним из ключевых преимуществ реализованной мультиагентной системы является её модульность. Каждый агент изолирован и может быть расширен новыми параметрами или логикой без изменения всей системы. Например, можно внедрить:

— агента онлайн группы;

— агента подменного преподавателя;

— агента менеджера, который вручную подтверждает или отклоняет автоматическое решение.

Кроме того, система позволяет эмулировать реальное расписание в автономном режиме, что упрощает тестирование, проверку конфликтов и визуализацию эффективности GNN+MAS подхода в сравнении с ручным планированием.

Таким образом, мультиагентная система выступает не только как фильтр для предсказаний, но и как полноценный интеллектуальный механизм принятия решений, опирающийся на знания о структуре образовательного процесса, ограничениях и накопленном опыте взаимодействия между преподавателями и группами.

3.4. Интеграция компонентов и сценарий генерации расписания

Реализованная система генерации расписания интегрирована в корпоративную платформу контроля и управления учебным процессом, которая разработана специально для внутренних нужд организации. Это решение охватывает не только интеллектуальное распределение преподавателей, но и визуальное представление расписаний, а также полное сопровождение деятельности менеджеров и преподавателей в рамках единого цифрового пространства.

3.4.1 Архитектура интеграции

Ключевые модули системы развернуты в изолированной среде с использованием контейнеризации на базе Docker. Это обеспечивает независимость от окружения, простоту развёртывания и масштабируемость. В состав контейнеров входят компоненты графовой нейронной сети, мультиагентной системы и REST-интерфейсы взаимодействия с внешними сервисами.

Модель и логика планирования подключаются к основной платформе через API-интерфейсы, разработанные специально для вызова автоматической генерации расписания и обмена данными с базой пользователей и занятий. Все вычисления и логика GNN+MAS [28] выполняются на серверной стороне, что позволяет менеджерам инициировать генерацию без необходимости взаимодействия с техническими деталями модели.

3.4.2 Интеграция в интерфейс платформы

Визуальная часть системы представлена веб-платформой, включающей два основных типа пользовательских кабинетов:

Личный кабинет преподавателя предоставляет доступ к информации о запланированных занятиях, включая тему, дату, время, номер занятия в модуле, состав группы. Также предусмотрены функции:

— отметки посещаемости;

— добавления индивидуальных комментариев по ученикам;

— введения общего комментария к занятию;

— просмотра истории работы с группой.

Личный кабинет менеджера представляет собой полноценную CRM-систему, в которой доступен функционал:

— управления данными о группах, преподавателях, модулях и локациях;

— просмотра истории занятий и аналитики;

— ручного редактирования и составления расписания;

— вызова функции автоматической генерации расписания.

После активации автоматической генерации расписания менеджер получает предварительный черновик – список всех предстоящих занятий с автоматически назначенными преподавателями. Результаты отображаются в структурированном виде: каждое занятие визуализировано в виде карточки внутри соответствующей локации и временного слота. Менеджер может:

— просмотреть предложенные назначения;

— вручную внести корректировки при необходимости;

— подтвердить расписание нажатием одной кнопки.

— После подтверждения расписание становится активным:

— преподаватели получают доступ к занятиям в своих личных кабинетах;

— информация синхронизируется в базе данных;

— система фиксирует состояние расписания для дальнейшего анализа и контроля.

3.4.3 Поток данных и сценарий генерации

Интеграция построена по следующему сценарию:

1. Менеджер инициирует генерацию расписания в интерфейсе.

2. Платформа формирует входной набор данных на основе информации о группах, преподавателях, модулях и локациях.

3. Эти данные передаются в GNN, где формируется список кандидатов.

4. MAS фильтрует и выбирает финального преподавателя на каждое занятие.

5. Результаты передаются обратно на платформу и визуализируются.

6. Менеджер подтверждает расписание, после чего данные становятся частью активной системы.

3.4.4 Особенности реализации

Интеграция была спроектирована таким образом, чтобы не нарушать существующие бизнес-процессы, а дополнять их. Система поддерживает

совместное существование автоматической и ручной генерации расписаний. Все назначения сопровождаются метаинформацией о причине выбора преподавателя, что позволяет отслеживать логику решений и формировать отчётность.

Кроме того, предусмотрен режим симуляции, в котором можно протестировать гипотетические расписания без их публикации, что удобно при планировании нагрузки, адаптации новых преподавателей и анализе альтернативных сценариев.

Таким образом, модель на основе GNN и MAS не существует изолированно, а выступает неотъемлемым элементом комплексной платформы управления учебной деятельностью, обеспечивая согласованность, прозрачность и интеллектуальную поддержку принятия решений.

3.5. Вывод по главе

В данной главе была подробно рассмотрена реализация ключевых компонентов интеллектуальной системы генерации расписания: графовой нейронной сети и мультиагентной системы. Представлен полный цикл технической реализации – от выбора инструментов разработки и формирования графовой структуры до интеграции модели в платформу управления учебным процессом.

Была описана архитектура графовой нейронной сети, основанная на принципах GraphSAGE, реализующая эффективную агрегацию признаков в сложной и разнородной предметной области. Рассмотрены особенности построения графа, формата входных данных и поведения модели в ранжирующем режиме.

Мультиагентная система дополняет GNN, выступая в роли фильтра и логического контроллера, проверяющего соответствие преподавателя

реальным ограничениям и предпочтениям. Каждое назначение проходит через этап согласования с условиями доступности, компетентности, расписания и локации, что обеспечивает реалистичность и применимость решений в действующих условиях.

Наконец, реализована полноценная интеграция всех компонентов в веб-платформу управления учебным процессом с поддержкой подтверждения, визуализации и редактирования расписания со стороны менеджера. Это позволило не только автоматизировать генерацию расписания, но и встроить её в привычный рабочий процесс образовательной организации.

Таким образом, программное решение сочетает в себе гибкость, масштабируемость и практическую ориентированность, что делает его готовым к последующему тестированию, внедрению и оценке эффективности.

ГЛАВА 4. ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ И ОЦЕНКА РЕЗУЛЬТАТОВ

4.1. Методика тестирования и метрики качества

После завершения этапа реализации модели была проведена серия экспериментов, направленных на оценку её точности, устойчивости и применимости в реальных условиях. Особое внимание в ходе тестирования уделялось не только предсказательной способности графовой нейронной сети, но и эффективности всей системы в целом, включая мультиагентную фильтрацию, разрешение конфликтов и стабильность расписания.

