межвузовский сборник научных трудов

МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

РГРТУ, 2022

УДК 004.4

Редакционная коллегия

Рецензент: к.т.н., доцент, зав. каф. «Информатики, информационных технологий и защиты информации» Липецкого государственного педагогического института Скуднев Д.М.

Математическое и программное обеспечение вычислительных систем: Межвуз. сб. науч. тр. / Под ред. А.Н. Пылькина – Рязань: Издательство ИП Коняхин А.В. (Book Jet), январь 2022. – 64 с.

ISBN 978-5-907400-95-5

Представлены статьи, посвященные различным аспектам разработки программных средств ЭВМ, вычислительных систем и сетей, вопросам математического моделирования, методам обработки информации.

Предназначен для научно-педагогических работников вузов, инженеров, аспирантов и студентов старших курсов.

Авторская позиция и стилистические особенности публикаций полностью сохранены.

ISBN 978-5-907400-95-5

ВВЕДЕНИЕ

Межвузовский сборник научных трудов содержит результаты научных исследований и разработок по следующим направлениям:

– программное обеспечение вычислительных систем, новые информационные технологии;

– прикладная математика, теория информации, искусственный интеллект;

– ЭВМ в системах обработки, управления и обучения;

– автоматизированное проектирование аппаратных средств вычислительных систем;

– информационные технологии в экономических и социальных системах.

Сборник сформирован на основе статей, в которых рассматриваются различные аспекты разработки программных средств ЭВМ, вычислительных систем и сетей, включая вопросы автоматизации проектирования, теории обработки информации и математического моделирования, и предназначен для студентов, аспирантов и преподавателей технических вузов и научных работников.

Материалы для сборника предоставлены сотрудниками и студентами:

– Областное государственное бюджетное профессиональное образовательное учреждение «Рязанский технологический колледж», Рязань

– Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина», Рязань

– Московский государственный технический университет им. Н. Э. Баумана.

.

УДК 004.891.2

Проказникова Е.Н., Ямпольская А. И.

СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПОДБОРА СИНОНИМОВ К СЛОВАМ ЕСТЕСТВЕННОГО ЯЗЫКА

Рязань: Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

В статье описываются существующие механизмы автоматизированного подбора синонимов к словам естественного языка, выделяются преимущества и недостатки, различия между методами, а также особенности применения каждого из них.

В современном мире в связи с растущим изобилием поисковых, диалоговых систем, интеллектуальных устройств, систем управления, основанных на применении обработки естественного языка (Natural Language Processing, NLP), всё чаще возникает потребность в использовании систем автоматизированного подбора синонимов к словам в исходном тексте. Это позволит улучшить взаимодействие между пользователем и каким-либо сервисом и поспособствует увеличению степени универсальности таких систем (снижению требований к чёткости отдаваемых пользователем команд).

Методы решения проблемы автоматизированного подбора синонимов к словам естественного языка в настоящее время активно развиваются. В связи с этим перед разработчиками таких систем всегда стоит задача оптимизации её работы, подбор наиболее сбалансированного соотношения затрачиваемых мощностей и скорости выполнения поставленной системе задачи с точностью её решения. С развитием моделей для подбора синонимов удалось приблизиться к оптимальным параметрам выполнения, а также предоставить выбор в используемых архитектурах и алгоритмах решения данной проблемы.

Распознавание именованных сущностей

Главный этап работы системы подбора синонимов заключается в извлечении (поиске) именованных сущностей (Named Entity Recognition, NER – объекты или понятия, распознаваемые в тексте) рабочей модели из предложенного текста. Распознавание именованных сущностей является одной из самых важных задач обработки естественного языка, в результате решения которой выделяются непрерывные фрагменты текста – спаны.

Разработаны различные подходы к решению задачи распознавания именованных сущностей. Глобально их можно разделить на четыре основные категории: ручные методы, метод онтологий, статистические методы и методы, основанные на применении нейронных сетей.

Ручные методы

Ручные методы извлечения именованных сущностей из текста основаны на применении рукописных правил. Стоит отметить, что ручной подход предоставляет достаточно удобную и качественную реализацию добавления *малого* количества принципиально *отличающихся* друг от друга сущностей.

Такой подход не отличается применимостью на практике из-за низкой работоспособности, однако в учебных и демонстративных целях со своими задачами справляется хорошо, обладая даже рядом незначительных *достоинств*:

1. Простота поддержки и тестирования, а также лёгкость устранения и ошибок и неточностей распознавания.
2. Простота расширения списка синонимов.
3. Отсутствие ярко выраженного проявления проблемы многозначности слов для специализированных моделей.
4. Отсутствие необходимости обучения модели и наличия размеченных корпусов текстов, сложных для поиска или создания под конкретную предметную область.
5. Полностью предсказуемое поведение системы, отсутствие вероятностного характера поиска.

Редкое использование в реальной жизни ручных методов объясняется существенностью их *недостатков*:

1. Трудоёмкий процесс составления правил.
2. Невозможность составления правил для крупных конфигураций и сущностей, значения которых не имеют ярко выделенных различий между несколькими типами.

Метод онтологий

Онтология – попытка формализации некой области знаний, то есть явное описание терминов (объектов и понятий) данной предметной области и отношений между ними (логических выражений, описывающих связь).

Достоинство данного метода:

1. Высокая точность распознавания.

Недостатки данного метода:

1. Отсутствие обобщающей способности.
2. Неприменимость к новым добавляемым объектам.

Статистические методы

При решении задачи распознавания именованных сущностей успешно используются алгоритмы машинного обучения, особенно популярны статистические модели, такие как скрытая Марковская модель и др.

Скрытая Марковская модель (СММ) представляет собой вероятностную модель последовательности. Задачей для СММ является прогнозирование неизвестных параметров на основе наблюдаемых.

Структура СММ. В скрытой Марковской модели наблюдаемыми являются только переменные, на которые оказывается влияние данным состоянием. Каждое состояние имеет вероятностное распределение среди всех возможных выходных значений. Поэтому о последовательности состояний можно судить по последовательности символов, сгенерированной скрытой Марковской моделью.

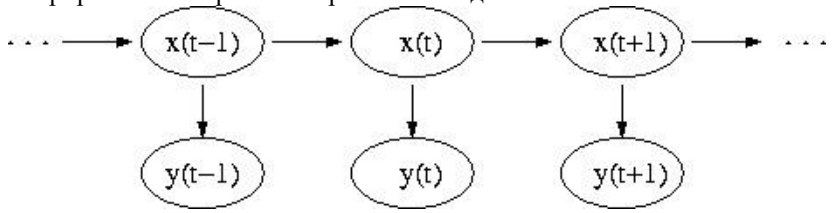

Рис 1. Диаграмма структуры СММ

Анализируя диаграмму, можно заметить, что значение *скрытой* переменной *x(t)* (в момент времени *t*) зависит только от значения *скрытой* переменной *x(t-1) (*в момент времени *t-1)*. Это явление принято называть свойством Маркова. Также стоит отметить, что значение *наблюдаемой* переменной *y(t)* зависит только от значения скрытой переменной *x(t)* (обе в момент времени *t*).

Скрытую Марковскую модель можно описать совокупностью значений n (количество состояний модели), m (количество возможных символов в наблюдаемой последовательности), A (матрица вероятностей переходов), B (распределение вероятностей появления символов в *j*-том состоянии), π (распределение вероятностей начального состояния). Такая модель может сгенерировать наблюдаемую последовательность $O = O_1O_2\ldots O_T$, где O_t – один из символов алфавита V, T – количество элементов в наблюдаемой последовательности.

При решении задач, поставленных перед скрытой Марковской моделью, проектируется отдельная модель для каждого набора примеров. В ходе работы СММ сначала модель обучается на предложенных примерах, настраивая наиболее корректные параметры A, B и π, согласно заданной точности. Далее подбирается последовательность состояний системы, которая лучше всего соответствует наблюдаемой последовательности. Затем выделяются свойства обучающих примеров, имеющие наибольший вес для определённого состояния. На этом этапе производится тонкая настройка модели. После проектирования, оптимизации и обучения набора моделей оценивается их работоспособность. Для этого модели предоставляется тестовый пример,

затем вычисляется функция соответствия тестового примера каждой модели. Тестовый пример относится к той модели, для которой функция будет иметь наибольшее значение. На этом этапе задача считается выполненной.

Недостаток статистических методов:

1. Нестабильность результатов распознавания именованных сущностей из-за часто наблюдаемого эффекта переобучения.

Методы, основанные на применении нейронных сетей

Впервые задача распознавания именованных сущностей была успешно решена посредством нейронной сети в 2011 году системой *«Natural Language Processing from Scratch»*. Был реализован метод векторных представлений слов. На начальных этапах развития превосходство нейросетевых методов над методами с применением классических алгоритмов машинного обучения не было столь значительно, результаты, выдаваемые обоими способами, были сопоставимы по качеству.

Далее модели активно развивались и экспериментально и теоретически было доказано, что качественные результаты решения задачи извлечения именованных сущностей наблюдаются при использовании рекуррентных нейронных сетей. Архитектура таких нейронных сетей обычно включает в себя несколько двунаправленных рекуррентных слоёв. Затем признаки с последнего слоя сети подаются в другой классический алгоритм.

Например, хорошие результаты показывает модель, содержащая слой условных случайных полей (CRF Layer), в качестве признаков к которому подаются веса с выхода последнего слоя двунаправленной нейронной сети.

Достоинства нейросетевых методов:

1. Скорость настройки всей системы.
2. Простота операций, требуемых от разработчика конкретной модели.

Недостатки такого способа распознавания именованных сущностей:

1. Отсутствие контроля над настройками нейронных сетей.
2. Вероятностный характер поиска сущностей.

Расширение списка синонимов

С помощью любого из вышеперечисленных способов можно настроить поиск именованных сущностей в пользовательском тексте запросов. Этот процесс является отлаживаемым, контролируемым и имеет возможности для последующего улучшения.

Далее возникает задача составления обширного списка синонимов для обеспечения наиболее качественного процесса распознавания. Для

этого применяют *аугментацию* – построение дополнительных данных из исходных при решении задач машинного обучения.

Помимо классических методов дополнения конфигурации, используют также и комбинированные способы. Один из таких методов представляет собой использование компонента *sugsyn*. Работа этого компонента тесно связана с дополнительным сервером ContextWordServer. Функционирование сервера основано на комбинировании двух методов:

1. Метод векторных представлений (Word Embedding), использование которого состоит в расчёте векторных представлений слов и отборе синонимов, расположенных в векторном пространстве ближе всего к целевому слову. Недостатком этого метода было недостаточно точное определение контекста.

2. *Метод масок(Masked Language Modelling)*, представляющий собой подбор наиболее подходящих слов для восстановления в предложении пропущенного слова. Недостатком этого метода было то, что в списке полученных слов многие из них являлись корректными по контексту использования, но были далеки семантически от заданного, то есть не могли считаться близкими синонимами целевого слова.

В результате объединения этих подходов были достигнуты наилучшие результаты подбора синонимов, так как при возврате списка слов, которые можно было бы подставить вместо маски, отфильтровывались слова, далёкие по значению, не являющиеся синонимами, то есть векторное расстояние которых до целевого слова больше определённого порогового значения.

Оценка результатов

Факторы, влияющие на качество синонимов, подбираемых в процессе расширения списка для поиска элементов заданной модели в предоставляемом тексте:

1. Широта списка синонимов, определяемого пользователем на этапе составления конфигурации элементов.

2. Количество примеров запросов к интентам, а также их естественность формулировки, корректность составления и распространённость использования.

3. Тип элемента и тип модели.

Последний фактор свидетельствует о непредсказуемости качества списка синонимов, предложенного нейронной сетью, так как это зависит от самой природы модели и её элемента. Вследствие чего для разных моделей и разных элементов необходимо внимательно рассчитывать значение минимального коэффициента достоверности предлагаемых вариантов, чтобы достичь сопоставимого уровня качества поиска синонимов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. С. Камов Поиск по синонимам в тексте — контролируем процесс или доверяемся нейросетям. URL: https://habr.com/ru/post/539528/ (дата обращения: 25.11.2021)

2. И. Смуров NLP. Основы. Техники. Саморазвитие. Часть 1. URL: https://habr.com/ru/company/abbyy/blog/437008/ (дата обращения: 25.11.2021)

3. И. Смуров NLP. Основы. Техники. Саморазвитие. Часть 2: NER URL: https://habr.com/ru/company/abbyy/blog/449514/ (дата обращения: 25.11.2021)

4. Collobert R. et al. Natural language processing (almost) from scratch //Journal of Machine Learning Research. – 2011. – Т. 12. – №. Aug. – С. 2493-2537

5. Trofimov I. V. Person name recognition in news articles based on the persons-1000/1111-F collections //16th All-Russian Scientific Conference Digital Libraries: Advanced Methods and Technologies, Digital Collection, RCDL. – 2014. – С. 217- 221.

6. С. Николенко Скрытые Марковские модели / Машинное обучение – ИТМО, 2006. URL: https://logic.pdmi.ras.ru/~sergey/teaching/mlbayes/06-hmm.pdf

7. С. Николенко Скрытые Марковские модели II / Машинное обучение – ИТМО, 2006. URL: https://logic.pdmi.ras.ru/~sergey/teaching/mlbayes/07-hmm2.pdf

8. Цепь Маркова. URL: https://ru.wikipedia.org/wiki/Цепь_Маркова

9. Лингвистика для математиков. URL: https://en.ppt-online.org/713273

10. Gernot A. Fink Что такое скрытые модели Маркова [пер. с англ.]. URL: https://habr.com/ru/post/135281/

11. Шахиди Акобир Онтология анализа данных. URL: https://basegroup.ru/community/articles/ontology

12. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика : учеб. пособие / Большакова Е.И., Клышинский Э.С., Ландэ Д.В., Носков А.А., Пескова О.В., Ягунова Е.В. — М.: МИЭМ, 2011. — 272 с.

УДК 004.65

Ю.Б. Щенёва, О.А. Бодров, С.А. Бубнов, К.А. Майков

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ УПРАВЛЕНИЯ ОРГАНИЗАЦИОННЫМ ПРОЦЕССОМ ПОДГОТОВКИ IT–СПЕЦИАЛИСТОВ В ВУЗЕ

Областное государственное бюджетное профессиональное образовательное учреждение «Рязанский технологический колледж», Рязань

Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина», Рязань

Московский государственный технический университет им. Н. Э. Баумана

В работе рассматриваются особенности повышения эффективности управления организационным процессом подготовки IT–специалистов в вузе. Предложена укрупненная структурная схема объекта управления, разработан процесс построения модели, приведен метод визуализации освоения образовательной программы.

Подготовка кадров в ВУЗе осуществляется с помощью организационной системы, состав и структура которой определяются нормативными документами, такими как Федеральный закон "Об образовании в Российской Федерации" от 29.12.2012 № 273-ФЗ, Федеральный государственный образовательный стандарт высшего образования (ФГОС ВО), основная профессиональная образовательная программа высшего образования (ОПОП ВО).

Организационный процесс подготовки IT–специалистов в ВУЗе – это процесс формирования, прежде всего, личности специалиста, от осознанного выбора абитуриентом соответствующей профессии до защиты студентом выпускной квалификационной работы.

