Санкт-Петербургский государственный университет

Московская Маргарита Дмитриевна

Магистерская диссертация

Создание рекомендательной системы фильмов

Направление 01.04.02 «Прикладная математика и информатика»

Основная образовательная программа ВМ.5759 «Цифровая экономика»

Научный руководитель:
кандидат физ.-мат. наук,
доцент Козынченко В. А.

Рецензент: Ежова Е. В.,
ИПМИ КарНЦ РАН

Санкт-Петербург

2021 год

Содержание

Введение

В современном мире никого не удивить персональными предложениями. Это могут быть предложения скидок как на весь ассортимент, так и на определенный товар, который должен быть наиболее точно подобран под пользователя — адресата предложения. Также, к примеру говоря о развлекательной сфере, людям будет всегда приятно получать уведомления о концертах исполнителей, потенциально им интересных, и чем точнее они соответствуют вкусам каждого человека, тем выше шанс улучшения продаж. Для реализации подобного существует различные решения, применимые как в целом для любой сферы, так и узкоспециализированные. В данной работе будет рассматриваться один из общих вариантов — рекомендательная система.

Рекомендательные системы (РС) — программы, которые пытаются предсказать, какие объекты (фильмы, музыка, книги, новости, веб-сайты) будут интересны пользователю, имея определенную информацию о его предпочтениях [1]. Далее будет идти речь о рекомендательной системе для фильмов, однако теоретически все РС похожи, и объект рекомендаций в большей степени влияет на технические детали при реализации.

Существует множество уже реализованных и применяющихся в реальных условиях систем рекомендаций. К сожалению, невозможно создать одну универсальную систему рекомендации. Во-первых, необходимо учитывать имеющуюся информацию и особенности объектов рекомендаций. Во-вторых, в действительность ни одна система не будет работать с объектами, разными по своему основному определению, то есть рекомендательная система книг не должна уметь рекомендовать блюдо на ужин, в подобном не будет смысла. Также стоит отметить, что рекомендации — субъективная вещь, а следовательно, в данной области все еще существует большой простор для исследований. К примеру, в связи с активным развитием

машинного обучения и нейронных сетей, эти инструменты все более часто применяются для разработки новых и улучшения уже имеющихся рекомендательных систем. Также, далее в данной работе будет предлагаться использовать нейронные сети.

Однако, для начала рассмотрим давно существующие и до сих пор активно использующиеся подходы в рекомендательных системах. В них не применяется сложных техник упомянутого машинного обучения, однако эти системы выдают качественные рекомендации. Итак, существует две основные стратегии создания рекомендательных систем — фильтрация на основе содержания и коллаборативная фильтрация.

Рассмотрим первый вариант — фильтрация на основе содержания. При данном варианте система рекомендации работает с данными об объектах рекомендации, то есть, в контексте этой работы, о фильмах, и может не иметь почти никакой информации о пользователях, помимо их взаимодействий с этими объектами. На основе имеющихся данных создается профиль пользователя, который затем используется для внесения ему предложений. Система выбирает фильмы, аналогичные тем, для которых пользователь уже выразил предпочтение. Поскольку со временем пользователь предоставляет больше входных данных и предпринимает действия по этим рекомендациям, система становится все более и более точной.

Следующий подход — коллаборативная фильтрация [2]. В данном случае рекомендательная система использует принцип, что если пользователям нравились одни и те же, или сильно похожие, объекты, то и дальше их поведение будет совпадать. Соответственно, при данном подходе можно не использовать большого объема информации об объектах, а исходить только из пользовательских оценок объектов. Далее приведем наглядный пример. Пусть есть пользователи U1 и U2, и пользователь U1 посмотрел фильмы M1, M2, M3, M4, а пользователь U2 посмотрел фильмы M2, M3, M4,

M5. Так как для обоих пользователей 3 из 4 фильмов общие (M2, M3, M4), то пользователю U1 можно порекомендовать фильм M5, а пользователю U2 — фильм M1.

В данной работе планируется рассмотреть отличный от описанных выше подход. Большой упор будет делаться на способы обработки информации и средство получения предсказаний пользовательских оценок. Однако, в упомянутых способах фильтрации заложена главная идея, которой стоит придерживаться при создании рекомендательных систем, а именно поиск пользователей с похожими предпочтения и рекомендация им фильмов, которые нравятся одному и не просмотрены другим, а также определение похожесть фильмов для рекомендации людям похожего на просмотренное ими кино.

Говоря о практическом применении, существует множество интернет-ресурсов с информацией о фильмах, на которых пользователи могут выставлять оценки просмотренным фильма. Чтобы не тратить время на поиски следующего фильма для просмотра, у пользователей подобных сайтов есть потребность в рекомендательной системе. Соответственно, созданная в рамках данной работы рекомендательная система может использоваться на упомянутых сайтах в качестве основного ядра для функциональности их системы показа рекомендаций.

Также, затрагивая тему применения полученной в этой работе системы, стоит заметить, что, учитывая различные уже имеющиеся базы данных фильмов и пользователей на интернет-ресурсах про кино, где как раз и возможно использование рекомендательной системы, любая такая система будет нуждаться, как минимум, в дообучении или перенастройке для корректной работы, то есть для хороших рекомендаций пользователям из конкретной базой данных конкретного сайта. И соответственно, рекомендательная система довольно легко может масштабироваться с

помощью обновления баз данных новыми пользователями и фильмами. В зависимости от системы может потребоваться некоторая обработка новых добавленных в базу данных.

В системе, созданной в ходе этой работы в главе 2, средством получения рекомендации будут нейронные сети. Данный подход называется нейронная коллаборативная фильтрация и описан в статье “Neural Collaborative Filtering” [3]. Однако, планируется его модернизировать и также применить больше средств обработки информации, а именно обработку естественного языка [4] для информации о фильмах. Подробное описание планируемого подхода представлено в главе 1.

Постановка задачи и целей

Сформулируем задачу рекомендации фильмов более конкретно и основываясь на информации, имеющийся для ее решения.

Рекомендательная система должна получать на вход уникальный идентификатор пользователя и возвращать ответ в виде списка фильмов и информации насколько каждый из них будет предпочтителен для пользователя. Эта информация является по сути предсказанием системы на основе ее алгоритма и может после интерпретироваться для, к примеру, сообщения пользователю лучших 10 фильмов для него. Рекомендательная система, соответственно, должна рассчитывать предпочтения для всех известных ей фильмов, которые не смотрел пользователь, чей уникальный номер поступил на вход.

Для создания такой рекомендательной системы необходима база данных, которая должна состоять из двух частей. Первая часть содержит список всех пользователей системы и их взаимодействий с фильмами (оценки и просмотры). Вторая часть — подробная и обширная база информации о самих фильмах.

Следовательно, перед созданием рекомендательной системы должны иметься следующие исходные данные:

- вся доступная информация о фильмах: название, краткий сюжет, жанр, ключевые слова, актерский состав, режиссер и прочее.
- информация о пользователях: для каждого пользователя есть конкретный набор некоторого количества оценок, каждая из которых связана с определенным фильмом.

В рамках данной работы планируется подробно рассмотреть и разработать подобную рекомендательную систему, способную по описанным выше данными прогнозировать реакцию каждого известного пользователя на

каждый фильм, который он еще не видел, и информация о котором будет в базе данных системы.

Целью данной работы ставится создание рабочего прототипа рекомендательной системы фильмов. Для ее выполнения следует найти и обработать необходимые для ее работы данные о пользователях и фильмах, изучить и реализовать выбранный подход по получению рекомендаций. Также, есть необходимость оценить его критериями качества для определения успешности полученной системы и возможности ее сравнения с другими рекомендательными системами.