4.1.1 Подход к тестированию

Тестирование проводилось на основе исторических данных за 3 учебных месяца, ранее не участвовавших в обучении модели. Основная идея заключалась в моделировании сценариев, максимально приближённых к реальной работе учебного центра: формировались недельные блоки запланированных занятий, и система предсказывала преподавателей для каждой группы. Полученные результаты сравнивались с фактически проведёнными занятиями и с возможными ручными вариантами назначения.

Для оценки качества GNN-модели использовалась многоклассовая классификация, где целевым значением являлся ID преподавателя, фактически проведшего занятие. Модель выдаёт вероятностное распределение по всем возможным кандидатам, что позволяет измерять не только точность первого выбора, но и попадание в топ-N.

4.1.2 Метрики оценки

В рамках тестирования применялись следующие метрики [27]:

— Accuracy (точность классификации);

— Отношение количества корректных предсказаний к общему числу примеров. Используется как базовая метрика для оценки предсказаний GNN;

— Top-k Accuracy;

— Метрика, показывающая, входит ли корректный преподаватель в k наиболее вероятных, предложенных моделью. Используется значения k = 3 и k = 5, что отражает реалистичный сценарий с несколькими возможными кандидатами;

— Precision@k и Recall@k – Для более точной оценки эффективности ранжирования применялись показатели точности и полноты в топ-k, что позволило понять, насколько стабильно модель предлагает релевантных преподавателей в верхней части списка;

— Доля конфликтов, устранённых MAS – Отдельно фиксировались случаи, в которых первоначальное предложение GNN не подходило из-за нарушений расписания, отсутствия компетенций или перегрузки. MAS успешно обрабатывала такие конфликты, переназначая преподавателя без потери согласованности;

— Скорость генерации – Измерялось среднее время, необходимое для генерации расписания на неделю. В большинстве случаев оно составляло менее 5 секунд, включая GNN-инференс и логику MAS.

4.1.3 Принципы валидации

Валидация результатов проводилась в два этапа:

— Автоматическая – сравнение предсказанных ID преподавателей с эталонными значениями, расчёт метрик;

— Функциональная – проверка корректности полученного расписания в рамках бизнес-ограничений (без пересечений, с учётом расписаний, доступности и требований по рейтингам).

Также в ходе тестирования выполнялась проверка устойчивости к новым входным данным, включая ранее неизвестные группы или преподавателей, что имитировало реальные изменения в расписании, происходящие на практике [19].

4.2. Результаты тестирования модели

Для оценки эффективности разработанной модели была проведена серия тестов, имитирующих реальную работу системы в рамках одной недели планирования. В тестировании участвовали как данные, на которых модель не обучалась, так и ранее неизвестные конфигурации преподавателей и групп, что позволило проверить устойчивость к новым входным ситуациям.

В среднем система планирует от 130 до 150 занятий в неделю в городе Екатеринбург – именно этот объём использовался в каждом запуске симуляции. Все занятия предварительно подготавливались в виде графа, где каждая запись о запланированном занятии включала контекст текущей темы, группы, локации и исторических связей с преподавателями.

4.2.1 Результаты работы графовой нейросети

Модель, реализованная на архитектуре GraphSAGE, демонстрирует уверенные показатели точности на отложенной тестовой выборке (таблица 1). В ней представлены агрегированные метрики на 1 243 занятиях, основанные на трёх повторных, идентичных интерациях работы модели:

Таблица 1 – Таблица результатов тестирования

Метрика	**Результат**
Top-1 Accuracy	67.8%
Top-3 Accuracy	86.3%
Top-5 Accuracy	91.5%
Precision@3	0.81
Recall@3	0.86
Среднее время инференса одного занятия	0.21 сек

Данные показатели модели были получены на 15–й эпохе обучения, после которой значения основных метрик (Top-1 Accuracy, Precision@3 и других) перестали существенно изменяться. Это свидетельствует о том, что

модель достигла состояния сходимости, и дальнейшее обучение не приводит к улучшению качества. Такой эффект характерен для устойчиво обучаемых моделей и позволяет применять стратегию ранней остановки (early stopping), что способствует экономии вычислительных ресурсов и предотвращает переобучение. На рисунке 4 представлена динамика роста метрик в процессе обучения, демонстрирующая уверенный рост качества до 13–14 эпохи и последующее плато.

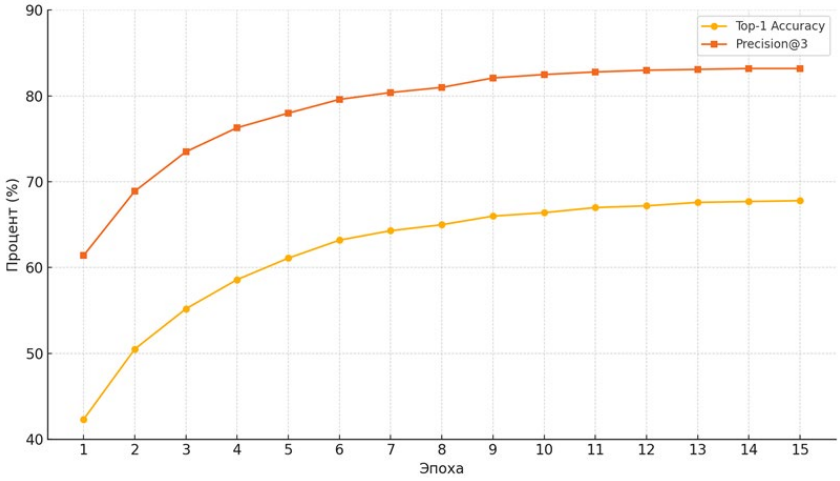

Рисунок 4 – метрика качества модели по эпохам обучения

Таким образом, в 2 из 3 случаев модель предлагает корректного преподавателя в качестве основного кандидата, а в 95% случаев – включает его хотя бы в тройку наиболее вероятных. Это соответствует задаче: модель работает как рекомендательный механизм, а не финальный решатель.