Данный процесс неразрывно связан с научной деятельностью. Преподавание учебных дисциплин осуществляется на уровне, максимально приближенном к последним достижениям науки и практики. Развитие науки выступает решающим фактором в изменении содержания, методики и организации обучения в высших учебных заведениях [1].

Качество организационного процесса зависит от научной квалификации и педагогического мастерства преподавателей, содержания и методов обучения, условий, предоставленных студентам, материально-технической базы ВУЗа и др. Качество освоения знаний зависит от многих факторов, в том числе от базовых знаний, мотивов к обучению и личностных качеств студента. Процесс подготовки IT–специалистов можно рассматривать как организационную систему, а задачу повышения эффективности управления организационного процесса подготовки IT–специалистов в ВУЗе как задачу управления данной системой.

Управление – это воздействие субъекта управления на управляемую систему (объект управления) с целью обеспечения

требуемого её поведения или изменения её характеристик. Под организационной системой понимают объединение людей, совместно реализующих некоторую программу или цель и действующих на основе определенных процедур и правил. Наличие процедур и правил, регламентирующих совместную деятельность членов организации, является определяющим свойством и отличает организацию от группы и коллектива [3]. Управление организационной системой – воздействие на управляемую систему с целью обеспечения требуемого ее поведения. Механизм управления организационной системой – совокупность процедур принятия управленческих решений.

Укрупненная структурная схема системы управления организационным процессом подготовки IT–специалистов в ВУЗе представлена на рисунке 1.

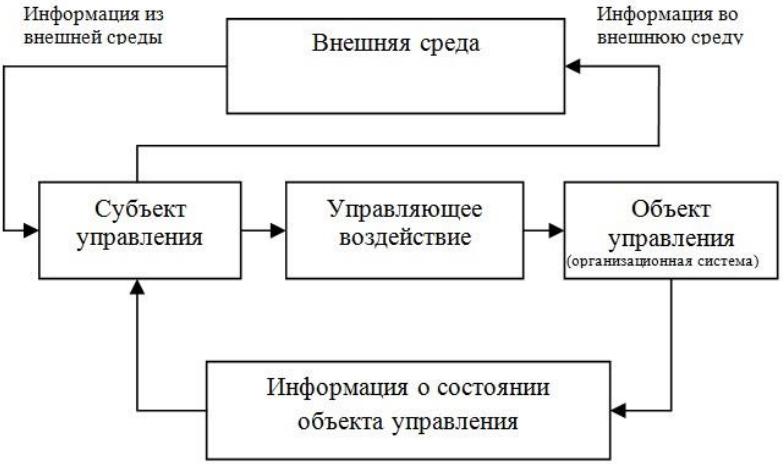

Рис. 1 – Укрупненная структурная схема системы управления организационным процессом подготовки IT–специалистов в ВУЗе

Объектом управления является организационная система для процесса подготовки IT–специалистов в ВУЗе, предметом управления – механизмы управления. При отклонении объекта управления от заданных параметров, субъект управления принимает управленческое решение, которое воздействует на объект управления (организационную систему). На схеме изображена организационная система, в которой контуры управления представляют собой информационные потоки, связывающие между собой объект управления, субъект управления и внешнюю среду.

Организационный процесс подготовки IT–специалистов в ВУЗе может быть представлен моделью, которая определяется траекторией в n–

мерном пространстве. Размерность пространства определяется количеством показателей, используемых в образовательном процессе. К таким показателям можно отнести:

–текущую аттестацию;
–промежуточную аттестацию;
–получение стипендии;
–участие в конференциях;
–научные публикации;
–другие индивидуальные достижения.

Чтобы построить модель необходимо выбрать «идеального» студента, т.е. студента с максимальными показателями. Максимальные показатели будут задавать идеальную траекторию освоения образовательной программы.

Реальные результаты освоения образовательной программы также образуют траекторию в n–мерном пространстве. Выбирая меру расхождения идеальной траектории от реальной, можно формировать управляющее воздействие в организационной системе и оценивать его эффективность. При реализации полученной модели необходимо использовать информационную систему, так как с помощью автоматизации процессов управления можно добиться получения быстрых и точных результатов.

При создании информационной системы необходимо придерживаться следующих этапов: планирование проектирования базы данных, проектирование базы данных, разработка компонент приложения информационной системы, тестирование приложения информационной системы.

В качестве визуализации освоения образовательной программы можно использовать метод представления данных «Лица Чернова». «Лица Чернова» (Chernoff Faces) – это схема визуального представления мультивариативных данных в виде человеческого лица. Каждая часть лица: нос, глаза, рот – представляет собой значение определенной переменной, назначенной для этой части. Поэтому анализ данных получается «натуралистичным». Легко делать сравнения и легко выявлять отклонения.

Каждое лицо – это массив из нескольких элементов, каждый из которых принимает значение от 0 до k макс , где k макс – максимальное значение параметра. Значению соответствует внешний вид соответствующей части лица. Параметры исследуемых объектов приводятся к этим значениям. Экстремумы реальных данных будут приняты как 0 и k макс. Все остальное – лежащим в этом промежутке. По полученному массиву конструируется лицо.

Подводя итог вышесказанному, можно заключить следующее: для повышения эффективности управления организационным процессом

подготовки IT–специалистов в ВУЗе необходимо учитывать множество факторов, от которых зависит рациональное построение организационной системы. Одним из основных определяющих факторов является спроектированная модель, которая определяется траекторией в n–мерном пространстве и визуализируется с помощью метода представления данных «Лица Чернова». С ее помощью можно формировать управляющее воздействие в организационной системе и оценивать развитие управления.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бурков В.Н., Новиков Д.А. Как управлять проектами: Научно-практическое издание. – М.: СИНТЕГ – ГЕО, 2019. – 188 с.

2. Бурков В.Н., Коргин Н.А., Новиков Д.А. Введение в теорию управления организационными системами.–М.: Либроком, 2009. – 264 с.

3. Новиков Д.А. Теория управления организационными системами. М.: МПСИ, 2005. – 584 с.

4. Сорокина Е. И. Организационные формы обучения в вузе/ Е. И. Сорокина, Л. Н. Маковкина. – Текст: непосредственный // Инновационные педагогические технологии: материалы III Междунар. науч. конф. (г. Казань, октябрь 2015 г.). – Казань: Бук, 2015. – С. 171–174. – [электронный ресурс] URL: https://moluch.ru/conf/ped/archive/183/8728/ (дата обращения: 22.12.2021)

УДК 004.891.3

Проказникова Е.Н., Чадакин К. А.

СИСТЕМА АНАЛИЗА РАСПРЕДЕЛЕНИЯ СИСТЕМНЫХ РЕСУРСОВ С ИСПОЛЬЗОВАНИЕМ АДАПТИВНОЙ СИСТЕМЫ РАСПОЗНАВАНИЯ

Рязань: Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

В статье описываются способ анализа системных ресурсов с использованием системы распознавания на основе адаптивного управления, преимущества этого метода анализа перед другими аналогами, а также математическое описание аппарата его реализации

Вычислительные машины с самого их появления служат человеку мощным и надежным инструментом для решения самых разнообразных задач. Сначала компьютеры использовались в ограниченном кругу областей, таких как научные вычисления или расшифровка закодированных сообщений. Теперь же можно сказать, что жизнь современного человека во многом зависит от компьютеров. Они не только находят решения сложнейших уравнений, но и используются для создания различных произведений искусства (фильмов, музыки, компьютерных игр и так далее), медицинского осмотра, проведения

медицинских операций, анализа погоды и предсказание ее изменение, а также для множества других задач. Сложно найти задачу, выполняя которую человек не обращался бы за помощью к компьютеру.

Сейчас все большую популярность набирают облачные вычисления. Облачные вычисления — технология распределённой обработки данных, в которой компьютерные ресурсы и мощности предоставляются пользователю как Интернет-сервис. Таким образом, основная часть вычислений производится не на локальном компьютере пользователя, а на специально выделенном сервере. Это позволяет проводить большое количество ресурсоемких операций, не занимаясь вопросами аппаратной части. Вся работа по настройке и поддержки вычислительных машин ложится на плечи провайдера сервиса.

Одной из проблем, с которой может столкнуться провайдер, является эффективное распределение вычислительных ресурсов. Рассмотрим пример системы предоставления среды для выполнения программного кода. Такая система должна иметь некоторое количество высокопроизводительных машин, которые смогут параллельно исполнять передаваемый им код. Для пользователя применение такой системы схоже с обычной средой разработки, установленной на рабочем компьютере - нужно подключиться к сервису, ввести код в отведенное для него место, запустить его выполнение и дождаться результатов.

Рассматриваемую систему можно представить как кластер, состоящий из распределительного (которых может быть несколько), назовем его планировщиком, и исполняющих узлов. Распределительный узел, получив входные данные (программный код пользователя) должен решить какому из исполнительных узлов отдать задачу на выполнение. Также одну и ту же задачу можно выполнить не нескольких исполняемых узлах одновременно, при условии, что она может быть разделена на несколько потоков.

Основные параметры, за которыми должен следить планировщик — это оперативная память исполняющего узла и время занятости процессора. Для этого он должен придерживаться какой-либо стратегии распределения, например, на основе принципа очереди (FIFO). Минусами таких стратегий является невозможность их адаптации под окружающие условия. Может быть предложена стратегию распределения ресурсов на основе системы адаптивного управления [1].

Система распознавания ресурсов на основе системы адаптивного управления состоит из набора нейронов, которые по своей сути являются отдельными распознающими системами. Каждый нейрон соединен с источником дискретного сигнала. Этим источником может являться как какой-либо датчик, так и другой подобный нейрон. Задача каждого нейрона на основе переданных ему сигналов распознать образ и передать дальше в систему сигнал, который означает что образ, за который данный

нейрон является ответственным, появляется в данный момент времени. Более того, кроме распознавания образов в конкретный момент времени (так называемый, пространственный образ), система из таких нейронов, соединенный в сеть, может распознавать пространственно-временные образы. Иными словами, сеть из нейронов может распознать образ причинно-следственной связи: если был распознан образ O_1, а за ним распознан образ O_2, то можно ожидать что будет распознан образ O_3. Таким образом, нейронная сеть имеет возможность предсказывать наступление образов.

Также до распознавания образа в задачи нейрона входит выявление того самого образа. Выявление является процессом обучения нейрона. Его суть состоит в следующем. В момент «рождения» нейрона, он является не обученным. В процессе жизни на него поступают сигналы по всем подключенным каналам в разные моменты времени. Среди этих сигналов нейрон должен выявить закономерно-появляющиеся и запомнить их. После того как нейрон их запомнил, он будет слать сигналы каждый раз, когда такой сигнал появляется на его входах.

Можно выделить 4 типа моделей нейронов, работающих по подобной схеме[1]. Рассмотрим самый простой из них тип I.

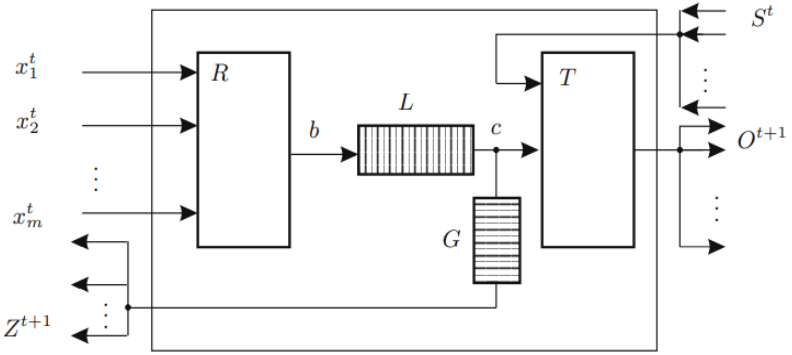

Рис.1. Модель самообучаемого нейрона типа I.

На Рис. 1 представлена схема модели самообучаемого нейрона типа I. На вход нейрона n_w в момент t поступает двоичный вектор

$$X_w^t = (x_1, x_2, ..., x_i, ..., x_m)$$

и сигнал S_w^t, где w – индекс нейрона, а n_w – нейрон.

В момент времени $t+1$ нейрон производит выходные сигналы:

$$O_w^{t+1} = \neg S_w^t \,\&\, ((b_w^t \,\&\, l_w^t) \vee O_w^t)$$ и

$$Z_w^{t+1} = b_w^t \,\&\, l_w^t \,\&\, g_w^t$$

Значение сигнала b_w^t определяется зависимостью

$$b_w^t = \begin{cases} 1, \text{если } \dfrac{h_w}{m} \geq p(N^t), \\ 0, \text{в других случаях.} \end{cases}$$

Где h_w – число таких компонент x_i вектора X^t, имеющих значение 1 в момент времени t; N^t есть число событий $b_w^t = 1$ в предыстории этого нейрона от t =0 до t; $p(N)$ – сигмоидальная функция, определенная для значений $N = 0,1,...,$ и уменьшается от $p(0) = p_{\max}, p_{\max} \leq 1$, до $p(\infty) = p_{\min}, p_{\min} < p_{\max}$.

Кроме того,

$p(M) = p_M$, $p_{\min} < p_M < p_{\max}$, где M – константа.

L_w^t показывает значение элемента L в момент времени t и принимает значения

$$l_w^t = \begin{cases} 0, \text{если } \mathrm{N}^t < M, \\ 1, \text{иначе.} \end{cases}$$

Элемент T подобен триггеру, который сигналом $(b_w^t \,\&\, l_w^t) = 1$ переключается (см. точку «с» на рис. 1) в состояние, при котором выходной сигнал Q_w^{t+1} становится равным 1, а сигналом $S_w^t = 1$ — в состояние, при котором выходной сигнал Q_w^{t+1} принимает значение 0. Переменная g_w^t определяется условием

$$g_w^t = \begin{cases} 0, \text{если } \mathrm{P}^t < Q, \\ 1, \text{иначе,} \end{cases}$$

Где P^t есть число единичных сигналов, наблюдавшихся в точке «с» в течение предыстории. Константы Q определены для каждого нейрона.

Векторы X^t, для которых $b_w^t = 1$, «обучают» нейрон (число N увеличивается). Число M для нейрона подобрано так, чтобы N не превысило M за время жизни нейрона, если такие векторы X^t суть случайные явления. С другой стороны, число N достигнет величины M в случае, если этот вектор есть неслучайное явление в системе (с заданной вероятностью ложной тревоги). Если событие $N^t = M$ случится с нейроном n_w, то мы говорим, что нейрон n_w обучен и образ O_w сформирован, начиная с этого момента t. Необратимый процесс роста N от 0 до M в нейроне n_w есть процесс обучения нейрона n_w и, тем самым, процесс формирования образа O_w. Если образ сформирован, то он не может уже быть «расформирован» (переучивание УС происходит за счет доучивания. Память о прежних образах и знаниях сохраняется в обученных нейронах). Сформированный образ может быть распознан в текущий момент $(Q_w^{t+1} = 1)$ или может быть не распознан ($Q_w^{t+1} = 0$). Несформированный образ не может быть распознан [1].

Описанный выше метод позволяет выявлять образы, позволяющие спрогнозировать затраты вычислительных ресурсов какой-либо системы на некоторое время. Кроме описанной выше задачи планирования распределения нагрузки для среды выполнения кода, такая система может анализировать любые наборы кластеров с целью распределить ресурсы так, чтобы не было простоя оборудования и нехватки ресурсов на каком-либо сервере.