Стоит отметить, что несмотря на уже имеющиеся сервисы о кино и сериалах, которые применяют собственные рекомендательные системы, в большинстве случаев основанные на коллаборативной фильтрации и фильтрации на основе содержания, сама тема рекомендаций и анализа предпочтений пользователей является как никогда актуальной. Во-первых, с момента разработки отмеченных подходов произошел большой технологический скачок. Во-вторых, сами сервисы, в которых пользователи взаимодействуют с различными объектами одной крупной категории, к примеру это могут быть те же фильмы, получили широкое распространение и массовость в связи с развитием интернета. В подобных системах в качестве функционала предоставляется в том или ином виде список объектов, которые пользователю могут понравиться, но он с ними не взаимодействовал. Подводя итог, можно сказать, что данная тема сейчас активно развивается, исследуется, по ней предлагаются различные варианты архитектур и алгоритмов рекомендательных систем на основе новых технологий и новых растущих потребностей.

Обзор литературы

Как было упомянуто, рекомендательные системы активно использующаяся и изучаемая область, поэтому далее будут приведены статьи с различными исследованиями в этой области. Также, в этом обзоре будут представлены книги по общей теории рекомендательных систем. В дополнение, так как в ходе этой работы в практическом эксперименте будут использоваться такие вычислительные средства, как нейронные сети, то далее также будут упомянуты источники информации и о них.

При рассмотрении рекомендательных сетей нельзя не упомянуть книгу “Recommender Systems: An Introduction.” [1] авторов D. Jannach, M. Zanker и других. Как следует из названия, в ней содержится вся необходимая начальная теоретическая информация об рекомендательных системах. Книга включает в себя все вводные понятия этой области, весь необходимый математический аппарат для получения рекомендаций и большое количество иллюстрирующих его примеров, с помощью которых авторы описывают все основные концепции получения рекомендаций. Среди этих концепций присутствуют коллаборативная фильтрация и фильтрация на основе содержания, а также есть обзор гибридных методов. Помимо этого, в книге описаны и сами современные задачи, в которых применяются рекомендательные системы.

Углубляясь в тему рекомендательных систем и новейших исследований в этой области, стоит упоминания статья “Wide & deep learning for recommender systems” [5] авторов H.-T. Cheng, L. Koc, J. Harmsen и других. В этой статье Google представил разработанный для Google Play store ансамбль глубокой и широкой нейронных сетей для прогнозирования рейтингов. В ней подробно описан процесс его создания, обучения, а также приведены результаты эксперимента. Глубокий компонент — это нейронная сеть с прямой связью, а широкая составляющая ансамбля представляет собой

обобщенную линейную модель нейросети. В ансамбле отдельные модели сетей обучаются отдельно, не зная друг друга, и их прогнозы объединяются только во время вывода, но не во время обучения. Авторы особо отмечают, что такая конфигурация нейронных сетей способна запоминать информацию о пользователях, но также она способна к обобщению, то есть видеть сходства между и разными пользователями, и объектами рекомендаций.

Компания Google также выпустила статью под названием “Deep Neural Networks for YouTube Recommendations” [6] о рекомендательной системе для своего другого сервиса, а именно YouTube. Особенность описанной системы заключается в том, что используется две независимые нейронные сети. Первая нейронная сеть на основе пользовательской истории из миллиона не просмотренных им видео отбирает сотни кандидатов для рекомендации, применяя метод ближайшего соседа на результат ее предсказаний. Следующим шагом другая используется нейронная сеть, которая обучена ранжировать отобранные видео с использованием большего объема информации о них.

Внимания также стоит статья “Collaborative Knowledge Base Embedding for Recommender Systems” [7] авторов F. Zhang, N. J. Yuan D. Lian, X. Xie, и W.-Y. Ma. В ней описано применение шумоподавляющего и сверточного автокодировщиков для обработки, соответственно, текстовой и визуальной информации об объектах рекомендации. Вдобавок к этому, в своей системе авторы используют и обрабатывают информацию о структуре и связях этих объектов.

Использование автокодировшиков для рекомендательной системы также описано в статье “Collaborative Deep Learning for Recommender Systems” [8]. В этой публикации автокодировщик используется для выявления сходства и скрытых связей между объектами, а на основе этого далее строятся рекомендации.

Переходя к специфике планируемой в ходе данной работы рекомендательной системы, то при ее создании будет уделяться большое внимание обработки информации, поступающей непосредственно в систему. Конкретизируя, это касается фильмов, а именно текстовых данных о них. В книге Йоава Гольдберга “Нейросетевые методы в обработке естественного языка” [4] подробно изложено как из описания объекта на человеческом языке перейти к понятным программным вариантам их представления, например, к векторному. Помимо этого, в ней указаны все этапы предобработки текстовой информации, необходимые для получения более точного представления объектов. Среди упомянутых этапов присутствуют стемминг и лемматизация. В книге можно найти как теорию об обработке естественного языка, так и наглядные примеры. Автор приводит примеры задач и обоснования применения данной обработки в их решении.

Также, при создании рекомендательной системы в рамках данной работы, будут применяться нейронные сети. В книге “Нейронные сети. Полный курс” [9] автора С. Хайкина подробно описаны всё, что их касается. Там можно найти информацию про персептроны, однослойные и многослойные, и также про другие модели и архитектуры нейросетей. Автор освещает все составные аспекты нейронов, такие как веса связей и функции активации. Также, в книге описаны виды обучения сетей, в частности обучения с учителем и без учителя. Соответственно, автор приводит информацию и про алгоритмы изменения весов при обучении, а именно изложен метод обратного распространения ошибки. Отдельно стоит упомянуть, что книга содержит изрядное количество теоретической информации об моделировании нейронных сетей, выборе их архитектуры и анализе процесса обучения. Помимо этого, в ней много как примеров различных задач, решаемых нейронными сетями, так и в целом примеров их применения.

Глава 1. Теоретическая основа для создаваемой рекомендательной системы

Опираясь на постановку задачи, опишем подход к ее решению, который будет рассматриваться в этой работе.

Для определения подходящих к рекомендации фильмов необходимо проанализировать имеющиеся оценки пользователей и информация о фильмах. Далее, на основе этого будут делаться предсказания для каждого не просмотренного фильма о том, с какой вероятностью конкретный пользователь выберет этот фильм для просмотра. Это будет реализовано с использованием алгоритмов машинного обучения. Некоторыми специалистами были предприняты попытки в создании рекомендательных систем на подобной основе. В статье “Neural Collaborative Filtering” [3] описана модель, которая опирается на коллаборативную фильтрацию, а основой для предсказаний выступает нейронная сеть. Эту модель, названную нейронной коллаборативной фильтрацией, можно использовать при создания рекомендательной системы для любых объектов, с которыми взаимодействуют пользователи, а не только для рекомендации фильмов. Далее опишем ее подробнее.

1.1. Нейронная коллаборативная фильтрация

Нейронная коллаборативная фильтрация (НКФ) — это использование нейронной сети определенной архитектуры для моделирования на основе имеющейся информации собственных векторов объектов и пользователей, а также изучения функции, описывающей их взаимодействие, на основе которой и составляются рекомендации.

Пусть имеется информация о M пользователях и N фильмах, следовательно получим матрицу взаимодействий $Y \in R^{M \times N}$, где для

пользователя u и фильма i элемент $y_{ui} = 1$, если пользователь посмотрел этот фильм, и иначе $y_{ui} = 0$.

Первым шагом, в данную модель подаются два различных вектора, характеризующие соответственно фильм и пользователя. Их моделирование — это отдельная подзадача. Допустимо использовать простой вариант бинарного вектора длины M для пользователя и N — для фильма, в котором компонента, равная 1, присутствует единожды на месте индекса объекта, который этот вектор описывает. Это так называемое one-hot кодирование. Оригинально, то есть в упомянутой в начале главы статье, нейронная коллаборативная фильтрация подразумевает именно такой вариант кодирования данных. Описанные вектора поступаю парой пользователь-фильм на вход нейронной сети. Далее, они по отдельности проходят через независимые нейронные уровни “embedding” для преобразования и уплотнения. На выходе после этого получаем собственные вектора фильма и пользователя. Соответственно, на этом этапе нейронной сети происходит обучение для получения осмысленного векторного представления данных, то есть перевод входных данных (one-hot вектора фильма и пользователя) в пространство признаков заданной длины. Полученные собственные вектора фильма и пользователя объединяются и поступают в следующий полносвязный нейронный слой. Далее, архитектура сети представляет собой полносвязные уровни, количество которых может варьироваться, а количество нейронов уменьшается с увеличением уровня.