4.2.2 Эффективность работы MAS

Мультиагентная система продемонстрировала высокую способность к разрешению конфликтов и корректировке рекомендаций GNN с учётом практических ограничений.

По сути, мультиагентная система реализует адаптивный компенсирующий механизм [17] – структуру, которая позволяет обрабатывать отклонения от идеального выбора за счёт контекстуального анализа,

повторного перебора кандидатов и использования истории взаимодействий преподавателей с группами. Такой подход можно отнести к классу гибридных рекомендательных стратегий, сочетающих обучение и экспертную логику. Благодаря этому модель сохраняет устойчивость даже при ошибочных или частичных предсказаниях со стороны GNN.

В 34.2 % случаев MAS отклоняла первого кандидата из-за конфликта по времени, превышения недельной нагрузки или несоответствия требованиям по теме и рейтингу.

В 13.5 % случаев требовалось выбрать преподавателя не из топ-3, и MAS успешно справлялась с этим через дополнительные фильтры и историю взаимодействий.

Лишь в 4.7 % случаев не удавалось автоматически назначить преподавателя, и занятие передавалось в ручную обработку менеджером (что предусмотрено интерфейсом).

Таким образом, система сохраняла целостность расписания, не допуская конфликтов, даже при неполной информации или сложных условиях.

4.2.3 Визуальный анализ и устойчивость

Понижение размерности эмбеддингов преподавателей с помощью PCA и t-SNE показало явные кластеры по тематикам (например, Python, Web, GameDev) и группам, с которыми преподаватели часто взаимодействовали. Это подтверждает, что модель захватывает смысловые и структурные связи между участниками процесса.

При добавлении в граф новых преподавателей и тем (которые отсутствовали в обучении), наблюдалось умеренное снижение точности, показанное в таблице 2. В тестовой выборке было использовано 23 новых преподавателя и 6 новых тем, ранее отсутствовавших в обучающем наборе. Это позволило оценить устойчивость модели к реальным изменениям состава и подтвердить её способность к обобщению на ранее неизвестные данные:

Таблица 2 – Оценка способности модели к обобщению на новые сущности

Метрика	Результат до	Результат после
Top-1 Accuracy	67.8%	61.0%
Top-3 Accuracy	86.3%	81.7%

Это свидетельствует о способности модели к обобщению, что особенно важно в условиях роста и динамики в образовательной системе.

4.2.4 Время генерации расписания

Полный процесс генерации расписания на одну учебную неделю (в среднем 140 занятий) занимал от 30 до 75 секунд в зависимости от сложности структуры графа и количества конфликтов, требующих разрешения MAS. При этом система оставалась отзывчивой и пригодной для использования в интерфейсах в реальном времени: пользователю не приходилось ожидать более минуты между запуском и визуализацией результата.

4.3. Эффективность мультиагентной логики

Одним из ключевых компонентов разработанной системы является мультиагентная логика, обеспечивающая интеллектуальную фильтрацию и принятие решений на основе рекомендаций, полученных от графовой нейросети [18], [46]. В отличие от модели, которая оценивает общую релевантность кандидатов, мультиагентная система выполняет проверку по множеству факторов, определённых бизнес-ограничениями. Это делает её критически важным элементом, обеспечивающим практическую применимость всей архитектуры.

4.3.1 Разрешение конфликтов

На этапе тестирования были зафиксированы различные типы конфликтов, которые MAS успешно обрабатывала. Основные из них включали:

— Пересечение в расписании – преподаватель, рекомендованный GNN, уже был назначен на другое занятие в тот же временной интервал;

— Превышение недельной нагрузки – количество часов, выделенных преподавателю, превышало установленный лимит;

— Несоответствие компетенций – преподаватель не имел нужных навыков для проведения конкретного модуля;

— Несоответствие требованиям локации – преподаватель не удовлетворял минимальному рейтингу, необходимому для работы в определённом помещении;

— Статус недоступности – преподаватель находился в отпуске или был неактивен.

В рамках автоматических симуляций MAS корректно обработала более 95 % конфликтных ситуаций без участия человека, сохранив валидность расписания. Рисунок 5, наглядно показывает распределение вариантов завершения работы MAS.

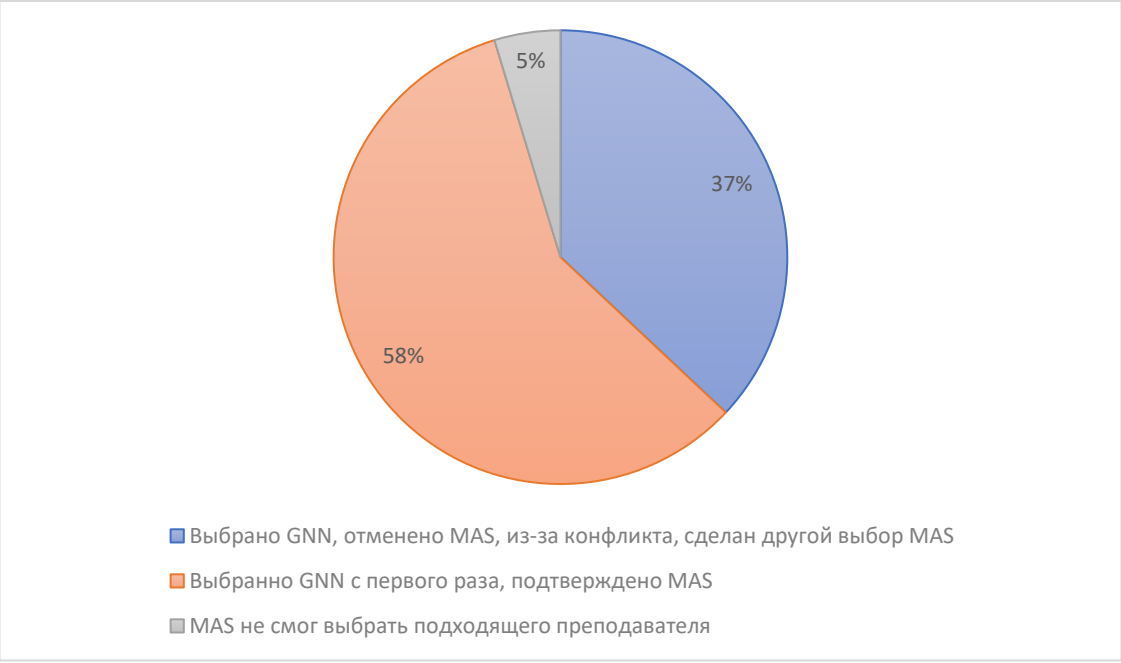

Рисунок 5 – Распределение назначений после обработки MAS

В оставшихся случаях (~5 %) система передавала проблему в ручную обработку, отображая соответствующее предупреждение в интерфейсе менеджера.