Способ конструирования распознающей системе на основе адаптивного управления схож с методом конструирования искусственных нейронных сетей (ИНС), но имеет некоторые преимущества перед ними. Основным из них является самообучение. Для построения модели ИНС нужно тщательно продумывать структуру нейронной сети и обучать ее на большом количестве примеров. Кроме того, при добавлении дополнительных образов к уже обученной сети есть риск «забывания» уже сформированных образов. Для успешного добавления нового образа, ИНС нужно заново обучить на новых наборах примеров, включающий новый образ. Система распознавания на основе адаптивного управления же выявляет образы в процессе ее работы. Это значит, что при изменении условий её эксплуатации такая сеть будет подстраиваться под условия окружения, имея накопленный опыт. Эта гибкость и является основной предпосылкой выбора именно такого типа нейронной сети.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. А. А. Жданов. Автономный искусственный интеллект. Москва, 2020.

2. Дж. Рис. Облачные вычисления. Санкт-Петербург, 2011. – стр. 98

3. Л. Клюев. Распределение ресурсов в больших кластерах высокой производительности. URL:https://habr.com/ru/company/yandex/blog/ 306786/

УДК 004.05

Самохина Л.А., Крошилина А.А.

ОЦЕНКА КРЕДИТНЫХ РИСКОВ И КЛАССИФИКАЦИЯ КЛИЕНТОВ

Рязанский государственный радиотехнический университет имени В.Ф. Уткина г. Рязань

Кредит является неотъемлемой частью современного общества, он позволяет значительно снизить время для удовлетворения хозяйственных и личных потребностей субъекта. В статье рассматриваются понятие кредитного скоринга, виды оценки кредитных рисков, классификация клиентов, а также оценка интенсивности ошибок при применении скоринговых систем.

Под кредитном скорингом, как правило, понимают анализ рисков по кредитованию физических лиц, хотя методы оценки надежности организаций также существуют.

Оценка кредитных рисков в соответствии с преследуемыми целями может быть разделена на 4 категории:

1) application scoring – оценка кредитоспособности заемщиков для выдачи кредитов;

2) behavioral scoring – оценка динамики состояния кредитного счета заемщика и кредитного портфеля в целом;

3) collection scoring – определение приоритетных дел и направлений работы с проблемными заемщиками, мониторинг

задолженности и выбор оптимального коллекторского воздействия;

4) fraud scoring – своевременно выявление мошенничества со стороны клиентов-заемщиков.

Кредитный скоринг [3] состоит в применении алгоритмов, полученных с использованием математических и статистических методов, с тем чтобы разделить потенциальные кредитные операции на непересекающиеся группы риска, хорошие и плохие. Плохие риски подразумевают большую вероятность нарушения обязательств заемщиком, поэтому необходимо выявлять факторы кредитного риска, их значимость и взаимозависимость. Предполагается, что созданные модели могут выявлять закономерности, поэтому кредитные операции в будущем будут иметь такой же исход, как и операции со схожими

характеристиками, для которых известна принадлежность к одному из рисков.

Факторы, учитываемые при кредитном скоринге, могут отличаться в зависимости от алгоритмов и целей скоринга. К таким факторам можно отнести демографические данные (семейное положение, возраст) и характеристики занятости заемщика, тип занятости, должность, информацию о кредитной истории и предыдущих отношениях с кредитором, характеристики предоставляемой услуги, данные о финансовом благополучии клиента.

Для оценки рисков не менее важны характеристики запрашиваемого кредита (например, кредиты в иностранной валюте, как правило, считаются более рискованными).

Процесс построения скоринговой системы можно условно разбить на три этапа:

1. Формулировка задачи и подготовка данных. С помощью экспертов в конкретной области формулируется задача скоринга, производится сбор и предварительная обработка данных.

2. Анализ данных и построение модели. Производится поиск оптимальной модели для решения поставленной задачи. Необходимо оценить точность работы различных моделей и выбрать наилучшую из них.

3. Применение и валидация модели. Модель применяется для реального принятия решений, при этом производится оценка ее точности на фактических данных. По прошествии времени модель должна перестраиваться, чтобы отражать произошедшие изменения.

Несмотря на то, что кредитный скоринг предназначен для автоматического принятия решения по выдаче кредитов, сам процесс построения модели для скоринга не может обходиться без непосредственного участия человека на каждом из этапов.

Наиболее очевидным критерием точности классификации клиентов является процент неверной классификации, или интенсивность ошибок, которая рассчитывается как отношение числа случаев неверной классификации к общему числу случаев.

Для задачи классификации с двумя классами это число должно быть между нулем (все случаи классифицированы корректно) и интенсивностью ошибок классификации по умолчанию (присваивающей во всех случаях класс, которому принадлежит большинство клиентов). По ряду причин построенная модель должна иметь меньшую интенсивность ошибок, чем классификация по умолчанию, при этом в реальных приложениях не существует моделей с нулевой интенсивностью ошибок.

Реальная интенсивность ошибок [1] определяется тестированием модели на настоящих данных. Она не может быть определена до тех пор, пока модель не будет протестирована на большом количестве реальных

случаев. Следовательно, в ходе построения модели этот показатель необходимо как-либо оценить.

Собственная интенсивность ошибок определяется как интенсивность ошибок на наборе данных, который был использован для обучения модели.

Для оценки собственной интенсивности ошибок применяется метод удержания тестовых данных [2]: исходный набор данных разделяется на обучающий (использующийся для построения модели) и тестовый (используемый для оценки точности) наборы. Предполагается, что тестовый набор выделяется случайным образом, независимо от самих данных. Определенная таким образом интенсивность ошибок называется тестовой интенсивностью ошибок. Обычно величина тестового набора составляет около 30% от всех данных. При величине тестового набора в 1000 записей тестовая интенсивность ошибок уже является статистически точной оценкой реальной интенсивности.

Введем следующие обозначения для рассмотрения типов ошибок: a – количество «плохих» клиентов, предсказанных верно; b – количество «хороших» клиентов, предсказанных верно; c – количество «плохих» клиентов, предсказанных как «хорошие»; d – количество «хороших» клиентов, предсказанных как «плохие».

Точность классификации можно описать одним числом – общей интенсивностью ошибок:

$$ER = \frac{b+c}{a+b+c+d}. \quad (1)$$

Однако для систем кредитного скоринга необходимо рассматривать отдельно вероятность ошибок первого и второго рода:

$$ER1 = \frac{c}{a+c}, ER2 = \frac{b}{b+d}. \quad (2, 3)$$

Число ER1 (2) характеризует кредитный риск – процент «плохих» клиентов, классифицированных как «хорошие». Значение ER2 (3) характеризует так называемый коммерческие риск, связанный с отказом «хорошим» клиентам.

В статье рассмотрены основные виды кредитной оценки клиентов банка, этапы построения скоринговой системы, а также критерий точности классификации клиентов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Алберт А. Регрессия, псевдоинверсия и рекуррентное оценивание - Москва, 2012. – 226 с.

2. Бабешко, Л. О. Математическое моделирование финансовой деятельности. Учебное пособие / Л.О. Бабешко. - М.: КноРус, 2016. – 224 с.

3. О.И. Лаврушин. Банковское дело: современная система кредитования: учебное пособие. – КНОРУС, 2007. – 264 с.

УДК 004.4

Попов М.С

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ СТРОИТЕЛЬНОЙ КОМПАНИИ

Рязанский государственный радиотехнический университет им. В.Ф. Уткина, г. Рязань

В статье рассмотрена значимость строительных калькуляторов, их актуальность, улучшенный вариант уже существующих строительных калькуляторов и их варианты развития.

ВВЕДЕНИЕ

Оценка современного состояния решаемой проблемы

Строительство собственного дома является ответственным и дорогостоящим делом. Поэтому часто будущие владельцы дома стараются свести до минимума любые лишние траты. Порой это приводит к серьезным ошибкам и недоработкам, что, в свою очередь, зачастую приводит к незапланированным тратам и расходам, которые связаны с исправлением ошибок.

Прежде чем приступить к непосредственному строительству, необходимо провести расчеты характеристик и расходов строительных материалов для той или иной постройки. Этот этап позволит избежать разрушений постройки, деформации ее элементов и прочих негативных факторов. Помимо этого, от качества произведенных расчетов зависит быстрота проведения строительных работ, так как отсутствие какого-либо материала может задержать строительство на неопределенный срок, в связи с тем, что поиск дополнительных материалов в некоторые периоды года, когда идет активное строительство, будет затруднен из-за их дефицита.

В связи с обозначенными проблемами можно отметить, что для их решения часто используются строительные калькуляторы. С помощью них можно произвести следующие расчёты

1. Расход материалов, необходимых для возведения всех основных элементов постройки;

2. Расчет необходимых размеров и параметров элементов;

расчет требуемых характеристик строительных материалов.

3. Также при помощи программного обеспечения для строительной фирмы, включающего строительный калькулятор и другой функционал, можно решить следующие задачи.

1. Помочь покупателю лучше разобраться и составить мнение о покупаемой продукции.

2. Реализовать возможность выбора материалов для строительства.

3. Повлиять на оценку качества или стоимости жилья, которая зависит от выбранных применяемых материалов.

Актуальность и научная новизна

Строительные компании для расчета своих затрат на строительство должны затратить много ресурсов. На текущий момент, нет такого программного обеспечения, включающего строительный калькулятор, которым можно было воспользоваться для того, чтобы проанализировать все свои возможные затраты. Поэтому задача разработки такого программного обеспечения является актуальной и практически значимой.

Научная новизна заключается в том, что существующее программное обеспечение строительных компаний не позволяет подробно проанализировать затраты и оптимизировать их, из-за чего строительные компании сталкиваются с значительным замедлением цикла продаж. Это связано с тем, что их клиенты не могут до конца представлять, что они хотят и на что им рассчитывать. Например, у клиента строительной компании может возникнуть вопрос, какой дом лучше всего будет построить и по материалам, и по себестоимости. Причем, строительный калькулятор и разработанное программное обеспечение можно спроектировать таким образом, чтобы и сам клиент мог проанализировать все интересующие его варианты и выбрать для себя наилучший. А научная значимость этого исследования направлена для решения поставленных задач и для их решения применяется подход, в котором все алгоритмы, которые реализуются в этом проекте, будут выполняться на клиентской части, при помощи собственных ресурсов компьютера, что значительно ускорит работу программного обеспечения.

На текущий момент уже есть такая строительная компания, которая заинтересована в данной разработке программного обеспечения строительного калькулятора, имеющего функционал, описанный выше.

Цели и задачи научного исследования

Цели.

1. Разработать калькулятор для строительной компании. Эту цель можно обозначить как ключевую.

2. Оптимизировать и ускорить цикл продаж строительной компании. На втором этапе реализация данной поставленной цели позволит более эффективно построить взаимоотношения между клиентом и строительной фирмой, что позволит добиться лучших результатов при строительстве.

Задачи.

1. Анализ предметной области программного обеспечения для строительной компании. Он необходим для того, чтобы понимать в каком направлении двигаться при решении поставленных целей, выявить функциональность и рамки проекта.

2. Разработка архитектуры программного обеспечения для строительной компании. Этот этап необходим для того, чтобы спроектировать весь функционал программного обеспечения и произвести первоначальную подготовку.

3. Разработка программного обеспечения для строительной компании. На данном этапе проходит основная разработка проекта.

4. Тестирование программного обеспечения для строительной компании. На этом этапе будут проведены различные тесты, на основе которых будут устранены дефекты и различные неточности в программном обеспечении.

5. Разработка документации программного обеспечения для строительной фирмы. Этап описания документации (руководства оператора, руководства программиста и системного программиста) и различных правил и алгоритмов использования разработанного программного обеспечения.

ОСНОВНАЯ ЧАСТЬ

Сущность работы

1. Анализ предметной области программного обеспечения для строительной компании.

При анализе предметной области необходимо сравнить уже существующие аналоги, выявить их плюсы и минусы, проанализировать, имеют ли они требуемый функционал.

Первоначальное сравнение нужно провести по наличию следующего функционала.

1. Наличие калькулятора.
2. Возможность рассчитать цену постройки.
3. Возможность рассчитать количество материала.
4. Вариативность (насколько разнообразным можно спроектировать дом при помощи калькулятора).
5. Проектирование дизайна дома.
6. Учет всех производственных условностей (мелкие детали при строительстве).
7. Проектирование фасада.

Рассмотрим десять фирм конкурентов, проанализируем какое программное обеспечение у них используется для решения обозначенных выше задач.

1. Ruplans [0], у данного конкурента отсутствует строительный калькулятор, т.е. данная строительная фирма не реализует требуемый функционал.

Аналогичный вывод можно сделать и о следующих строительных фирмах.

2. Капитал 62 [0].
3. ТАМАК [0].

4. M-strana [0].

5. Наш дом строй [0].

6. Brick House [0].

Остальные конкуренты уже имеют калькулятор, но его функционал значительно меньше, чем тот, которые планируется реализовать в данной разработке.

7. При помощи Stroy-Calc [0] есть возможность рассчитать только количество необходимого материала.

8. Idompk [0] имеет возможность рассчитать цену постройки и спроектировать фасад.

9. Kalk.pro [0]позволяет рассчитать количество материала и при помощи него можно спроектировать наиболее вариативную постройку.

10. Home-projects [0] имеет возможности проектирования дизайна, также в нём присутствует вариативность при проектировании самого дома и учёт всех производственных условностей (сопромата [13, 477]).

На основе всего вышесказанного можно сделать вывод, что ни один из конкурентов не имеет абсолютного преимущества над рассматриваемой строительной фирмой, т.е. у них либо совсем нет программного обеспечения, либо имеющееся программное обеспечение не полностью реализует описанный выше функционал.

2. Разработка архитектуры программного обеспечения для строительной компании.

Сначала необходимо определить за счет чего можно реализовать преимущество рассматриваемой строительной фирмы. Достигнуть поставленной цели можно не только за счёт более широкого функционала, чем у фирм-конкурентов, но также и за счет того, что программное обеспечение будет выполняться на клиентской части, при помощи собственных ресурсов компьютера, что значительно ускорит работу программного обеспечения.

Для разработки программного обеспечения прежде всего потребуется основная html страница и подключение основных css и js файлов, также необходимо подключить к проекту для более качественной и быстрой разработки bootstrap [0] и jquery [0].

3. Разработка программного обеспечения для строительной компании.

При разработке программного обеспечения была свёрстана страница в html файле.

Для элементов были прописаны стили, необходимые для качественного отображения.

В самом js файле был написан алгоритм, реализующий все ранее перечисленные действия.

4. Тестирование программного обеспечения для строительной компании

Никаких критических и прочих ошибок выявлено не было. Разработанное программное обеспечение полностью функционально и реализует все требования, которые были к нему предъявлены.

5. Документация программного обеспечения для строительной компании

Для использования этого программного обеспечения необходим браузер, поддерживается любой из перечисленных ниже:

минимальные Google Chrome, Firefox, Yandex Browser, Internet Explorer Opera;

рекомендуемые Google Chrome, Firefox, Yandex Browser.

Остальными требованиями можно пренебречь.

Далее планируется разработать требуемые по ГОСТ разделы (описание применения программы, руководства оператора, программиста и системного программиста).