Последний уровень, или выход сети, представляет собой однонейронный полносвязный уровень с логистической функцией активации. Соответственно, в качестве ответа сеть выдает вероятность взаимодействия пользователя с фильмом в диапазоне от 0 до 1. Обучается данная сеть с помощью стохастического градиентного спуска. Цель обучения — минимизации функции ошибки между имеющимися значениями матрицы Y и предсказанными $\hat{y}$. При обучении используются не только положительные

примеры, то есть просмотренные фильмы, но и отрицательные — фильмы, с которым конкретный пользователь никак не взаимодействовал.

В рамках данного исследования предлагается отказаться от кодирования фильмов по индексу, а воспользоваться имеющейся текстовой информацией с характеристиками и описанием фильмов для создания собственного вектора фильма вне рамок нейронной сети. Для реализации этого будет использована обработка естественного языка, и подробнее о данном методе написано далее.

1.2. Обработка естественного языка

Обработка естественного языка (Natural Language Processing, NLP) — направление исследований в области искусственного интеллекта, связанное с лингвистикой [4]. Оно изучает проблемы компьютерного анализа и синтеза естественных языков. Применительно к искусственному интеллекту анализ языка означает его понимание, а синтез — генерацию грамотного текста. Главная цель применения обработки естественного языка — добиться максимально хорошего понимания цифровой системой смысла слов таким же образом, как и человек, а не просто как набора символов. Например, слово ключ может иметь не одно значение, а между словами “плакат” и “картина” больше общего, чем между словами “плакат” и “плакать”, несмотря на одинаковые символы во второй паре слов.

Для любого фильма существует множество информации о нем, и большая ее часть является текстовой. Очевидно, что всю ее можно закодировать с помощью 0 и 1. К примеру, информацию о жанрах фильма можно закодировать, создав вектор, с размерностью равной количеству всех имеющихся жанров, где для каждого фильма в векторе будет только одна единица, стоящая на позиции, соответствующей его конкретному жанру, и эта позиция будет одинаковая для всех фильмов этой категории. Такой подход имеет очевидный минус, заключающийся в большой размерности векторов. А если подобным образом кодировать актеров, то возникает вопрос учитывания

актеров, которые сыграли только в одном фильме. Также подобным методом сложно закодировать названия фильмов, которые к тому же могут повторяться. Хотя намного вероятнее, что мы получим слишком большой вектор нулей, в котором будет присутствовать только одна единица, отвечающая за конкретное название, и такой вектор, уникальный для каждого фильма (повторяющиеся названия все же большая редкость), очевидно мало что скажет о самом фильме и причинах определенной оценки пользователем этого фильма. На основе вышесказанного очевидно, что для понимания компьютером текстовой информации о фильмах стоит применить к ней обработку естественного языка.

Перейдем к описанию алгоритму по обработке данных о фильмах, который будет использоваться в этой работы при создании рекомендательной системы.

1.2.1. Векторное представление документов.

Разберем интерпретацию текстовой информации о фильмах в рамках обработки естественного языка. Можно считать, что это набор похожих по содержанию массивов информации, описывающий обширное число однотипных объектов. Поэтому, эту имеющуюся информацию о фильмах имеет место интерпретировать как набор документов. Документ в данном случае является просто набором слов, описывающий фильм. Он формируется с помощью объединения в один массив в определенном порядке всех необходимых данных. Перед формированием документа, из набора должны быть удалены стоп-слова. Ими обычно считаются часто встречающиеся и не несущие смысловой нагрузки предлоги, артикли и прочие слова, которые ухудшают качество векторизации документов за счет внесения зашумления. Далее каждое слово должно быть подвергнуто лемматизации, то есть приведено в нормальную словарную форму, а также стеммингу, что означает выделение основы слова. Предобработка слов необходима для лучшей

векторизации, так как каждый токен несет в себе максимальную смысловую нагрузку.

Полученный массив текстовых токенов и есть документ. В итоге, обработав описанным образом информацию для каждого фильма, получим набор документов. На его основе, с помощь метода Doc2Vec, можно получить собственные вектора для каждого документа, которые можно непосредственно соотнести с соответствующими им фильмами. Суть этого метода заключается в рассмотрении документа как единого целого.

Для начала стоит упомянуть о существовании алгоритма Word2Vec. Для его работы требуется словарь всех слов, которые необходимо преобразовать в вектора. На основе этого словаря, главная составляющая этого метода, а именно — нейронная сеть, учится переводить слова в вектора небольшой заданной размерности так, чтобы близкие по смыслу слова были близко расположены друг к другу в получившемся векторном пространстве. Это достигается путем того, что в качестве обучающих примеров для сети используются предложения, слова из которых и составили словарь.

Однако, в рамках решаемой задачи требуется получить вектор документа. Поэтому, было разработано улучшение метода Word2Vec, а именно Doc2Vec [10]. Их основное отличие в том, что в процессе обучения нейронной сети участвует дополнительный вектор, который обновляется также, как и вектора слов, однако несет в себе информации обо всем документе. То есть, обучающий пример будет содержать в себе помимо образующих документ слов также и его уникальный идентификатор, который и будет преобразовываться в векторное представление документа.

Перейдем к описанию теории о нейронных сетях, участвующих в описанном подходе и необходимых для реализации рекомендательной системы.

1.3. Нейронные сети

Нейронная сеть — математическая модель, представляющая собой связную систему из элементов, называемых нейронами. Эти элементы можно описать нелинейной функцией, называемой функцией активации. Она обозначается F(x), где x — линейная комбинация входных значений, или сигналов. Также, нейронная сеть имеет следующие элементы: входной слой, один или несколько скрытых (вычислительных) слоев и выходной слой. Каждый их них состоит из некоторого количества нейронов. Для выходного слоя их число равно количество параметров, описывающих объекты обучения, а для выходного слоя — информации, которая нейросеть должна возвращать после обработки объекта. Количество нейронов в скрытых слоях является внешним настраиваемым гиперпараметром сети и выбирается эмпирически.

Опишем сам нейрон [11]. Он является вычислительным элементом сети, который получает на вход информацию, вектор входных сигналов и вычисляет значение функции активации на их основе. Полученное значение является или компонентой вектора входных значений для нейронов следующего уровня, или частью выходного значения всей сети. Говоря про функцию активации, то она рассчитывает выходное значение нейрона в зависимости от взвешенной суммы входных сигналов. Веса для этой суммы являются параметрами связи нейронов между уровнями. Подробнее устройство нейрона можно увидеть на рисунке 1.