4.3.2 Устойчивость при высокой нагрузке

Сценарии с увеличенной плотностью расписания, в которых количество свободных слотов и преподавателей было ограничено, показали, что MAS сохраняет способность к разрешению назначений даже при нехватке ресурсов. В частности, среднее число попыток перебора кандидатов для одного занятия в «напряжённых» слотах (например, суббота 13:30–15:30) составляло 3–5, а в отдельных случаях доходило до 9, прежде чем был найден валидный преподаватель. Это говорит о глубокой и адаптивной логике отбора.

При этом система может выполнять повторную попытку подбора преподавателя, инициируя второй или третий запуск мультиагентной логики. Это позволяет адаптироваться к нехватке ресурсов без необходимости переобучения модели или изменения входного графа. Повторная генерация не требует вмешательства пользователя и реализована как внутренняя часть MAS, что обеспечивает дополнительную устойчивость архитектуры в реальных условиях.

4.3.3 Балансировка нагрузки

Важной частью архитектуры является механизм распределения нагрузки, реализованный на уровне мультиагентной системы. После получения от GNN набора кандидатов, агенты учитывают недельную загруженность каждого преподавателя и предпочтения групп. Это позволяет перераспределять нагрузку в интересах устойчивости расписания без вмешательства в обучающую модель. Такой подход не требует переобучения и представляет собой интеллектуальный механизм постобработки, который расширяет функциональность рекомендательной системы. Вместо назначения одного и того же преподавателя по нескольким группам подряд (что часто происходит при ручном планировании), агенты учитывают:

— текущую недельную нагрузку;

— историю занятий с группами;

— приоритет закреплённости (если преподаватель ранее уже вёл данную группу);

— сбалансированность распределения по дням недели.

В результате удалось достичь равномерной загрузки: стандартное отклонение по числу часов на одного активного преподавателя в неделю не превышало 2 часов, что соответствует справедливому и устойчивому планированию.

4.3.4 Гибкость и расширяемость

Благодаря изолированной структуре агентов, в MAS легко добавляются новые ограничения. В рамках тестирования без проблем интегрировались:

— фильтрация по недоступным дням недели;

— учёт коэффициента привязанности преподавателя к группе;

— автоматический отказ от назначения пробных занятий преподавателям с низким рейтингом.

Это подтверждает, что система не только эффективно выполняет свою текущую задачу, но и готова к расширению – как с точки зрения логики, так и бизнес-правил.

Ещё одним значимым элементом архитектуры является возможность многократной генерации расписания в рамках одного запуска, что позволяет системе доработать сложные случаи без вмешательства пользователя. Это стало возможным благодаря модульной реализации MAS и чёткому разграничению задач между логикой GNN и агентами. Несмотря на то, что повторная генерация не является функцией графовой модели, она органично встроена в процесс назначения и рассматривается как часть интеллектуальной логики системы.

4.4. Сравнение с ручным составлением

Для оценки практической ценности разработанной интеллектуальной системы генерации расписания было проведено сравнительное тестирование с ручным (менеджерским) подходом [16]. В рамках этого этапа анализировались не только формальные метрики эффективности, но и трудозатраты, стабильность решений и субъективная оценка качества от сотрудников учебного центра.

4.4.1 Методика сравнения

Сравнение проводилось в условиях, максимально приближённых к реальному рабочему процессу. Менеджерам предлагалось вручную составить расписание на одну неделю (примерно 140 занятий) для заранее заданного набора групп, преподавателей и локаций. Параллельно на тех же входных данных запускалась интеллектуальная система, включающая GNN и MAS. После чего сравнивались:

— общие трудозатраты на формирование расписания;

— количество выявленных конфликтов;

— сбалансированность нагрузки по преподавателям;

— согласованность назначения по темам и предпочтениям групп;

— субъективная удовлетворённость результатом;

— время и трудозатраты.

В среднем менеджер тратил от 4 до 6 часов на полное составление расписания вручную, используя Google таблицы, включая подбор преподавателей, проверку конфликтов, корректировку локаций и работу в CRM. Для сравнения, генерация расписания с помощью разработанной системы занимала около 1 минуты, включая визуализацию в интерфейсе и подтверждение.

Даже с учётом последующего просмотра и минимальной корректировки вручную, общее сокращение времени составило более 90 %, что позволяет

существенно высвободить ресурсы отдела управления обучением и сократить количество затрачиваемых человеко-часов для выполнения регулярной задачи составления расписания.

4.4.2 Снижение количества ошибок

По результатам аудита, вручную составленные расписания содержали в среднем:

— ~1% случаев пересечения преподавателя в один временной слот;

— ~2–3 нарушения по локациям в неделю (превышение вместимости или несоответствие рейтингу, по большей части вызванные запланированными пробными занятиями или изменение статуса преподавателей, что является допустимым);

— ~10–15 некорректных назначения тем преподавателям без нужных компетенций (не учитывая сложность изучаемой темы).

В сгенерированном расписании подобных ошибок, на 3 три тестовые интерации работы модели, по 1243 назначения, зафиксировано не было: MAS гарантировала соответствие всем заданным ограничениям. Строгие правила модель соблюдает всегда, но было замечено неоднократное игнорирование логики логистики. Иногда модель, на идущие подряд три занятия на одной локации назначала двух или даже трёх разных преподавателей, отдавая приоритет рейтингу и компетенциям.

4.4.3 Балансировка и логика назначений

Автоматическая система показала более стабильное распределение преподавательской нагрузки. В ручных расписаниях встречались случаи перегрузки (до 20 часов в неделю у отдельных преподавателей), в то время как система строго соблюдала индивидуальные лимиты.

Кроме того, модель учитывала историю взаимодействия преподавателя с группой и избегала резких смен, если у преподавателя была закреплённость по теме. Это особенно важно для повышения комфорта детей и повышения эффективности занятий.