Обоснование выбранных методов решения задач

При достижении поставленной задачи было принято решение реализовывать всё на клиентской части, так как этот подход оказался крайне эффективным, при помощи него было достигнуто преимущество над аналогами.

После проведённого эксперимента, были проанализированы результаты работы по таким показателям, как общее время работы программы и время обработки самого результата. Разработанное программное обеспечение показало хорошие результаты, которые заключаются в том, что оно решает задачу намного быстрее, чем существующие аналоги. Кроме того, пользователи, которые тестировали разработанное программное обеспечение, смогли намного быстрее разобраться с интерфейсом и получить результат расчетов.

Описание полученных результатов

В результате разработано программное обеспечение, оптимизирующее цикл продаж и решающее задачи, указанные во введении.

ЗАКЛЮЧЕНИЕ

В результате проведенной научно-исследовательской работы были решены поставленные задачи. В итоге разработки создано программное обеспечение, полностью решающее поставленные задачи, чем обеспечило более эффективное функционирование строительной компании, для которой данное программное обеспечение было разработано.

Также хочется отметить, что разработанное программное обеспечение может использовать не только строительная фирма, но и ее покупатели, что поможет им лучше разобраться и составить мнение о покупаемой продукции, реализовать возможность выбора материалов для

строительства повлиять на оценку качества или стоимости жилья, которая зависит от выбранных применяемых материалов.

Разработанное программное обеспечение также можно будет использовать не только для рассматриваемой фирмы, но и для других фирм, занимающихся аналогичной деятельностью.

В дальнейшем планируется доработать разработанное программное обеспечение.

Его можно дорабатывать в следующих направлениях:

1. Добавить ещё больше функционала, в соответствии с тем, как в будущем будет расширяться предметная область;

2. Можно добавить конструктор дома, чтобы у пользователя был не только калькулятор, но и конструктор, при помощи которого можно собрать свой дом.

3. Также, чтобы у пользователя была возможность загружать свои объекты для дизайна и создавать их.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Ссылки на аналоги:

1. rupla0ns https://ruplans.ru/services/calculater/ (дата обращения: 25.11.2021)
2. Капитал 62 https://kapital62.ru (дата обращения: 25.11.2021)
3. Stroy-Calc https://story-calc.ru (дата обращения: 25.11.2021)
4. ТАМАК https://www.tamak.ru/ (дата обращения: 25.11.2021)
5. m-strana https://m-strana.ru/ (дата обращения: 25 11.2021)
6. наш дом строй https://nashdomstroi.ru/ (дата обращения: 25.11.2021)
7. Brick House https://b-h-c.ru/video/ (дата обращения: 25.11.2021)
8. home-projects https://www.home-projects.ru/calculator/?from=calc_bt (дата обращения: 25.11.2021)
9. idompk https://idompk.ru/proektirovanie-domov (дата обращения: 25.11.2021)
10. kalk.pro https://kalk.pro/ (дата обращения: 25.11.2021)
11. Bootstrap https://getbootstrap.com/docs/5.0/getting-started/introduction/ (дата обращения: 26.11.2021)
12. Jquery https://jquery.com/ (дата обращения: 26.11.2021)
13. Феодосьев В.И. Сопротивление материалов, 1999 - 471-555 стр.

УДК 681.3

Н. Ф. Пахомова, С.В. Крошилина

ВИДЫ АВТОМАТИЗИРОВАННОГО РАСПРЕДЕЛЕНИЯ ЗАКАЗОВ В АТЕЛЬЕ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина», Рязань

В статье рассматриваются возможные виды распределения заказов между портными. Целью работы является рассмотрение видов и в выбор эффективного метода для распределения заказов в ателье.

Отсутствие автоматизированного распределения заказов между сотрудниками – большая проблема. Рассмотрим варианты распределения заявок.

1 Распределение по сотрудникам случайным образом

Самое простое распределение, которое подходит ателье с маленьким штатом — это выдать заявку любому портному случайным образом. Для больших компаний такой подход не подойдет, так как для них нужен более сложный алгоритм, чтобы никто не остался без работы или наоборот с перегрузкой.

Чтобы настроить подобное автоматическое распределение, заказам нужно присваивать статусы, после чего указывать всех сотрудников, которые могут быть выбраны в качестве ответственных, а на следующем этапе бизнес-процесс уже сам выберет портного, который будет работать с данным заказом.

2 Второй вариант распределения - распределение по отделам или группам сотрудников случайным образом

Принцип действия такой же, как в предыдущем случае, только заказы распределяются не только по сотрудникам, но и по отделам или группам. Она подойдет для больших ателье, имеющих филиалы в других городах и регионах.

Назначение ответственных сотрудников по очередности

Распределение по очереди - более справедливо, чем случайное. Каждый сотрудник получает равное количество заказов. Чтобы это реализовать, нужно первым делом создать группу “Портные”. В нее добавляются все сотрудники, между которыми будут распределяться заявки.

Как только в системе возникает новый заказ, ответственным за него назначается первый сотрудник, потом второй, третий и так далее. Как только у всех будет по заявке, начинается второй круг заново.

Равномерное распределение

При подобной автоматизации новый заказ отправляется к портному, у которого заявок в настоящий момент меньше всего. Многие

руководители считают именно такое распределение справедливым - кто-то работает быстрее, кто-то медленнее, а так никто не останется без заявок или со слишком большим их количеством. Кроме того, эта автоматизация позволяет выявить более эффективных сотрудников. В нашем случае, сервер получает список портных, которые участвуют в распределении заказов и количество активных заказов у каждого из них. Система определяет, у кого их меньше всего, и именно этому портному достается заявка.

5 Распределение по графику работы портных

Некоторые ателье работают без выходных, для них такой автоматизированный подход идеален. Если есть необходимость быстро и без перерывов реагировать на запросы, то без учета графика работы не обойтись. Для настройки распределения необходимо указать, когда трудятся сотрудники, либо назначить ответственных в выходные дни. Заявки будут распределяться по этим портным автоматически, в случайном порядке, или в порядке очереди - в зависимости от настроек.

6 Распределение по лимиту

Для настройки такой автоматизации руководитель назначает каждому портному лимит, то есть определенное число заказов, которые менеджер может принять в заданное время, например в течение дня, или в течение часа. Редактировать это количество может только руководитель.

В соответствии с этим числом портному будут назначаться новые заказы, только если он еще не исчерпал свой лимит. Если же он выполнил лимит, то заявки он получит только в следующий период времени. Они поступают либо в порядке очереди, либо в конкретные интервалы времени.

7 Распределение по продуктивности сотрудников

В этом случае работает принцип “лучшее - лучшим”. Система оценивает успешность выполнения заказов портных в соответствии со сроками, и составляет из них ТОП-5. Новые заказы поступают к портным из этого списка, потому что именно они выполняют работу быстро. Ко всем остальным заявки переходят в следующую очередь.

Встает проблема выбора варианта для распределения заказа.

При подборе конкретного способа автоматизации или их сочетания необходимо принимать во внимание специфику работы ателье и заказов, которые в нее поступают [3].

Для заказов пошива подойдут простые способы автоматизации - случайное, равномерное распределение или назначение ответственного в порядке очереди. Но если заказы приходят по индивидуальному пошиву, с детальным описанием и примерками, лучше распределить заказ более успешным сотрудникам, которые более качественно работают именно с таким типом.

Без отрыва от распределения заказов должен рассматриваться вопрос о пропуске портного заявки. Ведь вся суть автоматизированного распределения - в организации своевременной и продуктивной работы с клиентом. Если ателье работает с большой нагрузкой, сложными заказами, то рано или поздно заказы на портном начнут зависать часами, а возможно и сутками. Избежать такой проблемы поможет совмещение нескольких типов автоматизаций. Например, первый - обычное простое распределение, а второй уже отслеживает, как ответственный сотрудник отработал заявку.

С распределением заказов портным станет работать легче, а руководитель избавится от лишней рутиной работы по высчитыванию загрузки каждого сотрудника. Автоматизированное распределение заказов обеспечит контроль работы ателье и своевременное выполнение заказов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Багиев, Г.Л. Основы организации маркетинговой деятельности на предприятии: учебник / Г.Л. Багиев — СПб.: Обл. правл. ВНТОЭ, 2018. — 24

2. Лябишев, К. А. Разработка предложений по совершенствованию маркетинговой деятельности // Научно-методический электронный журнал «Концепт». – 2017. – № 6 (июнь).

3. Пахомова Н.Ф., Крошилина С.В. Обзор существующих информационных систем и выбор эффективного способа автоматизации ателье // Математическое и программное обеспечение вычислительных систем: Межвуз. сб. науч. тр. / Под ред. А.Н. Пылькина - Рязань: Издательство ИП Коняхин А.В. (Book Jet), декабрь 2020.-92с.

4. Программа для работы с заявками [сайт] – https://www.bitrix24.ru/

УДК 004.932; ГРНТИ 89.57.35

М.А.Вертепа, Г.В.Овечкин

ИССЛЕДОВАНИЕ СИСТЕМ РАСПОЗНАВАНИЯ ТЕКСТА НА ИЗОБРАЖЕНИИ

Рязанский государственный радиотехнический университет имени В.Ф.Уткина, г. Рязань

В работе рассмотрено оптическое распознавание текста, его методы и алгоритм работы. Приведена сравнительная характеристика библиотек, которые осуществляют оптическое распознавание.

Введение

Распознавание текста на изображениях – это область исследований, в которой делается попытка разработать компьютерную систему, способную автоматически считывать текст с изображений.

В наши дни существует огромная потребность в хранении информации, доступной в формате бумажных документов, на компьютерном запоминающем диске с последующим повторным использованием этой информации в процессе поиска. Один из простых способов сохранить информацию из этих бумажных документов в компьютерной системе – сначала отсканировать документы, а затем сохранить их как изображения. Но для повторного использования этой информации очень трудно читать индивидуальное содержимое и искать содержимое в этих документах построчно и пословно. При этом возникают проблемы, связанные с характеристиками шрифтов символов в бумажных документах и качеством изображений. Из-за этих проблем компьютер не может распознать символы во время их чтения.

В рамках работы выполнен анализ методов распознавания текста на изображении.

Распознавание текста на изображении

Оптическое распознавание символов (OCR) – это электронное преобразование печатных, рукописных или напечатанных текстовых изображений в машинно-кодированный текст [1].

С помощью OCR огромное количество бумажных документов на разных языках и в разных форматах можно оцифровать в машиночитаемый текст, что не только упрощает хранение, но и делает ранее недоступные данные доступными для всех одним щелчком мыши.

Различные шрифты и способы написания одного символа усложняют распознавание. Перед выбором алгоритма OCR изображение должно быть предварительно обработано, чтобы оно было готово к «чтению» [2].

Методы предварительной обработки включают:

1. Устранение перекоса

Если документ не был правильно выровнен при сканировании, возможно, потребуется наклонить его на несколько градусов по часовой стрелке или против часовой стрелки, чтобы текстовые строки были полностью горизонтальными или вертикальными.

2. Удаление пятен

Удаление положительных и отрицательных пятен, сглаживая края.

3. Бинаризация

Преобразование изображения в черно-белое (так называемое «двоичное изображение», потому что есть два цвета). Задача бинаризации выполняется как простой и точный способ отличить текст (или любой другой требуемый элемент изображения) от фона.

4. Удаление строки

Очищение пустых полей и линий.

5. Анализ планировки или «зонирование»

Определение столбцов, абзацев, заголовков и т.д.

6. Распознавание строк и слов

Установление базовой линии форм слов и символов, при необходимости разделение слова.

7. Распознавание скрипта

В документах на нескольких языках сценарий может преобразовываться на уровне слов, и поэтому идентификация сценария имеет жизненно важное значение до того, как соответствующее OCR можно будет использовать для управления конкретным сценарием.

8. Изоляция или «сегментация» символов

Для символов OCR различные символы, связанные артефактами изображения, должны быть разделены, отдельные символы, разбитые на несколько частей на основе артефактов, должны быть связаны.

9. Нормализация

Нормализация соотношения сторон и масштаб.

Существует два основных метода извлечения функций в OCR:

1. В первом методе алгоритм обнаружения признаков определяет символ, оценивая его линии и штрихи (рисунок 1).

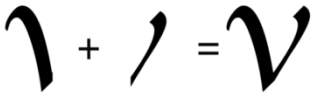

Рис. 1. Обнаружение признаков по первому методу

2. Во втором методе распознавание образов работает путем идентификации всего символа (рисунок 2).

Рис. 2. Распознавание образов по одиночному символу по второму методу

Можно распознать строку текста, выполнив поиск строк с белыми пикселями, между которыми есть черные пиксели. Точно так же можно узнать, где символ начинается и заканчивается.

Затем необходимо конвертировать изображение символа в двоичную матрицу, где белые пиксели равны нулю, а черные пиксели - единицам, как показано на рисунке 3.

Рис. 3. Образец двоичной матрицы

Затем можно найти расстояние от центра матрицы до самой дальней единицы с помощью выражения

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Далее необходимо создать круг этого радиуса и разбить его на более мелкие части. На этом этапе алгоритм сравнивает каждый подраздел с базой данных матриц, представляющих символы с разными шрифтами, чтобы статистически идентифицировать символ, который у него наиболее общий (рисунок 4).

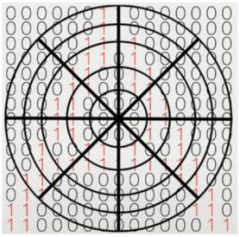

Рис. 4. Сравнение каждого подраздела с матричной базой данных

Точность распознавания текста можно повысить, если ограничить вывод лексиконом (списком слов, разрешенных в документе). Например, это могут быть все слова русского языка или более техническая лексика для определенной области.

Этот метод может быть менее эффективным, если документ содержит слова, которых нет в лексиконе, например, имена собственные.

К счастью, для повышения точности в Интернете есть бесплатные библиотеки OCR. Библиотека Tesseract использует свой словарь для управления сегментацией символов [3].

Выходной поток может быть отдельной строкой или символьным файлом, но более продвинутые системы оптического распознавания текста сохраняют исходную структуру страницы и, например, создают PDF-файл, содержащий как исходные страницы изображений, так и текстовое изображение с возможностью поиска.

Сравнительный анализ библиотек для распознавания

Для выбора библиотеки для распознавания следует провести сравнительную характеристику. Рассмотрим самые популярные библиотеки: Google Text Recognition API, Anyline и Tesseract.

Google Text Recognition API – это процесс распознавания текста в режиме реального времени.

Преимущества:

распознавание текста в реальном времени;

распознавание текста с картинок;

высокая скорость.

Недостатки:

большой размер.

Anyline предоставляет многоплатформенный SDK, который позволяет разработчикам легко интегрировать функции OCR в приложения.

Преимущества:

простая настройка шрифтов;

распознавание текста в реальном времени;

распознавание не только изображений, но и штрихкодов и QR-кодов.

Недостатки:

невысокая скорость распознавания;

библиотека платная.

Tesseract – это OCR библиотека с открытым исходным кодом для разных операционных систем.

Преимущества:

имеет открытый исходный код;

легко обучаема.

Недостатки:

точность распознавания, которая корректируется благодаря обучению;

для обучения в реальном времени требуется дополнительная обработка.