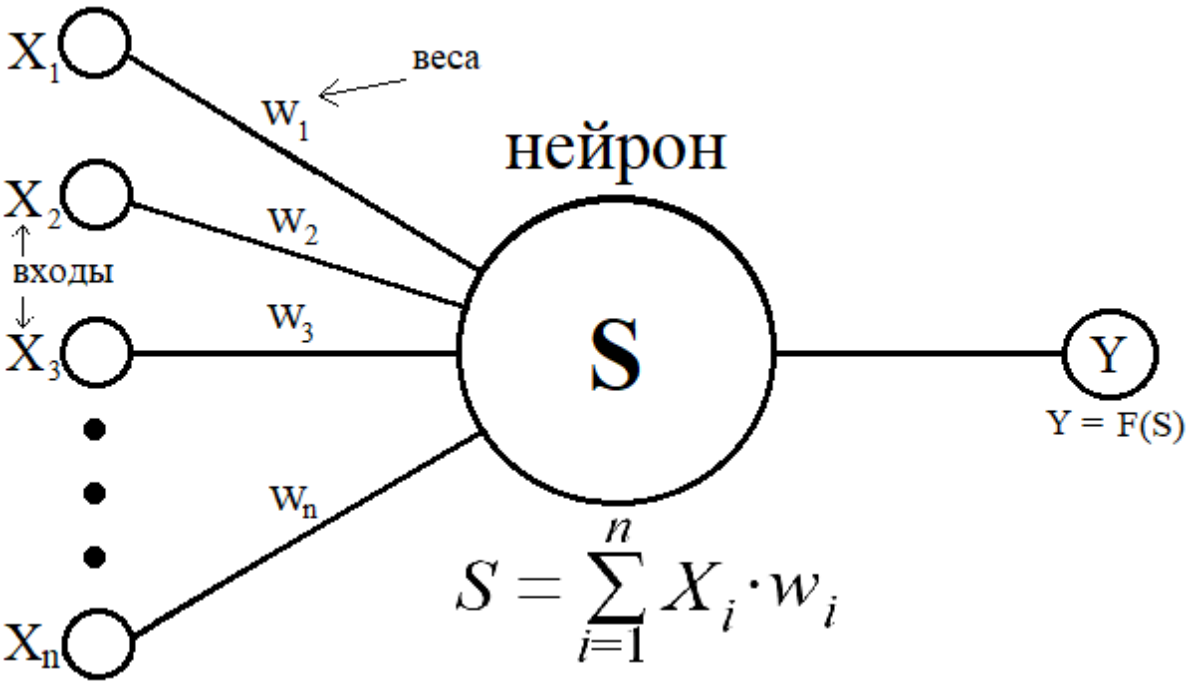

Рисунок 1. Схема искусственного нейрона.

Архитектура нейронной сети очень редко когда состоит из одного нейрона, и часто там больше одного скрытого слоя, поэтому связи между нейронами, которые расположены на соседних уровнях, играют важную роль. Нередко в архитектурах нейросетей считается, что все каждый нейрон одного уровня имеет связи со всеми нейронами предыдущего и следующего слоев. Вес у каждой такой связи уникальный и определяет степень ее влияния, или другими словами — важности.

Одна из самых простых и часто встречающихся сетей с одним скрытым слоем называется персептроном. На рисунке 2 изображена модель нейронной сети под названием многослойный персептрон, потому что скрытых слоев в ней больше 1.

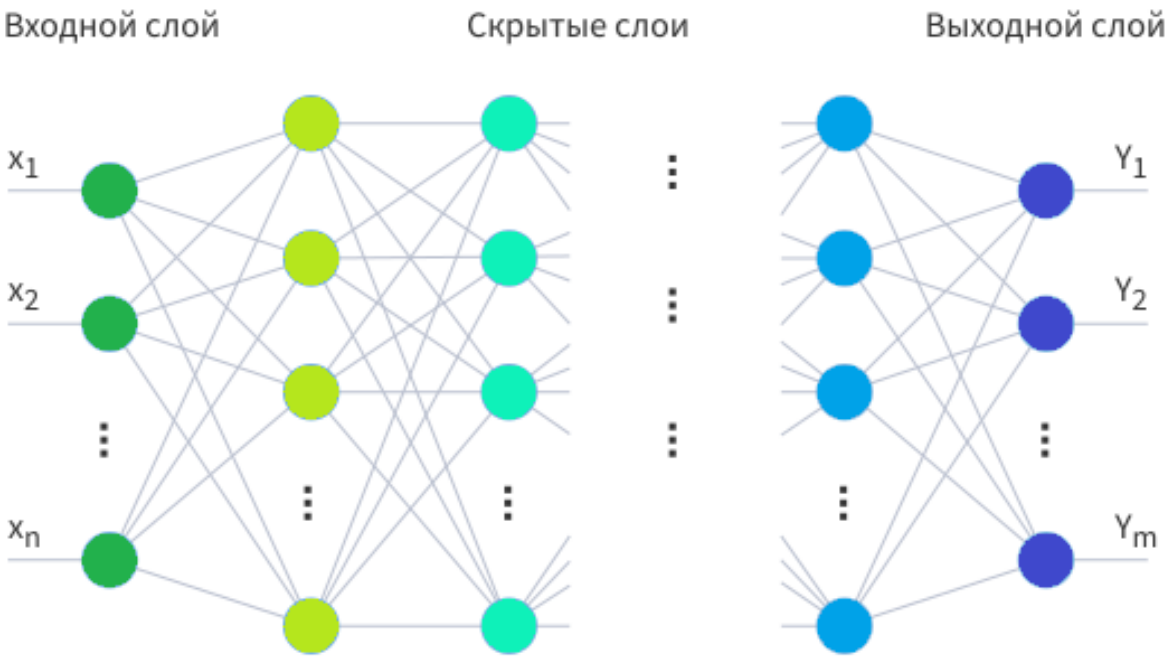

Рисунок 2. Многослойного персептрон.

В ходе практического эксперимента, который будет описан в следующей главе, при построении нейронной сети, будут использоваться следующие функции активации [12], описанные далее.

Сигмоида — нелинейная функция, хорошо подходящая для задач классификации, имеющая фиксированный диапазон значений, и стремящаяся перевести значения к концам этого диапазона. Формула сигмоидальной функции:

$$F(x) = \frac{1}{1+e^{-x}}$$

Функция активации ReLu имеет формулу $F(x) = \max(0, x)$ и так же является нелинейной. Её главное преимущество в легкости вычисления и разряженной активации нейронов, то есть из-за ее свойства возвращать 0 при отрицательных значениях аргумента большое количество нейронов не будут активированы.

Функция активации Softmax служит для обобщения логистической функции, или сигмоиды, для многомерного случая, то есть для задачи классификации с более, чем 2 класса.

Формула функции softmax:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$$

Далее, после того как нейроны с функциями активации собраны в сеть, наступает основной этап, необходимый для ее работы.

1.3.1. Обучение нейронных сетей.

Прежде, чем результатами обработки объектов, подаваемых на вход нейронной сети, можно будет пользоваться, модель и архитектура этой сети должны пройти этап обучения.

В этой работе нейронная сеть будет обучаться с помощью обучения с учителем. При этом методе обучения на вход нейронной сети подается не только непосредственно входные данные, то есть информация о фильме и пользователе, но и ожидаемый ответ сети, в случае поставленной задачи — оценка фильма пользователем. Для реализации обучения с учителем подготавливается специальный набор данных, называемый соответственно обучающим. Его основная особенность в том, что он состоит не только из подготовленных входных данных для нейронной сети, но к каждой единице обучающего набора данных добавляется реальное соответствующее реальное значение, называемое меткой, которое ожидается как выходное значение сети после обучения. Также, часть обучающих данных может не участвовать в обучении, тогда она называется тестовой и используется для оценки качества модели и расчета метрик. Период обучения нейронной сети, за который все обучающие наборы окажутся на входе сети ровно один раз, называется эпохой.

Следовательно, целью и результатом обучения является нейронная сеть, в результате работы которой ее выходные значения максимально приближены к обучающим меткам. Обучение происходит до тех пока, пока не будут достигнуты определенные значения метрик качества и функции ошибки, показывающие, что цель достигнута.

Одними из самых распространённых алгоритмов обучения являются методы первого порядка. В данной работе будет использоваться метод обратного распространения ошибки [13]. Для его применения необходимо рассчитать ошибку нейронной сети, то есть разницу метки и выходного значения, например, по методу наименьших квадратов. Также, для него необходимы вычисления производных функции активации. Соответственно, сам пересчет весов происходит по выбранной вариации метода градиентного спуска [14], то есть уменьшение ошибки сети достигается путем изменения весов по принципу движения в обратную сторону от рассчитанного градиента функций активации.

Пусть E — функция ошибки и для нахождения направления ее минимума необходимо найти градиент этой функций, или $\nabla E(w)$. Следовательно, чтобы вычислить значение изменения для конкретного веса w_{ij}, посчитаем частную производную ошибки по этому весу, или $\frac{\delta E}{\delta w_{ij}}$. Ее расчет происходит по формуле:

$$\frac{\delta E}{\delta w^{q}_{ij}} = \frac{\delta E}{\delta o^{q}_{j}} \frac{\delta o^{q}_{j}}{\delta (S_j)} \frac{\delta (S_j)}{\delta w^{q}_{ij}},$$

где o^{q}_{j} — выходное значение нейрона k и S_j это его взвешенная сумма выходных значений нейронов предыдущего уровня, которые связаны формулами:

$$o^{q}_{j} = \varphi(S_j) = \varphi(\sum_{i=1}^{n} w^{q}_{ij} o^{q-1}_{i}),$$ где q – номер уровня.