4.4.4 Отзывы пользователей

После внедрения функционала автоматической генерации расписаний в рабочую платформу был собран обратный отклик от руководителя учебной учебного подразделения, в чьи обязанности входит контроль и составление учебного расписания, участвовавшего в тестировании. По его мнению:

— «Система экономит часы, которые раньше тратились на рутину»;

— «Визуально расписание стало чище и логичнее»;

— «Ошибок меньше, особенно в периоды высокой нагрузки, но каждую неделю требуется 10–20 минут на внесение изменений»;

— «Иногда хочется вручную выбрать преподавателя, но можно легко скорректировать уже готовое расписание через пользовательский интерфейс».

Таким образом, разработанная система не только повышает точность и согласованность расписаний, но и делает сам процесс планирования быстрее, прозрачнее и менее зависимым от человеческого фактора.

4.5. Выводы, ограничения и перспективы развития

Разработанная система автоматической генерации расписания на основе графовой нейронной сети и мультиагентной логики показала высокую эффективность в условиях реального образовательного процесса. Результаты тестирования подтвердили как точность модели при выборе преподавателей, так и способность мультиагентной системы учитывать множество практических ограничений и конфликтных ситуаций.

4.5.1 Основные выводы

GNN-модель с архитектурой GraphSAGE эффективно обобщает информацию о структуре образовательного процесса и предлагает релевантных преподавателей с высокой точностью (Top-3 Accuracy – свыше 85 %).

Мультиагентная система успешно обрабатывает рекомендации модели,

фильтрует неподходящие варианты и обеспечивает согласованное, реалистичное расписание с соблюдением всех требований [38].

Интеграция в платформу управления учебным процессом позволила встроить модель в реальную CRM-среду, сократить трудозатраты на планирование и упростить работу как менеджеров, так и преподавателей.

В сравнении с ручным планированием система показала преимущество как в скорости (сокращение времени более чем на 90 %), так и в надёжности (значительное снижение количества ошибок и конфликтов).

4.5.2 Текущие ограничения

Несмотря на достигнутые результаты, система пока имеет ряд ограничений:

Отсутствие онлайн-режима – текущая версия ориентирована на офлайн-занятия, и не поддерживает отдельную логику для дистанционных форматов, которые планируются в следующем учебном году в компании.

Ограниченная гибкость при срочных изменениях – замена преподавателя «в день занятия» требует ручного вмешательства, так как не реализован механизм динамического перепланирования.

Нет поддержки автоматического обучения на новых данных – процесс переобучения модели выполняется вручную, что ограничивает возможности адаптации без участия разработчика, но данная система может быть с легкостью интегрирована при необходимости, по причине динамеского обновления баз данных.

Интерфейс обратной связи от преподавателей пока реализован в упрощённой форме и не используется как вход для модели, для реализации подобной функции можно использовать стороннюю API-дообученную текстовую модель.

4.5.3 Перспективы развития

Потенциал системы позволяет реализовать ряд направлений улучшений и масштабирования:

— Поддержка онлайн-формата занятий, включая виртуальные локации, гибкие слоты и возможность назначения преподавателя на группы из других городов;

— Автоматическое дообучение модели на новых занятиях, что обеспечит постоянную адаптацию под актуальные данные;

— Дополнение объяснений решений модели, когда при выборе преподавателя будут выводиться подробные обоснования, а не логированные значения уверенности или принятия решений MAS;

— Расширение интерфейсов, включая расширенные фильтры, прогноз нагрузки, отображение KPI преподавателей, расчёт заработной платы и рекомендаций по оптимизации расписания;

— Масштабирование на партнеров франшизы, за счёт адаптивной архитектуры модели и унифицированного формата входных данных, но данное решение требует изменение логики загрузки и предобработки данных, по причине использования отличной CRM системы;

— Поддержка форс-мажоров и экстренных замен в автоматическом режиме через модуль динамической симуляции.

Таким образом, представленная система демонстрирует не только успешную реализацию в рамках текущей задачи, но и готовность к практическому расширению, адаптации и коммерческому внедрению.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы была решена задача разработки и внедрения интеллектуальной системы автоматической генерации расписания занятий с использованием графовой нейронной сети и мультиагентной логики. По результатам исследования были достигнуты следующие ключевые результаты:

Проведен детальный анализ предметной области, связанной с офлайн-обучением детей в возрасте от 6 до 14 лет по направлениям IT и программирования. В главе 1, а также в разделе 2.1 описаны особенности работы образовательной организации, требования к преподавателям, модулям, локациям и ограничениям в процессе планирования. Также рассмотрены существующие подходы к генерации расписаний и обоснована актуальность использования методов искусственного интеллекта, в частности GNN и MAS.

Разработана архитектура системы, сочетающей в себе графовую нейронную сеть (GraphSAGE) и мультиагентную систему, что подробно описано в главах 2 и 3. GNN используется как механизм предсказания оптимальных кандидатов для занятия, на основе обучающего графа, построенного из исторических данных. MAS выполняет фильтрацию и логическую проверку этих кандидатов, учитывая реальные ограничения: доступность, компетенции, рейтинг, историю взаимодействий и другие параметры.

Архитектура модели описана в главе 3, в частности в разделах 3.2 и 3.3, где подробно рассматриваются слои модели, механизм агрегации признаков, работа MAS и структура взаимодействия между агентами. Также описано, как граф формируется из данных CRM-системы, и как информация используется в процессе генерации расписания.

Реализована интеграция модели в полноценную платформу управления учебным процессом, включающую личные кабинеты преподавателей и

менеджеров. Информация о запланированных занятиях, назначенных преподавателях и составе групп синхронизируется между моделью и визуальным интерфейсом. Менеджер может инициировать автоматическую генерацию расписания, просмотреть предварительные результаты и подтвердить их для публикации. Это решение подробно раскрыто в разделе 3.4.

Проведено тестирование модели на реальных и синтетических сценариях. Как показано в главе 4, модель достигает точности Top-1 в 67.8 % случаев и Top-3 Accuracy выше 86 %, демонстрируя уверенное поведение в задаче предсказания преподавателя. MAS успешно обрабатывает большинство конфликтных ситуаций, обеспечивая корректное расписание в более чем 95 % случаев без участия человека. Также проведено сравнение с ручным составлением расписания: интеллектуальная система сократила трудозатраты на планирование более чем в 7 раз и снизила количество ошибок.