Для распознавания символов лучше всего использовать библиотеку Tesseract OCR несмотря на её недостатки. Этот пакет содержит **двигатель OCR** и **программу командной строки**. Tesseract 4 добавляет новый механизм OCR на основе нейронной сети, который ориентирован на распознавание строк, но также по-прежнему поддерживает устаревший механизм Tesseract OCR Tesseract 3, который работает путем распознавания шаблонов символов. Совместимость с Tesseract 3 обеспечивается с помощью режима Legacy OCR Engine. Также необходимы файлы обученных данных, которые поддерживают устаревший движок, например, из репозитория tessdata.

Tesseract поддерживает **Unicode (UTF-8)** и может **распознавать более 100 языков** «из коробки».

Tesseract поддерживает **различные форматы вывода**: простой текст, hOCR (HTML), PDF, PNG, только невидимый текст, TSV. В главной ветви также имеется экспериментальная поддержка вывода ALTO (XML).

Во многих случаях, чтобы получить лучшие результаты OCR, нужно улучшить качество **изображения, которое** попадает к Tesseract.

Выводы

В ходе рассмотрения оптического распознавания символов были выявлены основные методы, которые необходимо использовать, а также расписан алгоритм действий. В ходе сравнительного анализа библиотек для оптического распознавания текста была выделена лучшая: Tesseract.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Article “Text recognition from images”. Pratik Madhukar Manwatkar, Shashank Yadav. 2015.

2. Article “Applications of OCR You Haven’t Thought Of”. Sukant Khurana. 2018.

3. Распознавание текста с помощью OCR:https://habr.com/ru/post/471542/.

004.032.26

Т.А. Лукьянова

КЛАСТЕРИЗАЦИЯ ДАННЫХ ПРИ ПОМОЩИ НЕЙРОННЫХ СЕТЕЙ

Возможность кластеризации данных с помощью нейронных сетей.

Ключевые слова: алгоритм, нейронные сети, классификация данных, нейрон, веса, карта

Введение

Кластеризация – значимая задача в интеллектуальном анализе данных. Это процесс объединение в группы объектов, обладающих схожими признаками, но при этом ни группы, ни их число заранее не известны и формируются в процессе работы системы исходя из определённой меры близости объектов.

Кластеризация применяется для решения многих задач: от сегментации изображений до экономического прогнозирования и борьбы с электронным мошенничеством.

Существует несколько способов кластеризации данных, например, самоорганизующиеся карты Кохонена[1].

Самоорганизующиеся карты Кохонена

Карты Кохонена – нейросеть, которая использует метод обучения без учителя, то есть результат обучения зависит только от структуры входных данных.

Структура

Самоорганизующаяся карта состоит из нейронов (узлов), обычно упорядоченных в двумерную сеть. Каждый нейрон представляет собой n-мерный вектор-столбец $w = [w_1, w_2, \dots, w_n]^T$, где n обусловливается размерностью входных векторов. Карта Кохонена отображается с помощью ячеек формой прямоугольника или шестиугольника.

Инициализация карты

При исполнении алгоритма заранее задаются конфигурация сетки и число в сети нейронов. В то же время начальный радиус обучения в существенной степени воздействует на способность обобщать, используя получившуюся карту.

Если число нейронов карты больше числа примеров в обучающей выборке, то успех применения алгоритма в сильно зависит от правильного выбора начального радиуса обучения. Тем не менее, если размер карты составляет десятки тысяч узлов, то время, необходимое для обучения карты, обычно очень велико для решения практических задач. Итак, следует добиться приемлемого компромисса при выборе числа нейронов.

Прежде чем начать обучать карту следует установить начальные веса для узлов. Правильно выбранный метод инициализации может значительно ускорить обучение и привести к лучшим результатам. Есть несколько способов инициирования начальных весов:

1) Инициализация всех координат случайными числами;

2) Инициализация примерами, когда значения случайно выбранных примеров из обучающей выборки устанавливаются в качестве начальных значений;

3) Инициализация значениями векторов, линейно упорядоченных вдоль линейного подпространства, которое проходит между парой основных собственных векторов исходного набора данных.

Обучение

Обучение заключается в последовательности исправлений векторов, которые представляют собой нейроны. На каждом этапе обучения произвольным образом избирается один из векторов из исходного набора данных, далее выполняется поиск наиболее похожего вектора коэффициентов нейронов. В этом случае избирается нейрон-победитель[2], который больше всего схож с вектором входов. Сходство в этой задаче относится к расстоянию между векторами, обычно вычисляемому в евклидовом пространстве. Итак, если мы обозначим нейрон-победитель как c, мы получим

$$\|x - w_c\| = min_i \{\|x - w_i\|\}.$$

После того, как нейрон-победитель получен, веса нейронной сети исправляются. В этом случае вектор, который описывает нейрон-победитель и векторы, которые описывают его соседей по сетке движутся в направлении входного вектора.

В то же время формула, которая применяется для изменения весов выглядит так:

$$w_i(t+1) = w_i(t) + h_{ci}(t) \times [x(t) - w(t)],$$

где t – это номер эпохи. В этом случае вектор x(t) находится произвольным образом из обучающей выборки на итерации t. Функция h(t) – это функция соседства нейронов, которая является нерастущей функцией от времени и расстояния между нейроном-победителем и соседними нейронами по сетке. Как правило используется одна из двух функций от расстояния:

1) простая константа – $h(d,t) = \begin{cases} const, & d \le \sigma(t) \\ 0, & d > \sigma(t) \end{cases}$

2) гаусова функция – $h(d,t) = e^{-\frac{d}{2\sigma(t)}}$

Давайте теперь рассмотрим функцию скорости обучения a(t). Она еще является функцией, убывающей от времени. Чаще всего применяются два варианта: линейная и обратно пропорциональная времени вида $a(t) = \frac{A}{t+B}$, где A и B – константы.

Использование такой функции приводит к тому, что каждый вектор из обучающей выборки вносит приблизительно одинаковый вклад в результат обучения.

Обучение включает в себя два важнейших этапа: во время первой фазы выбирается довольно значительное значение скорости обучения и радиуса, что позволяет разместить вектора узлов согласно распределению примеров в выборке, а после этого совершается точная корректировка весов, когда значения параметров скорости обучения значительно меньше начальных. Если применяется линейная инициализация, то начальную фазу грубой настройки можно пропустить.

Применение алгоритма

Этот алгоритм может быть применен для поиска и анализа закономерностей в исходных данных. Вслед за тем, как узлы будут расположены на карте, получившаяся карта может быть отображена[3].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Kohonen, T. Self-Organizing Maps, Springer, 1995.

2. Головко, В.А. Нейронные сети: обучение, организация и применение; под ред. проф. Галушкина, А.И. – Москва:ИПРЖР, 2001

3. Чубукова, И.А. Data Mining. – Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Бином. Лаборатория знаний, 2008.

УДК 004

Козанков И.Ю.

ПОМЕХОУСТОЙЧИВЫЕ КОДЫ, ИСПОЛЬЗУЮЩИЕСЯ В СОВРЕМЕННЫХ СТАНДАРТАХ СВЯЗИ

Рязанский государственный радиотехнический университет им. В.Ф. Уткина, г. Рязань

В данной работе рассматриваются сложные помехоустойчивые коды, которые существуют в настоящий момент. В статье представлено краткое описание каждого из помехоустойчивых кодов и рассмотрено, в каком стандарте и в каких отраслях связи используется тот или иной код.

Помехоустойчивые коды, использующиеся в современных стандартах связи, представляют из себя сложные алгоритмы, которые зарекомендовали себя в поиске и исправлении ошибок, а также в защите от помех, в связи с чем, данные коды в настоящий момент используются повсеместно в телекоммуникационной отрасли. К данным кодам относятся:

1. Код SMPTE. Обладает возможностью самосинхронизации и, как следствие, восстановления поврежденных данных. Код SMPTE является профессиональным кодом и применяется для синхронизации носителей звуковой и видеоинформации.

2. NRZ код является цифровым двоичным кодом, особенностью которого является то, что при передаче нуля данный код передает потенциал, который был установлен на предыдущем такте, а при передаче единицы потенциал инвертируется на противоположный. Благодаря этому данный код может с хорошей точностью распознавать ошибки в передаваемой информации. Из недостатков можно выделить, что этот код не обладает свойством самосинхронизации, а также имеет низкочастотную составляющую [1].

3. Манчестерское кодирование. Передаваемая информация кодируется перепадами потенциала в середине каждого такта, единица кодируется перепадом от низкого уровня к высокому, а ноль – наоборот. В связи с этим данный код обладает хорошей самосинхронизацией, а также в нём отсутствует постоянная составляющая. Манчестерское кодирование применяется в стандарте передачи цифровой информации IEEE802.3.

4. Код Рида-Соломона – это блочный недвоичный циклический код, символы которого представляют собой m-битовые последовательности. Данный код предназначен для исправления одиночных и групповых ошибок, кроме исправления ошибок код Рида-Соломона может также восстанавливать стёртые или же неразборчивые символы. Данные коды используются в таких стандартах связи, как IEEE802.16, Internet, CCSDS и т.п. [1].

5. Биполярный код AMI. Цифровой ноль в данном коде представляется нулевым напряжением, а цифровая единица – остальными значениями отличными от нуля. Благодаря этому код обладает хорошей синхронизацией, а также довольно прост в реализации. Недостатком является низкая скорость передачи данных. Данный код используется в телефонной связи.

6. Код HDB3. Отличается от AMI тем, что для представления цифрового нуля или единицы используется четыре значения в место одного. Данный код также используется в телефонной связи.

7. Код MLT-3 основывается на циклическом переключении уровней напряжения, где единице соответствует переход с одного уровня сигнала на другой. Данный код обладает хорошей синхронизацией и применяется в сетях FDDI, а также в FAST Ethernet 100BASE-TX. [1].

8. Свёрточные коды с применением декодера Витерби являются оптимальными и достаточно легко реализуемыми для коротких свёрточных кодов. Из недостатков можно выделить тот факт, что данный способ применяется только для декодирования коротких кодов, т.к. с ростом длины кода возрастает и его сложность реализации. Данный вид кодирования применяется в беспроводных сетях IEEE802.11, IEEE802.16 дальней космической связи CCSDS, спутниковой связи TIA-1008 и т.п. [3].

9. Свёрточные коды с применением последовательного декодера. Данный способ помехоустойчивого кодирования применяется в отношении свёрточных кодов с большой конструктивной длиной. Из недостатков следует выделить, что данный способ кодирования работоспособен только в области меньшей, чем вычислительная скорость канала, что накладывает серьёзные ограничения на использование этого алгоритма. В частности, последовательное декодирование применяется в стандарте TIA-10008. [3].

10. Каскадные коды. В основе лежит идея совместного использования нескольких составляющих кодов, например код Рида-Соломона, код Хэмминга, код с проверкой на чётность и т.п., широко применяются в таких стандартах связи, как CCSDS, DVB-H/T/S, IEEE802.16 и т.п. [3].

11. Многопороговый декодер самоортогональных кодов (МПДСОК). С помощью данного декодера возможно декодировать очень длинные коды с линейной от длины кода сложностью реализации. Данные алгоритмы используются в таких стандартах связи, как CCSDS, IEEE802.16 и т.п. [3].

12. Свёрточные турбокоды (Turbo Convolutional Code – TCC). Данный вид алгоритмов образуется путём параллельного каскадирования двух кодов через перемежитель. Применяются такие алгоритмы в

основном беспроводной связи в таких стандартах, как CCSDS, TIA-1008, CDMA2000, UMTS.

13. Турбокоды произведения (Turbo Product Code – TPC). Они образуются путём последовательного каскадирования алгоритмов и применяются в таких стандартах связи, как INTELSAT, IEEE 802.16.

14. Низкоплотностные коды (LDPC-коды). LDPC-коды представляют собой линейные блоковые коды, задаваемые с помощью проверочной матрицы H. Особенностью является малая плотность значимых элементов проверочной матрицы, за счёт чего достигается относительная простота реализации средств кодирования. Данные коды применяются в таких стандартах связи, как DVB-S2, 802.11n, 802.16e. [2], [3].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Королев А.И. Коды и устройства помехоустойчивого кодирования информации. Минск: 2002. 286 с.

2. Верификация LDPC-кодов / Н.В. Астахов, А.В. Башкиров, А.С. Костюков, М.В. Хорошайлова, О.Н. Чирков // Вестник Воронежского государственного технического университета. 2017. Т. 13. № 1. С. 74 – 77.

3. Зубарев Ю.Б., Овечкин Г.В. Помехоустойчивое кодирование в цифровых системах передачи данных // Электросвязь. 2008. № 12. С. 58 – 61.

УДК 004.85

Кирилова Л.М.

ФУНКЦИЯ АКТИВАЦИИ ИСКУССТВЕННОГО НЕЙРОНА И АРХИТЕКТУРА ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ

Рязанский государственный радиотехнический университет им. В.Ф. Уткина, г. Рязань

Функция активации искусственного нейрона

Функция активации – функция, результатом вычисления которой является выходной сигнал искусственного нейрона. В качестве аргумента принимает сигнал Y, получаемый на выходе входного сумматора. Можно выделить три основных типа функций активации.

1.Функция единичного скачка, или пороговая функция. В этой модели выходной сигнал нейрона принимает значение 1, если выход нейрона неотрицателен, и 0 – в противном случае. Этот тип функции показан на рисунке 1, и описывается следующим образом:

$$f(x) = \begin{cases} 1, & \text{если } x \geq 0 \\ 0, & \text{если } x < 0 \end{cases}.$$

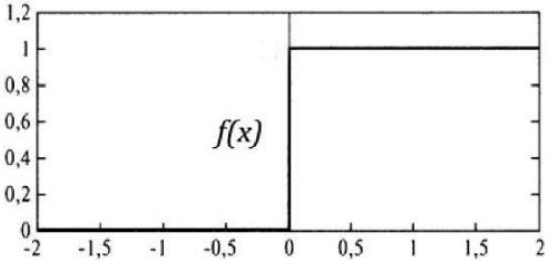

Рисунок 1 – Функция единичного скачка

2. Кусочно-линейная функция. Кусочно-линейная функция, показанная на рисунке 2, описывается следующим выражением:

$$f(x) = \begin{cases} 1, & \text{если } x \geq \frac{1}{2} \\ |x|, & \text{если } \frac{1}{2} < x < -\frac{1}{2} \\ 0, & \text{если } x < -\frac{1}{2} \end{cases}.$$

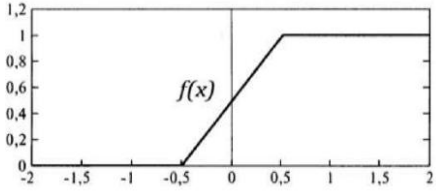

Рисунок 2 – Кусочно-линейная функция

3. Сигмоидальная функция. Сигмоидальная функция, график которой напоминает букву S, является, самой распространенной функцией, используемой для создания искусственных нейронных сетей. Это быстро возрастающая функция, которая поддерживает баланс между линейным и нелинейным поведением. Примером сигмоидальной функции может служить логистическая функция (logistic function), задаваемая следующим выражением: $f(x) = \frac{1}{1+e^{-ax}}$,

где a – параметр наклона сигмоидальной функции (рисунок 3). При уменьшении a сигмоид становится более пологим, при $a = 0$ вырождаясь в горизонтальную линию на уровне 0,5, при увеличении a сигмоид приближается по внешнему виду к функции единичного скачка. Из выражения для сигмоида очевидно, что выходное значение нейрона лежит в диапазоне [0,1]. Также следует отметить, что сигмоидная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того, она обладает свойством усиливать слабые сигналы лучше, чем большие, и предотвращает

насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

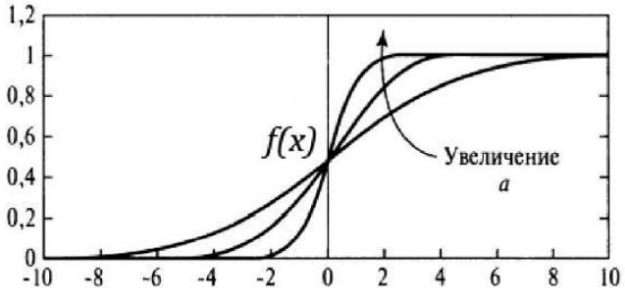

Рисунок 3 – Сигмоидальная функция для различных значений параметра a

Архитектуры искусственных нейронных сетей

Для серьезных нейронных вычислений необходимо соединять нейроны в сети. При этом связь между нейронами определяется архитектурой нейронной сети. В общем случае можно выделить три фундаментальных класса нейросетевых архитектур.