Так как в последнем множителе, $\frac{\delta(S_j)}{\delta w_{ij}}$, только одно слагаемое зависит от веса, то:

$$\frac{\delta(S_j)}{\delta w^q{}_{ij}} = \frac{\delta}{\delta w^q{}_{ij}}(w^q{}_{ij} o_i^{q-1}) = o_i^{q-1}$$

Следующий множитель, производная выходного значения нейрона j по его входному значению — это просто частная производная функции активации:

$\frac{\delta w^q{}_{ij}}{\delta(S_j)} = \frac{\delta}{\delta(S_j)} \varphi(S_j)$, где φ — функция активации.

Далее, первый множитель, $\frac{\delta E}{\delta o_j}$, легко считается для нейронов выходного слоя по формуле для производной функции ошибки, так как:

$\frac{\delta E}{\delta o_j^q} = \frac{\delta E}{\delta \hat{y}}$, где $\hat{y}$ — ответ сети.

Рассмотрев функцию ошибки для нейрона j на слое q, как принимающую на вход последовательно все N нейронов предыдущих слоев сети, то получим рекуррентную формулу расчета ее производной по их выходным значениям:

$$\frac{\delta E}{\delta o_j^q} = \frac{\delta E(S_1, S_2, \ \dots \ , S_N)}{\delta o_j^q} = \sum_{i=1}^{N} \frac{\delta E}{\delta o_i^{q-1}} \frac{\delta o_i^{q-1}}{\delta S_i} w^q{}_{ji},$$

Обозначим $\frac{\delta E}{\delta o_j^q} \frac{\delta o_j^q}{\delta(S_j)} = \delta_j^q$. Тогда изменение веса на слое q можно записать формулой:

$$\Delta w^q{}_{ij} = -\eta \frac{\delta E}{\delta w_{ij}} = -\eta \left[\sum_{k=1}^{N} \delta_k^{q+1} w^q{}_{jk}\right] \frac{\delta o_j}{\delta(S_j)} o_i^{q-1} = -\eta \delta_j^q o_i^{q-1},$$

где η — коэффициент, регулирующий скорость обучения в методе градиентного спуска.

Так как значение этого коэффициента достаточно важная часть формулы, помогающая скорректировать веса именно так, чтобы достигнуть минимума функции ошибки, то здесь возможны разные модификации, связанные с применением оптимизаторов.

Далее в работе будет использоваться оптимизатор под названием Adam, суть которого в том, что скорость обучения настраивается для каждого веса связи отдельно с помощью деления коэффициента на скользящие средние значения недавних градиентов и их вторых моментов для этого веса [15].

В итоге формула изменения веса выглядит так:

$$\Delta w^i = -\frac{\eta}{\sqrt{\hat{u}_i + \epsilon}}\hat{m}_i, \text{ где}$$

$$\hat{m}_i = \frac{m_i}{1-\beta_1^t} \text{ и } \hat{u_t} = \frac{u_i}{1-\beta_2^t}, \text{ а}$$

$$m_i = \beta_1 m_{i-1} + (1-\beta_1) \nabla E(w^i),$$

$$u_i = \beta_2 u_{i-1} + (1-\beta_2)(\nabla E(w^i))^2$$

При описании обучения несколько раз была упомянута функция ошибки. Основное назначение данной функции при обучении с учителем — посчитать разницу между выходным значением нейронной сети и ожидаемым результатом, то есть обучающей меткой. На основе выбранной функции ошибки принимается решение о точности модели нейронной сети, окончании обучения, а также в целом делаются выводы об его результатах. В этой работе будет использоваться одна из самых популярных — логистическая функция ошибки.

Формула логистической функции ошибки:

$$E(y, \hat{y}) = -y\log\hat{y} - (1-y)\log(1-\hat{y}),$$

где y — реальное значение, а $\hat{y}$ — значение нейронной сети для обучающего примера с ожидаемым значением, равным, соответственно, y.

Далее, перейдем к описанию не только еще одного аспекта обучения, о и по совместительству следующего за ним этапа. Анализ получаемых в результате работы нейронной сети ее выходных значений.

1.3.2. Метрики.

В процессе и после обучения, помимо значения функции ошибки, для оценки качества модели нейронной сети используются метрики.

В данной работе для оценки качества обучения будут использоваться 3 основные метрики, а именно accuracy, recall и precision. Они являются одними из самых часто используемых. Также, дополнительно были изучены и выбраны для расчётов еще две метрики, mAP и nDCG, более соотносящиеся с рассматриваемой задачей по созданию рекомендательной системы. Перейдем к их краткому описанию.

Для начала объяснения основных метрик далее необходимо ввести 4 понятия. При бинарной классификации можно посчитать следующее:

- число верно предсказанных оценок класса A, или true positive (t_p)
- число неверно предсказанных оценок класса A, или false positive (f_p), то есть отнесенные к классу A объекты класса B
- число верно предсказанных оценок класса B, или true negative(t_n)
- число неверно предсказанных оценок класса B, или false positive (t_n)

Эти числа легко представить в виде матрицы ошибок, где y — метка объекта, а $\hat{y}$ — ответ алгоритма на этом объекте:

	y	$\hat{y}$
y	true positive (t_p)	false positive (f_p)
$\hat{y}$	false positive (t_n)	true negative(t_n)

Соответственно, на основе данных значений можно рассчитать следующие метрики:

- Доля правильно классифицированных объектов, или Accuracy:

$$A = \frac{t_p + t_n}{t_p + t_n + f_p + f_n}$$

- Полнота, или recall:

$$R = \frac{t_p}{t_p + f_n}$$

- Точность, или precision:

$$P = \frac{t_p}{t_p + f_p}$$

Когда классов больше, чем 2, то данные метрики рассчитываются через суммы по каждому классу числителей и знаменателя. К примеру, также использующаяся далее метрика mAP, или mean average precision, рассчитывается по формуле:

$$mAP = \frac{\sum_{i=1}^{N} Precision_i}{N},$$

где N — количество классов в задаче, а $Precision_i$ — точность для класса i.

Перейдем к следующей метрики, которая будет рассчитана для оценки получившейся модели, будет nDCG, или normalized discounted cumulative gain [16]. Эта метрика часто используется при оценке в поисковых системах и ее суть заключается в оценке результатов поисковой выдачи на основе релевантности документов в поисковой выдаче и их позиции в списке. Релевантность означает степень соответствия поискового запроса и конкретного документа в выдаче. Нормализованность этой метрики означает, что чем выше в списке релевантный документ, тем большее его значение будет влиять на итоговое значение метрики и наоборот, соответственно. Можно заметить, что в поставленной ранее задаче по созданию рекомендательной системы, из всех ее предсказаний также важны в основном только первые

фильмы, которые потенциально больше всего понравятся пользователю, если он их посмотрит. Поэтому, nDCG метрику можно применить для оценки рекомендательной системы. Релевантность в данном случае можно рассматривать как предсказанную оценку фильма и ее точность.

Формула DCG:

$$DCG_p = \sum_{i=1}^{p} \frac{r_i}{\log_2(i+1)}$$, что эквивалентно

$$DCG_p = \sum_{i=1}^{p} \frac{2^{r_i} - 1}{\log_2(i+1)},$$

где r_i — релевантность документа на позиции i, а p — финальная позиция.

$IDCG_p$, или идеальная DCG_p, считается по такой же формуле, только документы в выдаче отсортированы по релевантности.

Получаем формулу $nDCG_p$:

$$nDCG_p = \frac{nDCG_p}{IDCG_p}.$$

Подытоживая теоретическую часть, опишем план следующего далее практического эксперимента. Для начала текстовая информация о фильме преобразуется с помощью обработки естественного языка. Затем, используя описанные функции активации, будет разработана архитектура нейронной сети, для которой будут подобраны все гиперпараметры. Далее, исходя из принципов нейронной коллаборативной фильтрации, модель нейросети будет обучена с помощью логистической функции ошибки с применением метода обучения с учителем. Финальным этапом будет расчет и анализ метрик качества полученной обученной модели нейронной сети, которая будет представлять собой требуемую по постановке задачи рекомендательную систему. Планирующиеся использоваться при этом метрики также были описаны в данной главе.