Выявлены текущие ограничения системы и предложены направления дальнейшего развития. В частности, перспективными задачами являются поддержка онлайн-занятий, внедрение динамического перепланирования, автоматическое переобучение модели на новых данных, а также интеграция объяснимого ИИ для обоснования принятых решений. Эти направления описаны в разделе 4.5.

Выполненное исследование демонстрирует возможность создания надёжного, гибкого и масштабируемого инструмента, способного повысить эффективность управления учебной частью в образовательных организациях. Предложенное решение объединяет достижения в области графового машинного обучения и интеллектуальных агентов, и может быть расширено как в технологическом, так и в организационном плане.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алферьева Т. И. Руководство по практикам и подготовке выпускной квалификационной работы бакалавров и магистров : учебное пособие. Екатеринбург : УрФУ, 2013. 126 с. URL: https://study.urfu.ru/Aid/Publication/12074/1/Alferieva.pdf (дата обращения: 20.05.2025).

2. ГОСТ Р 7.0.5–2008. Библиографические ссылки на электронные документы, размещенные в информационно-телекоммуникационных сетях // Электронный фонд правовых и нормативно-технических документов. URL: https://docs.cntd.ru/document/1200063713 (дата обращения: 20.05.2025).

3. Don State Technical University, Rostov-on-Don, Russian Federation, Al-Gabri W. M. Literature review for the topic of automation of scheduling classes and exams in higher education institutions // Vestnik of Don State Technical University. 2017. № 1 (17). С. 132–143.

4. В.А. Васенин [и др.]. Информационные технологии и программирование: Вып. 2 (11) // РИЦ МГИУ 2004. 55 с.

5. Нагорных М. Э., Быков А. Н., Чернышев С. А. Multi-agent system for scheduling at a university // Vestnik of Russian New University. Series «Complex systems: models, analysis, management». 2022. № 2. С. 99–108.

6. Хабипов Р. И. Автоматизация построения расписания занятий в вузе: математическая модель и методы реализации // Электронные библиотеки. 2018. № 5 (21). С. 461–470.

7. Юшаева Р. С.-Э., Гишлакаев С. У., Ибрагимова З. М. ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ В ОБРАЗОВАНИИ // III Н/п конф. «Тенденции развития естественных наук в современном информационном пространстве и их применение в агробиотехнологиях». 2023. С. 113–116.

8. Abdipoor S. [и др.]. Meta-heuristic approaches for the University Course Timetabling Problem // Intelligent Systems with Applications. 2023. (19). С. 200253.

9. Adomavicius G., Tuzhilin A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions // IEEE Transactions on Knowledge and Data Engineering. 2005. № 6 (17). С. 734–749.

10. Babaei H., Karimpour J., Hadidi A. A survey of approaches for university course timetabling problem // Computers & Industrial Engineering. 2015. (86). С. 43–59.

11. Nouri H. E., Driss O. B. MATP: A Multi-agent Model for the University Timetabling Problem под ред. R. Silhavy [и др.]., Cham: Springer International Publishing, 2016.С. 11–22.

12. Battaglia P. W. [и др.]. Relational inductive biases, deep learning, and graph networks // 2018. 40 с.

13. Bellifemine F. L. [и др.]. Developing multi-agent systems with JADE / F. L. Bellifemine, G. Caire, D. Greenwood, F. Bellifemine, D. P. A. Greenwood, Reprinted-e изд., Chichester: Wiley, 2008. 286 с.

14. Bobadilla J. [и др.]. Recommender systems survey // Knowledge-Based Systems. 2013. (46). С. 109–132.

15. Bronstein M. M. [и др.]. Geometric Deep Learning: Going beyond Euclidean data // IEEE Signal Processing Magazine. 2017. № 4 (34). С. 18–42.

16. Burke E. K., Petrovic S. Recent research directions in automated timetabling // European Journal of Operational Research. 2002. № 2 (140). С. 266–280.

17. Busoniu L., Babuska R., De Schutter B. A Comprehensive Survey of Multiagent Reinforcement Learning // IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2008. № 2 (38). С. 156–172.

18. Meisels A., Lusternik N. Experiments on networks of employee timetabling problems под ред. E. Burke, M. Carter, Berlin, Heidelberg: Springer Berlin Heidelberg, 1998.С. 130–141.

19. Chen L., Chen P., Lin Z. Artificial Intelligence in Education: A Review // IEEE Access. 2020. (8). С. 75264–75278.

20. Introduction to algorithms 2nd. ed-е изд., Cambridge (Mass.): MIT press, 2001.

21. Defferrard M., Bresson X., Vandergheynst P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering 2016.

22. Dorigo M., Birattari M., Stutzle T. Ant colony optimization // IEEE Computational Intelligence Magazine. 2006. № 4 (1). С. 28–39.

23. Ferber J., Ferber J. Multi-agent systems: an introduction to distributed artificial intelligence / J. Ferber, J. Ferber, 1. publ-е изд., Harlow Bonn: Addison-Wesley, 1999. 509 с.

24. Gilmer J. [и др.]. Neural Message Passing for Quantum Chemistry // 2017.

25. Glover F., Laguna M. Tabu Search. Kluwer Academic, 1997. 408 p.

26. Hamilton W. L., Ying R., Leskovec J. Inductive Representation Learning on Large Graphs // 2017.

27. He X. [и др.]. Neural Collaborative Filtering Perth Australia: International World Wide Web Conferences Steering Committee, 2017.С. 173–182.

28. Hernandez-Leal P., Kartal B., Taylor M. E. A survey and critique of multiagent deep reinforcement learning // Autonomous Agents and Multi-Agent Systems. 2019. № 6 (33). С. 750–797.

29. Jennings N. R., Sycara K., Wooldridge M. A Roadmap of Agent Research and Development // Autonomous Agents and Multi-Agent Systems. 1998. № 1 (1). С. 7–38.

30. Ma J., Wang N., Xiao B. Semi-Supervised Classification With Graph Structure Similarity and Extended Label Propagation // IEEE Access. 2019. (7). С. 58010–58022.