1. Однослойные сети прямого распространения. В нейронной сети нейроны располагаются по слоям. В простейшем случае в такой сети существует входной слой узлов источника, информация от которого передается на выходной слой нейронов (вычислительные узлы), но не наоборот. Такая сеть называется сетью прямого или ациклической сетью. На рисунке 4 показана структура такой сети для случая четырех узлов в каждом из слоев (входном и выходном). Такая нейронная сеть называется однослойной, при этом под единственным слоем подразумевается слой вычислительных элементов (нейронов). При подсчете числа слоев во внимание не принимаются узлы источника, так как они не выполняют никаких вычислении:

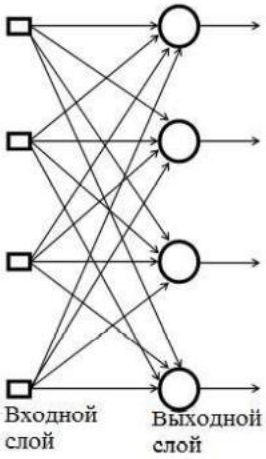

Рисунок 4 – Сеть прямого распространения с одним слоем нейронов

2. Многослойные сети прямого распространения. Другой класс нейронных сетей прямого распространения характеризуется наличием одного или нескольких скрытых слоев, узлы которых называются скрытыми нейронами или скрытыми элементами. Функция последних заключается в посредничестве между внешним входным сигналом и выходом нейронной сети. Добавляя один или несколько скрытых слоев, мы можем выделить статистики высокого порядка. Такая сеть позволяет выделять глобальные свойства данных с помощью локальных соединений за счет наличия дополнительных синаптических связей и повышения уровня взаимодействия нейронов. Способность скрытых нейронов выделять статистические зависимости высокого порядка особенно существенна, когда размер входного слоя достаточно велик. Узлы источника входного слоя сети формируют соответствующие элементы шаблона активации (входной вектор), которые составляют входной сигнал, поступающий на нейроны (вычислительные элементы) второго слоя (то есть первого скрытого слоя). Выходные сигналы второго слоя используются в качестве входных для третьего слоя и так далее. Обычно нейроны каждого из слоев сети используют в качестве входных сигналов выходные сигналы нейронов только предыдущего слоя. Набор выходных сигналов нейронов выходного (последнего) слоя сети определяет общий отклик сети на данный входной образ, сформированный узлами источника входного (первого) слоя. Сеть, показанная на рисунке 5, называется сетью *5-4-2*, так как она имеет 5 входных, 4 скрытых и 2 выходных нейрона. В общем случае сеть прямого распространения с *m* входами, h_i нейронами первого скрытого слоя, h_2 нейронами второго скрытого слоя и *q* нейронами выходного слоя называется сетью *m*-h_i-h_2-*q*.

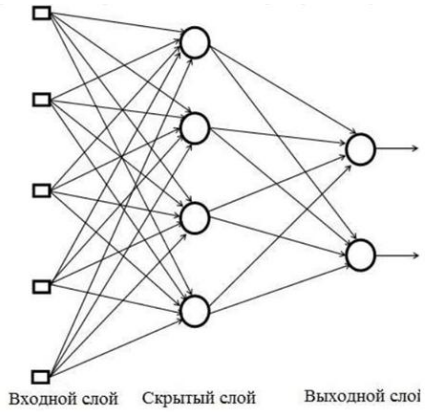

Рисунок 5 – Полносвязная сеть прямого распространения с одним скрытым и одним выходным слоем

Нейронная сеть, показанная на рисунке 5, считается полносвязной в том смысле, что все узлы каждого конкретного слоя соединены со всеми узлами смежных слоев.

3. Рекуррентные сети. Рекуррентная нейронная сеть отличается от сети прямого распространения наличием по крайней мере одной обратной связи. Понятие обратной связи характерно для динамических систем, в которых выходной сигнал некоторого элемента системы оказывает влияние на входной сигнала этого элемента. Таким образом, некоторые внешние сигналы усиливаются сигналами, циркулирующими внутри системы. Обратная связь подразумевает использование элементов единичной задержки, что приводит к нелинейному динамическому поведению, если, конечно, в сети содержатся нелинейные нейроны:

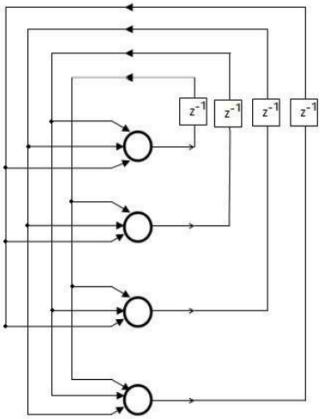

Рисунок 6 – Рекуррентная сеть без скрытых нейронов обратных связей

Например, рекуррентная сеть может состоять из единственного слоя нейронов, каждый из которых направляет свой выходной сигнал на входы всех остальных нейронов слоя. Архитектура такой нейронной сети показана на рисунке 6. Следует обратить внимание, что в приведенной структуре отсутствуют обратные связи нейронов с самими собой. Рекуррентная сеть, показанная на рисунке 6, не имеет скрытых нейронов. Но существует и другой вид рекуррентных сетей – со скрытыми нейронами. В них обратные связи исходят как из скрытых, так и из выходных нейронов.

УДК 004

Д.И. Калугин

РАЗРАБОТКА АЛГОРИТМА ПРОГНОЗИРОВАНИЯ БЕЙСБОЛЬНЫХ ПОДАЧ

Для прогнозирования результатов матчей был выбран метод Монте-Карло. Под методом Монте-Карло понимается численный метод решения математических задач при помощи моделирования случайных величин. Для получения результата матча необходимо смоделировать набор подач. В среднем за матч делается примерно 250-300 подач.

Чтобы определить результат каждой подачи, необходимо провести шесть испытаний:

X_1 - попадание мяча в страйк-зону (мяч попал в зону, мяч пролетел мимо зоны, мяч попал в игрока с битой);

X_2 - реакция игрока на летящий мяч – игрок взмахнул битой, игрок не взмахнул битой;

X_3 - попадание битой по мячу – игрок ударил битой по мячу, игрок промахнулся;

X_4 - сила удара по мячу – игрок выбил фолл-бол, сингл, дабл, трипл или хоум-ран;

X_5 - выведение игрока в аут – игрок не выведен в аут, игрок получил флай-аут, попаут, граундаут;

X_6 - дополнительные условия – в аут выведены сразу два игрока нападения, игрок на второй или третьей базе перебежал на следующую базу (сэк-флай).

Для моделирования исходов бейсбольных подач необходимо 16 основных коэффициентов, определяющих вероятность каждого из событий:

t_1 – мяч попал в страйковую зону;

t_2 – мяч попал в бэттера;

t_3 – взмах на мяч в страйковой зоне;

t_4 – взмах на мяч вне страйковой зоны;

t_5 – попадание битой по мячу;

t_6 – выбит фолл-бол;

t_7 – выбит сингл;

t_8 – выбит дабл;

t_9 – выбит хоум-ран;

t_{10} – выбит трипл;

t_{11} – поп-аут на фолл-боле;

t_{12} – кража хоум-рана (игрок в поле поймал на лету мяч, который летел за пределы поля);

t_{13} – граунд-аут;

t_{14} – флай-аут;

t_{15} – сэк-флай;

t_{16} – дабл-плей.

Также вводятся 10 дополнительных коэффициентов, корректирующих вероятности появления основных событий. Дополнительными коэффициентами являются:

k_1 – размеры стадиона, на котором проводится матч;

k_2 – номер бэттера в линейке отбивающих;

k_3 – коэффициент усталости питчера;

k_4 – количество занятых баз в текущий момент;

k_5 – количество боллов, сделанных питчером;

k_6 – количество страйков;

k_7 – позиция бэттера на поле;

k_8 – «коэффициент рук» - бэттер-правша лучше отбивает подачи от питчера-правши и наоборот;

k_9 – номер иннинга;

k_{10} – число аутов в этом иннинге.

Коэффициент усталости питчера вычисляется на основе следующих коэффициентов:

n_1 – число подач, сделанных в матче этим питчером;

n_2 – мастерство питчера;

n_3 – выносливость питчера;

n_4 – число питчеров, выходивших на горку в этом матче.

Каждая подача моделируется 5000 раз. На основании набора из 5000 результатов подач определяется наиболее вероятный результат подачи. Коэффициенты $k_1 - k_{10}$ остаются неизменными.

Исход события X_1 зависит от k_3, n_2, k_5, k_6. Если $x_1 \in [1, t_1 - (k_3 - n_2) + (k_5 - k_6))$, то исходом X_1 будет попадание мяча в зону, если $x_1 \in [\, t_1 - (k_3 - n_2) + (k_5 - k_6), t_2\,)$, то исходом будет промах мимо страйковой зоны, если $x_1 \in [\, t_2, 3000]$, будем считать, что мяч попал в бэттера.

Исход события X_2 зависит от k_5, k_6. Вероятность взмаха на мяч, летящий в страйковую зону выше. Если $x_2 \in \left[1, t_j + (k_5 - k_6)\right)$, то бэттер проигнорирует этот мяч, если $x_2 \in \left[t_j + (k_5 - k_6), 100\right]$, то бэттер взмахнёт битой. Испытание X_2 не проводится, если мяч попал в бэттера. Если мяч летит в зону, то $j = 3$, в противном случае $j = 4$.

Исход события X_3 зависит от $k_2, k_3, k_5, k_6, k_8, n_2$. Также на исход события X_3 влияет исход события X_2. Испытание X_3 не проводится, если бэттер не взмахнул битой. Если $x_3 \in [1, t_5 + n_2 - k_2 - k_3 - (k_5 - k_6) - k_8)$, то бэттер промахнётся мимо мяча, если $x_3 \in [t_5 + n_2 - k_2 - k_3 - (k_5 - k_6) - k_8, 2000]$, то бэттер попадёт битой по мячу.

Исход события X_4 зависит от $k_1, k_2, k_3, k_4, k_5, k_6, k_7$. Также на исход события X_4 влияет исход события X_3. Если игрок на бите является питчером, то коэффициенты t_8 – t_{10} уменьшаются, а t_6 и t_7 увеличиваются.

Если $x_4 \in [1, t_6 - k_2 + k_4 + k_5 + k_6]$, то выбивается фолл-бол, если $x_4 \in (t_6 - k_2 + k_4 + k_5 + k_6, t_6 + t_7 - k_2 + k_4]$, то выбивается сингл; если $x_4 \in (t_6 + t_7 - k_2 + k_4, t_6 + t_7 + t_8 - k_2 + k_1 + k_4)$, то выбивается дабл, при $x_4 = t_6 + t_7 + t_8 - k_2 + k_1 + k_4$, то выбивается граунд-рул дабл (ground-rule double, мяч от земли отлетает за забор). В случае, если $x_4 \in (t_6 + t_7 + t_8 - k_2 + k_1 + k_4, t_6 + t_7 + t_8 + t_9 - k_2 - k_3 + k_1 + k_4)$, то выбивается хоум-ран.

Если $x_4 \in (t_6 + t_7 + t_8 + t_9 - k_2 - k_3 + k_1 + k_4, 2000]$, то бэттер выбивает трипл. Если бэттер не попал по мячу, то испытание X_4 не проводится.

Исход события X_5 зависит от k_9. С каждым иннингом защита устает, вероятность ошибок в защите увеличивается. Эту закономерность отображает коэффициент k_9. Также на исход события X_5 влияет исход события X_4.

Испытание X_5 не проводится, если не проводилось испытание X_4 или если исход испытания X_4 – граунд-рул дабл.

Если исходом испытания X_4 является фолл-бол, то проводится проверка на поп-аут и флай-аут. Если $x_5 \in [1, t_{11} - k_9)$, то будем считать, что произошёл поп-аут, если $x_5 \in [t_{11} - k_9, t_{11} + t_{14} - k_9)$, то произошёл флай-аут. Если $x_5 \geq t_{11} + t_{14} - k_9$, мяч упал в фолл-территорию без ловли.

Если исходом испытания X_4 является хоум-ран, то испытание X_5 проверяет наличие кражи хоум-рана. Если $x_5 \in [1, t_{12}]$, то кражи хоум-рана не произошло, в противном случае происходит кража хоум-рана.

Если исходом испытания X_4 является сингл, дабл или трипл, то проводится проверка на флай-аут и граунд-аут. Если $x_5 \in [1, t_{13}]$, то

произошёл граундаут. Если $x_5 \in (t_{13}, t_{14} + t_{14}]$, то произошёл флай-аут. Если $x_5 \geq t_{13} + t_{14}$, мяч упал в поле без ловли.

Испытание X_6 проводится если было проведено испытание X_5 и его исходом является граунд-аут, поп-аут или флай-аут. Если произошёл флай-аут, то возможно три исхода испытания X_6: сэк-флай, дабл-плей, отсутствие дополнительного исхода. Если $x_6 \in [1, t_{15}]$, $k_{10} < 2$, занята вторая или третья база; то произошёл сэк-флай. Если $x_6 \in (t_{15}, t_{15} + t_{16}]$, $k_{10} < 2$, занята вторая или третья база; то произошёл дабл-плей на флай-ауте.

Если произошёл граунд-аут, то возможно два исхода испытания X_6: дабл-плей или отсутствие дополнительных исходов. Если $x_6 \in [1, t_{16}]$, $k_{10} < 2$, занята первая база; то произошёл дабл-плей. В противном случае остается граунд-аут.

Если исход испытания X_1 – хит-бай-питч, то исходом подачи будет хит-бай-питч. «Страйк» объявляется если бэттер промахнулся по летящему мячу или он не взмахнул на мяч, летевший в страйк-зону. «Болл» объявляется если бэттер не взмахнул на мяч, которой не попал в страйк-зону. Если исход испытания X_5 – отсутствие аута, то исходом подачи объявляется исход события X_4. Исход испытания X_5 объявляется исходом подачи, если исход испытания X_6 – отсутствие дополнительных исходов. В противном случае исходом подачи объявляется исход испытания X_6.