Глава 2. Разработка рекомендательной системы

Разобрав всю теорию, перейдем к практической части — созданию рекомендательной системы. Ее важнейшей составляющей является база данных с информацией о пользователях, объектах рекомендации и их связей один–к-одному в виде оценки или другого индикаторам взаимодействия конкретного человека с определенным фильмом из базы. Перейдем к описанию данных, которые будут использоваться в ходе этой работы при реализации рекомендательной системы.

2.1. Данные о фильмах и пользователях

В качестве базы информации для рекомендательной системы был выбран самый популярный и обширный набор данных “MovieLens” [17]. Этот датасет реальную информацию и был специально собран исследовательской лабораторией для рекомендательных сетей, их изучения, анализа, разработки и экспериментов. “MovieLens” содержит данные о 283228 пользователях и их оценках, которых в сумме 27753444. Имеющиеся оценки относятся к 58098 фильмам, о которых присутствуют краткие сведения: названия и список жанров. Оценки могут принимать значения от 1 до 5 с шагом в 0,5.

Основным файлом данного датасета является “ratings.csv”. Он состоит из 4 колонок: в первых двух колонках записаны уникальные идентификаторы пользователя и фильма, в третьей колонке содержится оценка, которую этот пользователь присвоил указанному фильму, а в четвертой колонке – идентификатор времени данного события.

Для работы с данными использовалась библиотека Pandas для языка программирования Python [18, 19], который также использовался далее на всех этапах практического эксперимента. На рисунке 3 приведены в пример 10 строчек, находящиеся в файле “ratings.csv”.

	userId	movieId	rating	timestamp
1057	10	3253	3.0	948881454
1058	10	3260	3.0	948882230
1059	10	3265	4.0	948882177
1060	11	48	3.0	1112135389
1061	11	158	3.5	1112135405
1062	11	527	4.0	1112135526
1063	11	1193	4.5	1112135548
1064	11	1282	4.5	1112135458
1065	11	1639	4.0	1112135379
1066	11	1722	3.5	1112135463

Рисунок 3. ratings.csv

Так как “MovieLens” содержит малое количество информации о фильмах, то она была дополнена с помощью “The Movies Dataset” [20]. В этом наборе содержится информация о 45 000 фильмов, выпущенных до июля 2017 года включительно. Датасет состоит из файлов формата .csv, которые включают в себя все доступную информацию о фильмах. Основные данные содержатся в файле “movies_metadata.csv”. Там записаны названия, описания, слоганы, жанры, страны производства, бюджеты, кассовые сборы, даты выхода в прокат и многое другое для каждого из фильмов. Однако, их ключевые слова вынесены в отдельный файл “keywords.csv”. Также, целая группа информации про фильмы содержится в другом файле “credits.csv”, а именно актерский состав, режиссер и другие члены команды, и тому подобное На рисунках 4 и 5 приведены примеры необработанных фрагментов данных о трех фильмах. В частности, на рисунке 4 можно увидеть в каком виде записаны жанры фильмов, а рисунок пять демонстрирует самый большой и важный фрагмент имеющейся в датасете информации о фильмах — описание сюжета.

title	tagline	genres
Wrongfully Accused	It's not just a movie. It's every movie.	[{'id': 28, 'name': 'Action'}, {'id': 12, 'name': 'Adventure'}, {'id': 35, 'name': 'Comedy'}]
Next Stop Wonderland	Romance Is Her Destination.	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'name': 'Drama'}, {'id': 10749, 'name': 'Romance'}]
All I Wanna Do	Every Girl Has A Secret Desire	[{'id': 35, 'name': 'Comedy'}]

Рисунок 4. Пример жанров в“The Movies Dataset”.

title	overview
Swing Kids	The story of a close-knit group of young kids in Nazi Germany who listen to banned swing music from the US. Soon dancing and fun leads to more difficult choices as the Nazi's begin tightening the grip on Germany. Each member of the group is forced to face some tough choices about right, wrong, and survival.
Halloween: H20	Two decades after surviving a massacre on October 31, 1978, former baby sitter Laurie Strode finds herself hunted by persistent knife-wielder Michael Myers. Laurie now lives in Northern California under an assumed name, where she works as the headmistress of a private school. But it's not far enough to escape Myers, who soon discovers her whereabouts. As Halloween descends upon Laurie's peaceful community, a feeling of dread weighs upon her -- with good reason.
L.A. Story	With the help of a talking freeway billboard, a "wacky weatherman" tries to win the heart of an English newspaper reporter, who is struggling to make sense of the strange world of early-90s Los Angeles.
The Jerk	After discovering he's not really black like the rest of his family, likable dimwit Navin Johnson runs off on a hilarious misadventure in this comedy classic that takes him from rags to riches and back to rags again. The slaphappy jerk strikes it rich, but life in the fast lane isn't all it's cracked up to be and, in the end, all that really matters to Johnson is his true love.
Dead Men Don't Wear Plaid	Juliet Forrest is convinced that the reported death of her father in a mountain car crash was no accident. Her father was a prominent cheese scientist working on a secret recipe. To prove it was murder, she enlists the services of private eye Rigby Reardon. He finds a slip of paper containing a list of people who are "The Friends and Enemies of Carlotta."

Рисунок 5. Описание сюжета фильмов в “The Movies Dataset”.

Далее опишем структуру рекомендательной системы, какие модули в нее будут входить и как они будут взаимодействовать.

2.2. Схема реализации

Описанные выше датасеты и информация очевидно не могут быть напрямую использованы в рекомендательной системе, поэтому первым этапом будет их предобработка. Целью предобработки будет получить готовый к использованию обучающий набор данных для нейронной сети, которая и будет основным ядром для рекомендаций. На этом этапе вся текстовая информация о кинокартинах будет отобрана и подвергнута обработки естественного языка для получения собственных векторов каждого фильма. Это достигается с помощью готового решение метода Doc2Vec из библиотеки Gensim [21], которая так же предоставляет средства для лемматизации и стемминга. Помимо обучающего набора, результатом этого этапа станет

модель, преобразующая данные о фильмах в вектора, которая будет сохранена для дальнейшего использования.

Следующим этапом будет непосредственная разработка модели нейронной сети, способной обучиться и предсказывать пользовательские оценки для не просмотренных ими фильмов. И, соответственно, далее следует ее обучение. На этих двух этапах будет использоваться библиотека Keras [22] и полученный ранее обучающий набор.

На этом реализацию прототипа рекомендательной системы можно признать завершенной. На вход обученной нейронной сети можно подавать любых пользователя и предобработанный сохраненной Doc2Vec моделью вектор фильма из базы данных. В результате ее работы будет получено предсказание о том, понравиться ли пользователю этот фильм.

Перейдем к описанию непосредственно процесса реализации, который был осуществлен в ходе выполнения поставленной в этой работе задачи.

2.3. Реализация прототипа системы

Как уже было отмечено, первым этапом данные фильмов и пользователей необходимо обработать. Сначала, было произведено объединение всех описанных датасетов и информации в них. В частности, оценки пользователей соотносились с фильмами, которые эти оценки получили.

Отдельно была собрана воедино вся текстовая информация о фильмах. Затем она была проанализирована и только часть данных о фильмах, наиболее показательный сведения, использовались далее для получения векторного представления. Среди них оказались: название, список жанров, ключевые слова, слоган, краткое описание сюжета, актерский состав и продюсеры. Далее эти данные подверглись обработке естественного языка, с их помощью для каждого фильма формировался свой документ. Полученный набор документов

позволил сформировать собственные вектора фильмов. Однако, для последующего обучения нейронной сети, к имеющимся для каждого пользователя спискам просмотренных фильмов необходимо добавить также и не просмотренные им фильмы. В итоге был составлен тренировочный набор для рекомендательной системы, в котором полученные вектора фильма и уникальный номер пользователя образую обучающую пару, а обучающими метками стали 1 — для оцененных кинокартин и 0 — для случайно выбранных фильмов из числа всех остальных, то есть неоцененных, что эквивалентно непросмотренности в контексте данной работы.