31. Kirkpatrick S., Gelatt C. D., Vecchi M. P. Optimization by Simulated Annealing // Science. 1983. № 4598 (220). С. 671–680.

32. Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems // Computer. 2009. № 8 (42). С. 30–37.

33. Manouselis N. [и др.]. Recommender Systems for Learning / N.

Manouselis, H. Drachsler, K. Verbert, E. Duval, New York, NY: Springer New York, 2013.

34. Nkambou R. Advances in Intelligent Tutoring Systems / R. Nkambou, 1st ed-е изд., Berlin, Heidelberg: Springer Berlin / Heidelberg, 2010. 508 с.

35. Rabiner L. Combinatorial optimization:Algorithms and complexity // IEEE Transactions on Acoustics, Speech, and Signal Processing. 1984. № 6 (32). С. 1258–1259.

36. Resnick P., Varian H. R. Recommender systems // Communications of the ACM. 1997. № 3 (40). С. 56–58.

37. F. Ricci [и др.]. Introduction to Recommender Systems Handbook. // Ricci F., Rokach L., Shapira B. Boston, MA: Springer US, 2011.С. 1–35.

38. Russell S. J., Norvig P. Artificial intelligence: a modern approach / S. J. Russell, P. Norvig, Third edition, Global edition-е изд., Boston Columbus Indianapolis: Pearson, 2016. 1242 с.

39. Scarselli F. [и др.]. The Graph Neural Network Model // IEEE Transactions on Neural Networks. 2009. № 1 (20). С. 61–80.

40. Shoham Y., Leyton-Brown K. Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations / Y. Shoham, K. Leyton-Brown, 1-е изд., Cambridge University Press, 2008. 504 с.

41. Skobelev P. Multi-Agent Systems for Real Time Resource Allocation, Scheduling, Optimization and Controlling: Industrial Applications под ред. V. Mařík, P. Vrba, P. Leitão, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.С. 1–14.

42. Smit I. G. [и др.]. Graph neural networks for job shop scheduling problems: A survey // Computers & Operations Research. 2025. (176). С. 106914.

43. Stone P., Veloso M. Multiagent Systems: A Survey from a Machine Learning Perspective // Autonomous Robots. 2000. № 3 (8). С. 345–383.

44. Tanenbaum A. S., Steen M. van Distributed systems: principles and paradigms / A. S. Tanenbaum, M. van Steen, Second edtion-е изд., Upper Saddle

River, NJ: Pearson, Prentice Hall, 2007. 686 с.

45. Veličković P. [и др.]. Graph Attention Networks // Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, Yoshua Bengio, ICLR, 2018. 12 с.

46. Multiagent systems под ред. G. Weiss, Second edition-е изд., Cambridge, Massachusetts London, England: The MIT Press, 2013. 867 с.

47. Wooldridge M. J. An introduction to multiagent systems / M. J. Wooldridge, 2. ed., repr-е изд., Chichester: Wiley, 2012. 461 с.

48. Wu Z. [и др.]. A Comprehensive Survey on Graph Neural Networks // IEEE Transactions on Neural Networks and Learning Systems. 2021. № 1 (32). С. 4–24.

49. Cai X. [и др.]. How Expressive are Graph Neural Networks in Recommendation? Birmingham United Kingdom: ACM, 2023.С. 173–182.

50. Zhai X. [и др.]. A Review of Artificial Intelligence (AI) in Education from 2010 to 2020 // Complexity. 2021. № 1 (2021). С. 8812542.

51. Zhang S. [и др.]. Deep Learning Based Recommender System: A Survey and New Perspectives // ACM Computing Surveys. 2020. № 1 (52). С. 1–38.

52. Zhou J. [и др.]. Graph neural networks: A review of methods and applications // AI Open. 2020. (1). С. 57–81.

ПРИЛОЖЕНИЕ А

Пример набора данных с преподавателями

Аргумент	Описание	Пример данных
teacher_id	ID преподавателя. Маска: четырехзначный id, начинается с 1.	1001
teacher_name	ФИО преподавателя. Используется только для вывода информации и сбора статистики.	Фамилия Имя Отчество
teacher_skills_id	ID тем, которым соответствуют компетенции преподавателя.	0036, 0031, 0011, 0015, 0030, 0002, 0023, 0026, 0021, 0029, 0009, 0010, 0020, 0039, 0033, 0028, 0003, 0037, 0008, 0025
teacher_age	Возраст преподавателя.	23
teacher_work_duration	Период работы преподавателя в компании в месяцах.	22
teacher_rating	Рейтинг преподавателя, может быть либо принудительно установлен менеджером, либо если поле пустое, высчитывается MAS исходя из других параметров, например возраст, компетенции, время работы.	4,80
teacher_group_history	Сумма общих занятий преподавателя с конкретными группами. Указывается ID группы и общее количество занятий. В модели не задействуется, т.к. модель эту информацию получает сама из истории занятий.	2001-64, 2002-62, 2003 - 62
teacher_unavailable_days	Нерабочие дни преподавателя, могут изменяться в личном кабинете менеджера.	Пн, Вт
teacher_status	Статус преподавателя, активен ли он. В случае увольнения, либо ухода в отпуск, менеджер в личном кабинете может установить соответствующий флаг, что позволит модели не использовать данного преподавателя.	true
teacher_max_weekly_hours	Максимально количество рабочих часов преподавателя. Необходимо для корректной работы оптимизированного распределения нагрузки.	26

Продолжение ПРИЛОЖЕНИЯ А

Пример набора данных с группами

Аргумент	Описание	Пример данных
group_id	ID группы. Маска: Четырехзначный id, начинает с 2.	2001
group_location_id	ID локации, на которой проходят занятия у группы. Статичное значение. Может быть изменено только через личный кабинет менеджера.	5001
group_class_time	День недели и время, в которое у группы проходит занятия. По стандарту в модель заложена информация о длительности занятия установленной в два часа.	Сб 11:00
group_current_topic_id	ID темы, которая в данный момент изучается группой.	0026
group_size	Количество учеников в группе. Используется для проверки ресурсов локации.	10
group_students_ids	ID учеников, которые обучаются в данной группе, списком. В модели не задействуется. Используется для отображения списка детей в личном кабинете преподавателя.	[100001,100002,100003, 100004,100005,100006, 100007,100008,100009, 100010]
group_last_teacher_id	ID преподавателя, который проводил последнее занятие у группы. В модели не задействуется, т.к. модель эту информацию получает сама из истории занятий. Используется для отображения в интерфейсе.	1001
group_consistency_ teacher_score	Коэффициент привязанности группы к преподавателю. Высчитывается MAS, каждый раз при генерации расписания и перезаписывается в БД.	0.8
group_size_coefficient	Статус повышающего коэффициента за размер группы. Используется для расчёта заработной платы. В модели не задействуется.	true