Вывод

В данной статье рассмотрен пример алгоритма прогнозирования бейсбольных подач, определены необходимые испытания для определения исхода каждой подачи. Полученный набор испытаний и совокупность их исходов позволяют описать большинство возможных исходов подач в бейсболе.

СПИСОК ЛИТЕРАТУРЫ

1. Метод Монте-Карло и его точность [Электронный ресурс] // Хабр. URL: https://habr.com/ru/post/274975/ (Дата обращения: 06.05.2021)

УДК 004.85

Гиголо Мпена Ж.В

СОЗДАНИЕ RESTFUL ВЕБ-СЕРВИСОВ С ПОМОЩЬЮ SPRING BOOT, ИСПОЛЬЗУЯ MONGODB.

Рязанский государственный радиотехнический университет им.В.Ф Уткина,г.Рязань

В статье представлено создание Restful веб-сервисов для приема студентов с помощью Spring Boot используя MongoDb и контроллер класса, показаны создающие Restful веб-сервисов автоматического охватывающих все функции CRUD и создающие Restful Api и тестировать его.

Веб-Сервисы - приложения, которые взаимодействуют через интернет с использованием протокола HTTP. REST, Representational State Transfer («Передача состояния представления») – определяется, как архитектура построения веб-приложений, основанная на клиент-серверной модели и использующая кэшируемый протокол. Веб-сервисы Restful, предоставляя вызывающему клиенту API из вашего приложения безопасным, единообразным способом без сохранения состояния. Вызывающий клиент может выполнять предопределенные операции, используя сервис Restful. Такие архитектуры предназначены для передачи и манипулирования данными по сети с помощью предварительного написанного набора операций. На spring boot реализации — это будет просто и быстро. Фреймворк делает всю черновую работу за программиста и оставляет возможность просто сосредоточить усилия на бизнес-проблеме. Основной задачей является написание приложение для приема студентов на spring boot, которое предоставляет RESTFul—Сервисы.

Для того, чтобы создать Rest-сервис, описанный выше необходимо: java (intellij idea и eclipse самые популярные) и также средство для работы с MongoDB.

Использование Spring Initializr для создания проекта и Добавление необходимых зависимостей

Spring предлагает инструмент под названием «Spring Initializr», который поможет ускорить запуск вашего проекта Spring Boot. Доступ к инициализации осуществляется по ссылки https://start.spring.io/ , или сразу в своей любимой среде IDE и также добавлять зависимости. Для создания проекта необходимо выбрать следующие зависимости с использованием spring initilizr,

1)Spring Web —содержит общие веб-утилиты для сред сервлетов и портлетов.

2)Spring Data MongoDB — это позволяет нам создать интеграцию с MongoDB и Spring Boot.

3)Lombok - самый крутой плагин, чтобы оживить java. Никогда больше не пишите другой метод получения или равенства, с одной аннотацией ваш класс имеет полнофункциональный конструктор, автоматизирует переменные журналирования и многое другое.

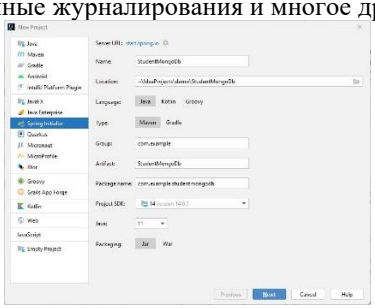

Рис. 1. Интерфейс «создание проект»

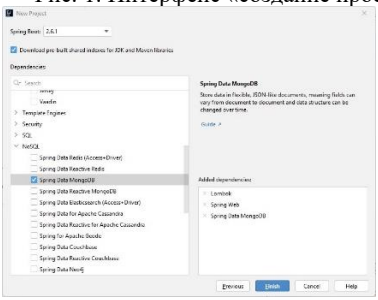

Рис. 2. Интерфейс «добавление зависимости»

Определение классов модели как документов

Теперь, когда проект загружен из Spring Initializr, вы можете открыть в среде IDE, чтобы приступить к редактированию проекта. Во-первых, нам нужно добавить модель Студентов, чтобы Spring знал, какую информацию вернет база данных. Мы можем сделать это, создав новую папку src/main/java/[package name]/

На Spring Boot нужно использовать аннотацию @Document из библиотеки mongodb, чтобы идентифицировать класс java — модели как коллекцию для mongodb. Здесь коллекция — это подобие таблицы в SQL. Итак, здесь документ представляет сущность в JPA.

Можно определить имя пользовательской коллекции при определении документа следующим образом:

@Document(collection = "author_collection").

Модель автора будет представлять набор данных, связанных с автором, в базе данных. Будем добавлять только простые параметры.

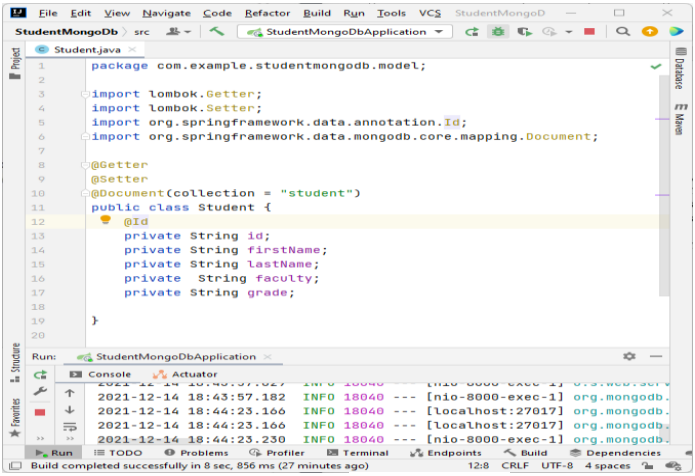

Рис. 3. Модель данные

Модель определит отношение, которое нужно для хранения набора данных внутри API.

Добавление Репозитория

Была создана модель, которая идентифицирует структуру данных, хранящую в базе данных. Для Spring Boot, теперь надо сделать соединитель между моделью и MongoDB.Это через интерфейс репозитория.

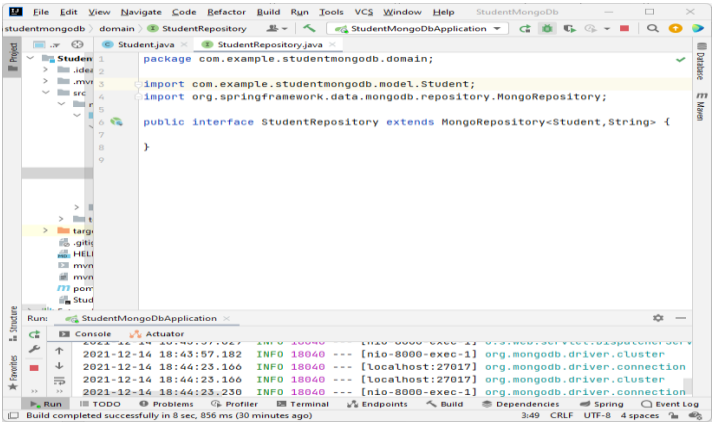

Рис. 3. Добавление Репозитория

Создание контроллера REST и Добавление RESTful Веб-Сервисов

Spring должен иметь возможность подключаться к MongoDB, поэтому нужно установить конечные точки, с которыми будет связаться

для взаимодействия с базой данных. Это будет сделано в контроллере Spring Rest, который использует сопоставления запросов для обработки запросов с помощью функций.

Аннотация @RestController указывает, что этот класс будет контроллером для веб-Сервиса RESTful.

@ResponseBody : указывает, что результат, возвращаемый этим методом, напрямую записывается в тело ответа HTTP, которое обычно используется при асинхронном получении данных и используется для построения RESTful API.

@RequestMapping : предоставляет информацию о маршрутизации, отвечающую за сопоставление URL-адресов с конкретными функциями в контроллере.

@PostMapping:Эта аннотация используется для отображения запросов HTTP POST на определенные методы-обработчики.

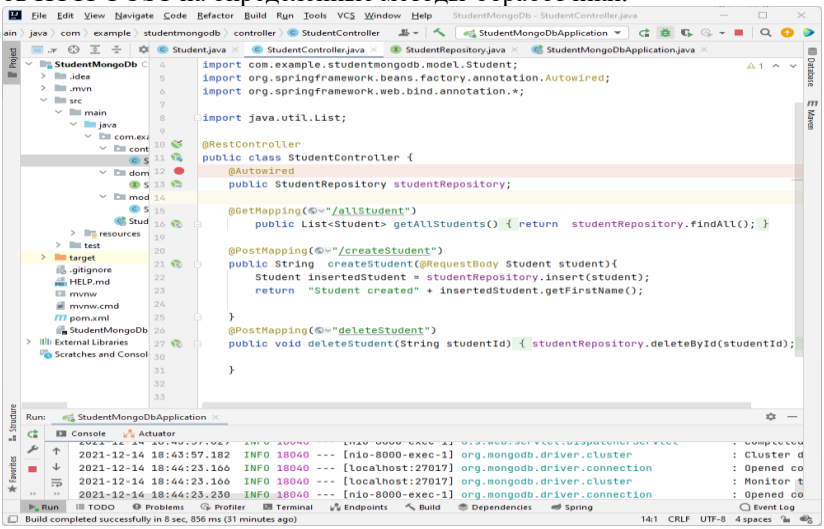

Рис. 4. Создание контроллера Rest

конфигурирование строку подключения с базой данных

Папка ресурсов содержат файл application.properties который использовать для хранения пар свойств.

server.port=8000
spring.data.mongodb.uri=mongodb://localhost:27017/studentdb
spring.data.rest.basePath=/api

Тестирование APi и Веб-Сервисов

У контроллера есть все конечные точки, теперь начать тестирование нашего API на Postman.

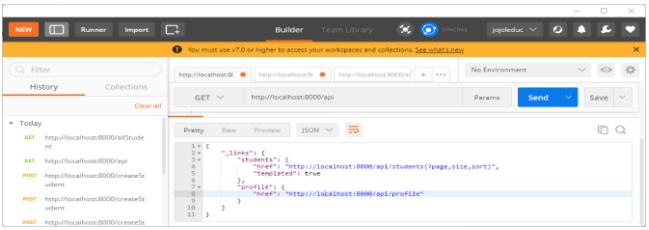

Рис. 5. Проверка ссылку на RESTful Веб-Сервисов

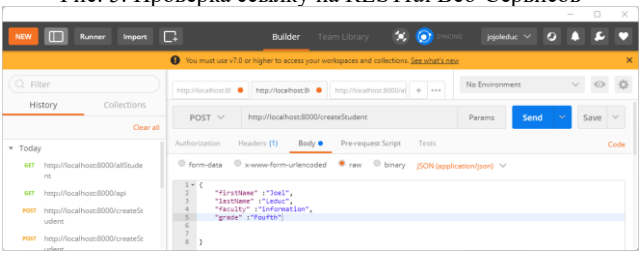

Рис. 6. Добавление новый студент в базе

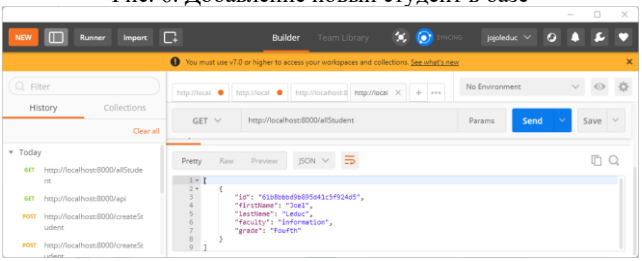

Рис. 7. Получение список студентов из базы данных

В статье был создан Restful Веб-Сервис, который можно использовать как backend для любого приложений на frontend. Такой Сервис позволяет сократить количество ошибок и ускорить работу. Приложение будет надёжным (не нужно сохранять информацию о состоянии клиента, которая может быть утеряна), производительным (за счёт использования кэша) и масштабируемым.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Spring boot Аннотация
// https://bushansirgur.in/spring-annotations/.

Hands-on full stack development with Spring Boot 2 and React build modern and scalable full stack applications 2019, Juha Hinkula

УДК 004.9

Беляев А.В., Дмитриева Т.А.

АНАЛИЗ ПРОГРАММНЫХ СРЕДСТВ ДЛЯ АВТОМАТИЗАЦИИ ДЕЯТЕЛЬНОСТИ МЕДИЦИНСКОГО ЦЕНТРА СТОМАТОЛОГИИ

ФГБОУ ВО «Рязанский государственный радиотехнический университет им. В.Ф. Уткина», г. Рязань

Обоснована тема исследования, изучена предметная область, сформулированы цели и задачи исследования. Проведен анализ существующих решений для автоматизации деятельности медицинского центра стоматологии.

В данной статье рассматриваются трудности и недостатки организации работы с возникающими проблемами в медицинском центре стоматологии. Текущий метод учета проблем заключается в следующем: при возникновении какой-либо проблемы, клиент должен обратиться к медицинскому центру и описать ему суть проблемы, после этого сотрудник центра обращается к вышестоящему лицу, для решения проблемы.

Используемый данный подход имеет ряд недостатков. Например, администратор точно не знает, на месте ли у себя вышестоящее лицо или нет, имеется ли возможность решить проблему клиента сейчас или нет и т.д. Таким образом, очевидна необходимость решения описанной задачи. В качестве решения предлагается разработать автоматизированное программное обеспечение для управления медицинским центр, посредством программного обеспечения 1С: Предприятие.

После внедрения разрабатываемой системы медицинского центра, будут достигнуты следующие технические эффекты: упростится и ускорится процесс информирования между сотрудниками; появится способ информирования клиентов; сократится возможность потери данных и недопонимания.

Это, в свою очередь, приведет к следующему экономическому эффекту [1]: сократится время, затрачиваемое сотрудниками на передвижение в медицинском центре; сократится время, затрачиваемое сотрудниками медицинского центра на обработку заявок.

Таким образом, внедрение разрабатываемого программного обеспечения приведет к положительному экономическому эффекту в виде повышения эффективности работы персонала медицинского центра.

В процессе анализа существующих решений было проведено сравнение разрабатываемого ПО с его альтернативами по следующим критериям: мультиплатформенность; возможность формирования и обработки записей на добавление / исправление / редактирование; возможность формирования отчетов; возможность сбора информации о сотрудниках.

Результаты анализа приведены в таблице.

Таблица – Анализ существующих решений

Программное обеспечение	**Критерий**	**Соответствие критерию**
1С:Медицина.Больница	Платформа	Windows
	Возможность формирования и обработки записей на добавление / исправление / редактирование информации записей 1С.	-
	Возможность формирования отчетов	+
1С:Розница.Аптека	Платформа	Windows
	Возможность формирования и обработки записей на добавление / исправление / редактирование информации записей 1С.	-
	Возможность формирования отчетов	+
1С:Медицина.Больничная аптека	Платформа	Windows
	Возможность формирования и обработки записей на добавление / исправление / редактирование информации записей 1С.	-
	Возможность формирования отчетов	+

По итогам проведенного анализа можно сделать вывод о том, что на рынке нет ни одного ПО, предоставляющего возможность формирования и обработки записей на добавление / исправление / редактирование информации записей в 1С, которое при этом бы не потребляла столько ресурсов. Таким образом, необходимо разработать собственную автоматизированную систему, соответствующую вышеизложенным критериям.