При таком подходе могут возникнуть две проблемы: несбалансированность выборки или ухудшение рекомендации, потому что выбранные фильмы для отметки 0, считающиеся негативным примером, в действительности могут оказаться фильмами с высоким шансом на рекомендацию в ситуации, когда они не попали в эту выборку. Следовательно, чем больше число таких фильмов, тем хуже работает система, но маленькое число негативных примеров если и не ухудшит рекомендации, то приведет к несбалансированности выборки, то есть проблеме 1.

Во попытке избежать эти ситуации, рассмотрим систему, где в качестве негативных примеров для каждого пользователя в обучающем множестве были случайно выбраны фильмы из числа им не просмотренных в количестве, равном просмотренным этим человеком кино. Это с одной стороны сбалансирует выборку, а с другой — общее количество фильмов на порядок больше суммы участвующих в обучении примеров для каждого отдельно взятого пользователя, поэтому можно ожидать отсутствие ухудшения результатов работы системы из-за отметки маленького числа фильмов, потенциально пригодных к рекомендации, как негативный пример. В итоге, не просмотренные фильмы выбирались рандомно, для каждого пользователя независимо, из общей базы кинокартин в количестве, равном размеру личного списка просмотренных пользователем фильмов.

После завершения подготовки обучающего набора, следующим этапом была разработка и реализация нейронной сети подходящей архитектуры, в ходе которого были самостоятельно определены такие ее параметры, как количество уровней и размерность их выходов.

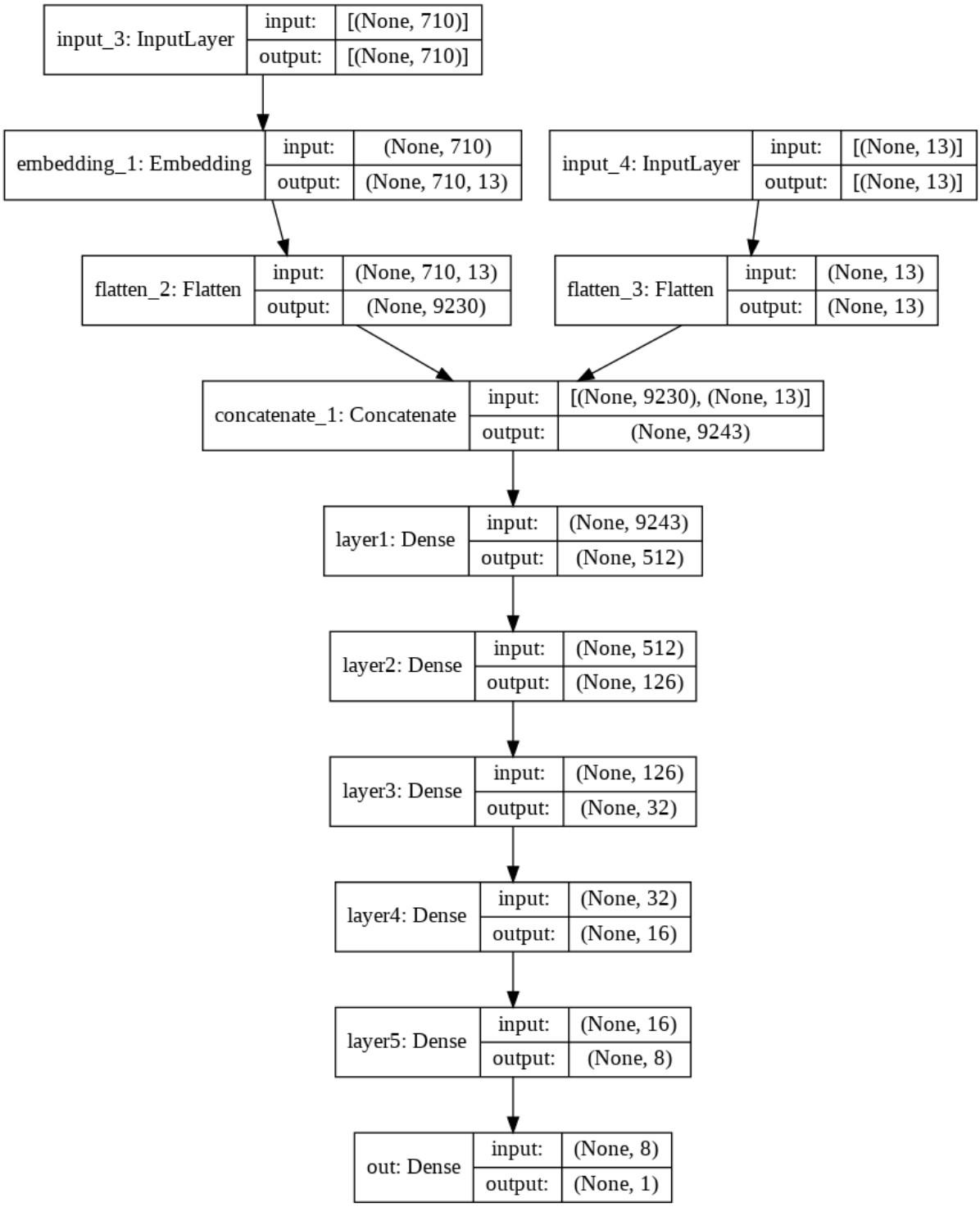

Рисунок 6. Архитектура нейронной сети.

Как показано на рисунке 6, в полученной модели НКФ роль входного слоя выполняют два независимых тензора для фильма и пользователя соответственно. Затем, one-hot вектор, обозначающий уникальный номер пользователя, связан с уровнем векторного представления. Далее нейронная сеть содержит объединяющий уровень, один из входов которого это уже предобработанная информация о фильме, а второй — полученный ранее собственный вектор пользователя. После, в модели следуют 5 полносвязных скрытых уровней с функциями активации в виде ReLU. В конце сети выходной уровень с сигмоидальной функцией активации выдает ответ в виде вероятности взаимодействия пользователя и фильма, на основе которой и осуществляется рекомендация.

Для обучения был выбран оптимизатор “Adam”, оно проходило в 100 эпох, на каждом шаге рассматривалось 64 обучающих примера. Графики обучения представлены на рисунках 7 и 8.

В результате тренировки данной нейронной сети были получены следующие результаты:

- значение функции ошибки = 1.3
- значение метрики “Accuracy” = 0.95

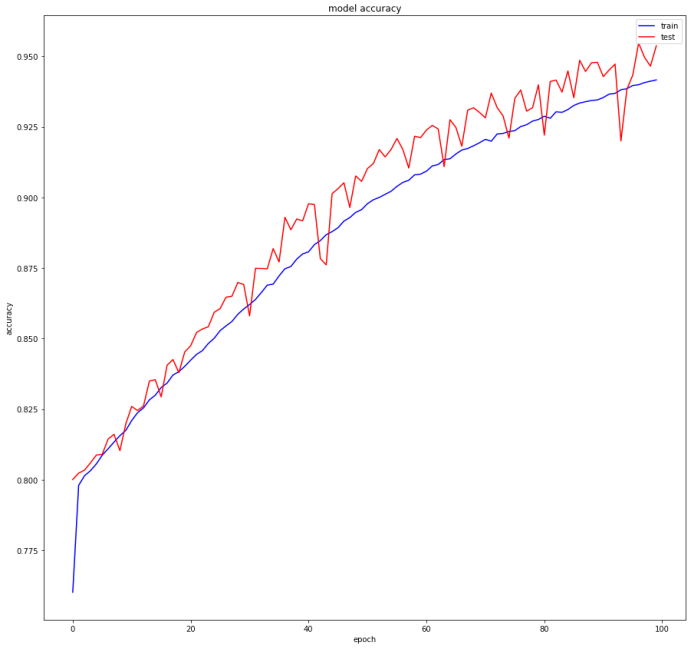

Рисунок 7. График метрики “Accuracy”.