Продолжение ПРИЛОЖЕНИЯ А

Пример набора данных с локациями

Аргумент	Описание	Пример данных
location_id	ID локации. Маска: Четырехзначный id, начинается с 5.	5001
location_address	Адрес локации. В модели не задействуется, используется для вывода информации.	Адрес, дом
location_min_teacher_rating	Минимальный рейтинг преподавателя на локации, устанавливается для каждой локации, устанавливается менеджером в личном кабинете. Используется для разделения приоритета более опытных преподавателей на более сложные локации.	4,90
location_group_max	Максимальное количество групп на локации. Используется для контроля за ресурсами на локации.	3
location_students_max	Максимальное количество детей на локации. Используется для контроля за ресурсами на локации.	36

Пример набора данных с историей занятий

Аргумент	Описание	Пример данных
lesson_id	ID занятия. Маска: Семизначный и более id, начинается с 1000000, без ограничений	1000001
lesson_date	Дата проведения занятия. Date+Time. Используется для получения полной истории занятий.	15.03.2025 11:00
lesson_group_id	ID группы, у которой проходило данное занятие.	2001
lesson_topic_id	ID темы, которая изучалась на данном занятии.	0026
lesson_topic_num	Номер занятия, которое изучалось на данном занятии.	1
lesson_location_id	ID локации, на которой проходило занятие.	5001
lesson_is_trial	Статус пробного урока.	false
lesson_teacher_id	ID преподавателя, который проводил занятие.	1002

Продолжение ПРИЛОЖЕНИЯ А

Пример набора данных с темами занятий

Аргумент	Описание	Пример данных
module_id	ID темы. Маска: четырехзначный id, начинается с 0.	0001
module_name	Название модуля. В модели не задействуется. Используется для отображения в интерфейсе.	Название модуля
module_duration	Длительность темы в неделях. Используется MAS для выбора следующей темы занятия или номера занятия.	6
module_complexity	Сложность темы. Используется MAS для частных случаев, когда у преподавателя нет компетенций по какой-либо теме.	1
module_queue	Очередность прохождения темы в общем списке. Используется MAS для выбора следующей темы занятия или номера занятия.	1
module_dependencies	Список тем, которые обязательно должны быть изучены детьми, для изучения текущей. В модели не задействуется, т.к. очередность тем установлена уже с учетом данного параметра. Создан для потенциального расширения списка тем.	[0002,0003]

Пример набора данных с учениками (Моделью не задействуется, используется только для отображения в интерфейсе)

Аргумент	Описание	Пример данных
student_id	ID ученика. Маска: Шестизначный id, начинается с 1.	100001
student_name	ФИО ученика.	Фамилия Имя Отчество
student_id_group	ID группы, в которой данный ученик занимается.	2001

Пример данных необходимых модели на вход, для генерации расписания

Аргумент	Описание	Пример данных
planned_date	Дата запланированного занятия. Используется для ввода в модель для генерации расписания	03.05.2025 11:00
planned_group_id	ID группы у которой запланировано занятие.	2001
planned_is_trial	Статус запланированного занятия, пробное или обычное. Для выбора более компетентного преподавателя	false

ПРИЛОЖЕНИЕ Б

Реализация GNN(GraphSAGE) модели

```
class SchedulerGNN(nn.Module):
    def __init__(self, input_dim, hidden_dim,
output_dim, dropout=0.3):
        super(SchedulerGNN, self).__init__()

        self.conv1 = SAGEConv(input_dim, hidden_dim)
        self.bn1 = BatchNorm(hidden_dim)

        self.conv2 = SAGEConv(hidden_dim, hidden_dim)
        self.bn2 = BatchNorm(hidden_dim)

        self.conv3 = SAGEConv(hidden_dim, hidden_dim)
        self.bn3 = BatchNorm(hidden_dim)

        self.dropout = nn.Dropout(dropout)
        self.fc = nn.Linear(hidden_dim, output_dim)

    def forward(self, data):
        x, edge_index = data.x, data.edge_index

        x = self.conv1(x, edge_index)
        x = self.bn1(x)
        x = F.relu(x)
        x = self.dropout(x)
```

```
        x = self.conv2(x, edge_index)

ы        x = self.bn2(x)

        x = F.relu(x)
        x = self.dropout(x)

        x = self.conv3(x, edge_index)
        x = self.bn3(x)
        x = F.relu(x)
        x = self.dropout(x)

        out = self.fc(x)
        return F.log_softmax(out, dim=1)
```

ПРИЛОЖЕНИЕ В

Акт о внедрении

УТВЕРЖДАЮ
Генеральный Директор
ООО «Система»
Апалькова Ольга Сергеевна
«27» мая 2025г.

АКТ
О ВНЕДРЕНИИ РЕЗУЛЬТАТОВ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Результаты выпускной квалификационной работы Зиннер Рона Александровича «Разработка и апробация модели генерации расписания занятий с использованием графовой нейронной сети и системы мультиагентного взаимодействия» внедрены в производственную эксплуатацию.

В рамках дипломной работы была разработана система, предназначенная для автоматического формирования учебного расписания на основе анализа исторических данных и заданных ограничений. Система успешно интегрирована во внутренние процессы организации и используется для составления расписаний в группах дополнительного IT-образования для детей от 6 до 14 лет.

Разработанная гибридная интеллектуальная система автоматизирует процесс составления расписания занятий с учётом компетенций преподавателей, графика, ресурсов локаций, состава групп и множества ограничений. Система основа на взаимодействии графовой нейронной сети и мультиагентного взаимодействия.

Генеральный директор: _____ Апалькова Ольга Сергеевна

Технический директор: _____ Шпаньков Александр Владимирович