Система будет разрабатываться на языке программирования 1С с использованием технологии обмена данными между базами посредством сериализации и десериализации данных. Используемые средства разработки: конфигуратор 1С: Предприятие 8.3 и легковесный сервер Apache server.

Таким образом, в связи с экономической целесообразностью и отсутствием альтернатив, удовлетворяющих необходимым критериям, разработка собственного программного обеспечения для управления записями для дальнейшей их обработки в медицинском центре для

клиента, посредством приложения 1С: Предприятие является обоснованной.

В результате проделанной работы были достигнуты следующие результаты: проанализирована предметная область; сформулировано обоснование актуальности темы исследования, был выбран объект и предмет исследования; сформулированная цель и поставлена задача исследования; проведен анализ существующих аналогов, выбраны средства для разработки. В дальнейшем будет спроектировано, реализовано и протестировано программное обеспечение для автоматизации медицинского центра стоматологии.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Фаулер М. Архитектура корпоративных программных приложений. М.: Издательский дом «Вильямс», 2007. 544 с.

УДК 004.891.2

Артёмов Я.А.

АЛГОРИТМ СОСТАВЛЕНИЯ МОДЕЛИ СВЁРТОЧНОЙ НЕЙРОННОЙ СЕТИ ДЛЯ РЕШЕНИЯ ЗАДАЧИ КЛАССИФИКАЦИИ

Рязань: Федеральное государственное бюджетное образовательное учреждение высшего образования «Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

В статье описывается один из возможных алгоритмов составления модели свёрточной нейронной сети, а также его тестирование на наборе данных nMnist.

Искусственный интеллект - это одно из наиболее востребованных направлений в области анализа данных, позволяющее решать задачи машинного обучения, одной из которых является задача классификации. Но для правильного решения задачи необходимо составить подходящую модель нейронной сети. Целью задачи классификации является определение принадлежности рассматриваемого объекта к заданным классам на основании признаков объекта. В данной статье будет рассмотрен один из возможных алгоритмов составления модели свёрточной нейронной сети и его применение для создание нейронной сети. Нашей целью является создание эффективной модели свёрточной нейронной сети, при помощи которой будет решаться задача распознавания рукописных цифр на изображениях. Для решения поставленной задачи мы используем средства языка программирования Python, а также методы библиотеки Tensorflow.

Модель свёрточной нейронной сети

Любая модель свёрточной нейронной сети состоит из

Первичная модель нейронной сети составляется на основе предположений и личного опыта разработчика. В данном случае мы будем использовать модель нейронной сети, состоящей из множества

слоев различного вида: сверточные слои (*Convolutions*), слои подвыборки (*Subsampling*) и полносвязные слои (*Fully connected*). Каждый из них выполняет свою функцию. Выходом каждого слоя являются карты признаков (*Feature maps*). Первые два типа слоев(convolutional и subsampling), чередуясь между собой, формируют входной вектор признаков для полносвязного слоя, который в свою очередь и решает, к какому классу относить рассматриваемый объект. Такая архитектура позволяет сети сосредоточиться на низкоуровневых признаках в первом скрытом слое, затем скомпоновать их в признаки более высокого уровня в следующем скрытом слое и т.д. Подобная иерархическая структура - это модель коры головного мозга, отвечающей за зрительное восприятие.

Составление простой модели для свёрточной нейронной сети

Из указанного выше следует, что нейронная сеть будет работать имея только 3 вида слоёв: свёрточный, слой подвыборки и полносвязный слой. Для начала стоит составить простую модель сети из 3 слоёв, что мы и сделаем. Проверять работоспособность модели вы будем на наборе данных nMnist. Нейронная сеть обучалась в течение 10 эпох. В результате мы получили, что нейронная сеть имеет параметр точности равный 0.9844, что означает её способность правильно соотнести объект с его классом в 9844 случаях из 10000. Но если обратить внимание на графики точности (Рисунок 1) и ошибки (рисунок 2) нейронной сети во время обучения, то можно отметить значительное расхождение между показателями точности и ошибки на тестовой и валидиционной выборках. Также можно отметить Эти показатели свидетельствуют тенденцию графика точности к убыванию, а графика ошибки к возрастанию на валидационных выборках ближе к концу обучения нейронной сети. Эти признаки свидетельствуют о низком качестве обучения модели нейронной сети.

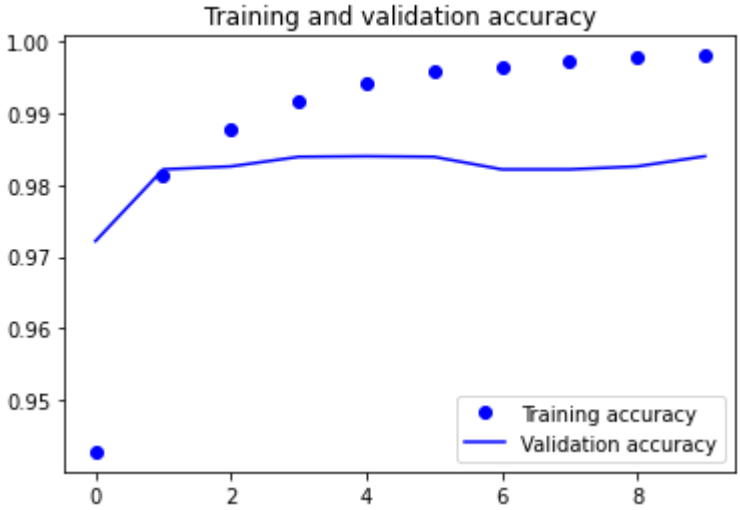

Рисунок 1 - график точности свёрточной нейронной сети с простой моделью во время обучения.

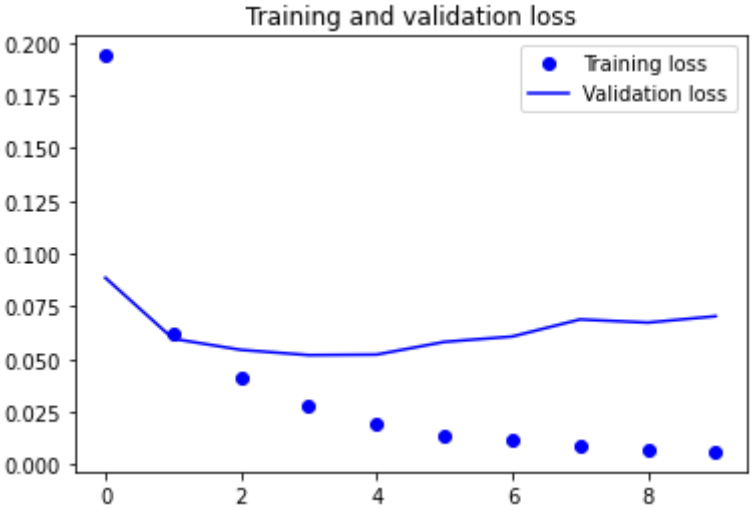

Рисунок 2 - график ошибки свёрточной нейронной сети с простой моделью во время обучения.

Поскольку полносвязный слой служит лишь для классификации и выполняет простую, по сравнению со свёрточным и под выборочным слоями задачу, то усложнение его не имеет смысла и следует сосредоточиться на улучшении двух других типов слоёв.

Доработка модели для свёрточной нейронной сети

Для улучшения качества обучения нейронной сети следует добавить в модель свёрточных и подвыборочных слоёв, что позволит повысить точность распознавание и качество результирующей карты признаков, которая подаётся на вход классифицирующего полносвязного слоя. Для этого мы добавим в модель ещё по 2 слоя каждого вида, тем самым увеличив количество нейронов, что способствует увеличению качества сети. Эти действия позволили нам добиться точности нейронной сети в 99.33%, что означает успешное распознавание 9933 объектов из 10000. Однако, обратив внимание на графики точности (Рисунок 3) и ошибки (Рисунок 4) нейронной сети во время обучения можно заметить скачкообразный вид графиков ближе к концу обучения, что свидетельствует о возникновении эффекта переобучения, который возникает в том случае, если модель нейронной сети начинает сосредотачиваться на побочных признаках объектов и забывать о главных. Появление этого эффекта ведёт к ухудшению модели нейронной сети

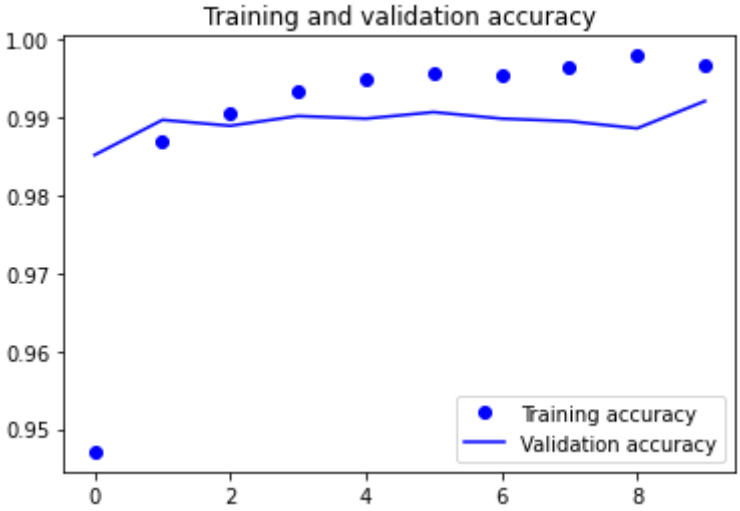

Рисунок 3 - график точности свёрточной нейронной сети с усложнённой моделью во время обучения

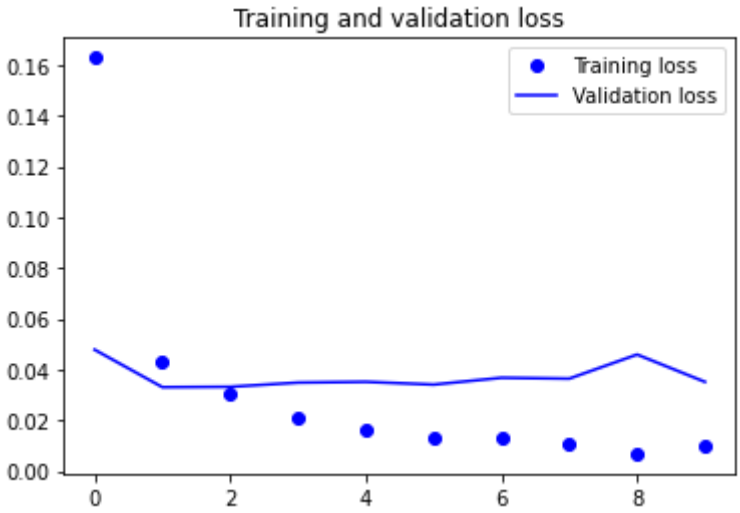

Рисунок 4- график точности свёрточной нейронной сети с усложнённой моделью во время обучения

Для того, чтобы избавиться от этого эффекта мы следует добавить в модель слои DropOut, которые будут случайным образом отключать некоторый процент нейронов из слоя, что позволит предотвратить переобучение нейронной сети. Вышеописанные действия позволили достичь точности нейронной сети в 99.5%, что означает успешное распознавание 9950 объектов из 10000. Также на графиках (Рисунок 5 и рисунок 6) видно, что линии точности и ошибки на валидационной и тестовой выборках теперь близка друг к другу, что свидетельствует о хорошо подобранной модели нейронной сети.

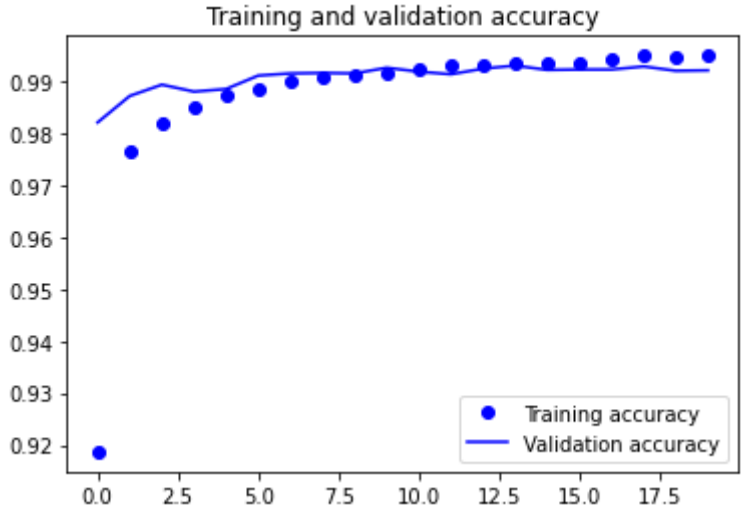

Рисунок 4 - график точности свёрточной нейронной сети с усложнённой моделью без переобучения

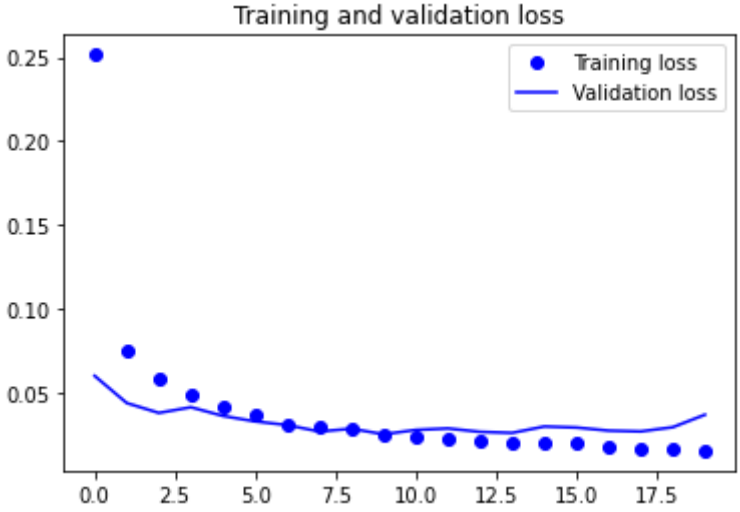

Рисунок 5 - график точности свёрточной нейронной сети с усложнённой моделью без переобучения

Дальнейшее усложнение модели не имеет смысла, поскольку не даст значительного прироста в качестве обучения, но сильно усложнит процесс расчёта весовых коэффициентов.

Алгоритм подбора модели для свёрточной нейронной сети

Таким образом мы пришли к простому алгоритму подбора модели для свёрточной нейронной сети:

Составить максимально простую модель нейронной сети.

Если результаты модели Вас не устраивают, то усложнить “свёрточную” часть модели

Если возникает эффект переобучения, то нужно избавиться от него путём добавления слоёв DropOut или слоёв с активационными функциями.

При помощи этого простого алгоритма у Вас наверняка получится составить подходящую модель для свёрточной нейронной сети.

СОДЕРЖАНИЕ

МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Межвузовский сборник научных трудов

Подписано в печать 27.01.2022. Формат 60x84 1/16.
Бумага офсетная. Печать цифровая.
Гарнитура «Times New Roman».
Усл. печ. л.4.
Тираж 100 экз. Заказ № 4953

Издательство ИП Коняхин А.В. (Book Jet)

Отпечатано в типографии Book Jet
390005, г. Рязань, ул. Пушкина, д. 18
Сайт: http:// book jet.ru
Почта: info@ book jet.ru
Тел.: +7(4912) 466-151

ISBN 978-5-907400-95-5