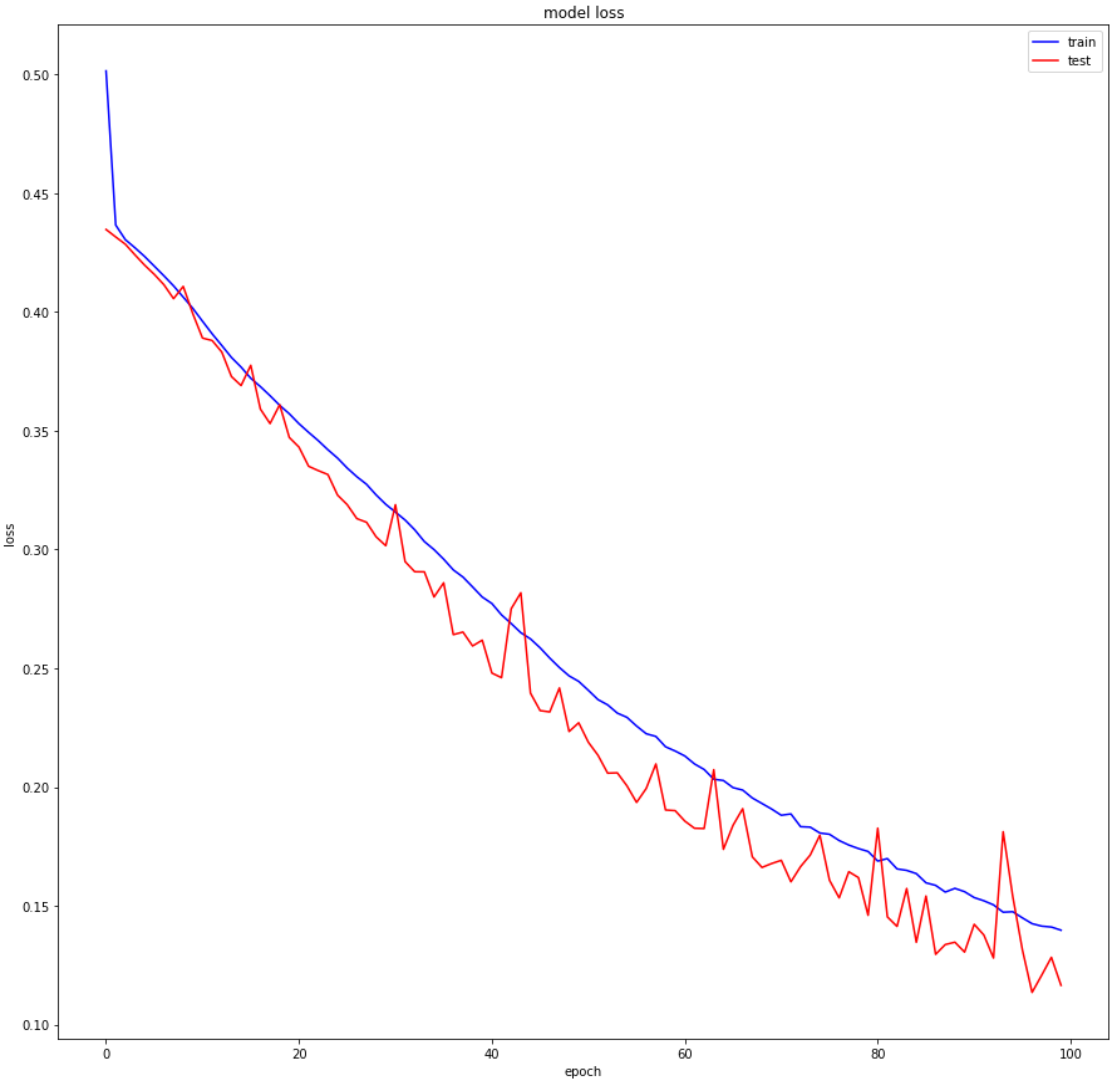

Рисунок 8. График функции ошибки.

Существует уже упомянутая система, разработанная и описанная в статьи “Neural Collaborative Filtering” [23]. Основным различием между ней и полученной в данной работе рекомендательной системы является использование обработки естественного языка, когда как векторное представление фильмов для системы из статьи получается в процессе общей работы нейронной сети для предсказаний пользовательских оценок. В таблице 1 приведены полученные авторами значения метрик. Они посчитаны для 10 самых вероятных для рекомендации фильмов. поэтому часть из них сложно сравнить с результатами аналогичных метрик для разработанной в рамках этой работы системой.

Таблица 1. Значения метрик.

MAP	nDCG@k	Precision@k	Recall@k
0.107720	0.396118	0.347296	0.180775

Для сравнения приведем результаты обученной в ходе описываемого эксперимента системы. Она имеет следующие значения метрик:

- Precision = 0.952
- Recall=0.951
- MAP = 0.2
- nDCG =0.99

В целом, можно считать полученные результаты не вызывающими сомнений в качестве разработанной рекомендательной системы. Значения метрики MAP при сравнении указывают на то, что использование обработки естественного языка для информации о фильмах оказало положительный эффект на результаты предсказаний системы.

Так как, в начале главы были упомянуты проблемы, связанные с разбиением фильмов на два класса, просмотренные и не просмотренные, рассмотрим следующую модификацию системы — изменим способ, которым выставляются метки фильмам. Стоит отметить, что интуитивно и опираясь на уже имеющиеся рекомендательные системы, хочется получать список рекомендаций пользователя в формате рейтинга: от самого лучшего объекта для выбора к менее предпочтительному. Так как общеприняты и активно используются оценки пользователями фильмов по шкале от 1 до 10, то далее изменим полученную рекомендательную систему для предсказания оценки фильма по этой шкале, когда как сейчас она предсказывает вероятность просмотра фильма.

Для начала, в используемом наборе данных уже содержится информация о рейтинге просмотренных фильмов от каждого пользователя для просмотренных им фильмов по шкале от 1 до 5, однако в ней 10 возможных значений оценок, часть из которых — это числа с запятой, то есть 1.5, 2.5 и так далее. Чтобы избавиться от этих значений, а также перейти на десятибалльную шкалу, все имеющиеся оценки умножаются на 2. В рамках обучающего набора полученные пользовательские оценки кодируются в бинарные массивы длины 10, установив в нем цифру 1 на позицию, равную кодируемой оценке.

Изучая полученные десять классов фильмов на предмет их равномерного распределения по этим классам, было выявлено, что люди предпочитают оценивать фильмы на оценки 6, 10, а самая популярная — 8. Полное распределение представлено в таблице 2 далее.

Таблица 2. Распределение фильмов по оценкам.

оценка	количество фильмов	процент от общего числа фильмов
1	1865	2,67%
2	2575	3,68%
3	1192	1,70%
4	5214	7,46%
5	3314	4,74%
6	14355	20,53%
7	7161	10,24%
8	18165	25,97%
9	5844	8,36%
10	10251	14,66%

В целом, представленные данные уже требуют изучения в области человеческой психологии. С точки зрения применения обучающей выборки с таким распределением по классам, то такую выборку нельзя назвать

сбалансированной, значения метрик при обучении могут получиться немного хуже и за этого. Однако, в ситуации рекомендательной системы, когда надо предположить оценки пользователя для не просмотренных фильмов, не имеет большого значения разница в 2 балла между оценками. Этим можно воспользоваться как аргументом для игнорирования факта несбалансированности выборки.

Для новой ситуации в данных и их классификации была изменена архитектура нейронной сети, разработанной на предыдущем этапе.

Во-первых, уменьшилось количество скрытых слоев с 5 до 4. Во-вторых, установлена размерность выходного слоя равную 10, а также выбрана функция активации Softmax, более подходящую к задаче многоклассовой классификации. Остальные параметры, в частности оптимизатор, использующийся при обучении, остались без изменений. Полная архитектура этой модели представлена выше на рисунке 9.

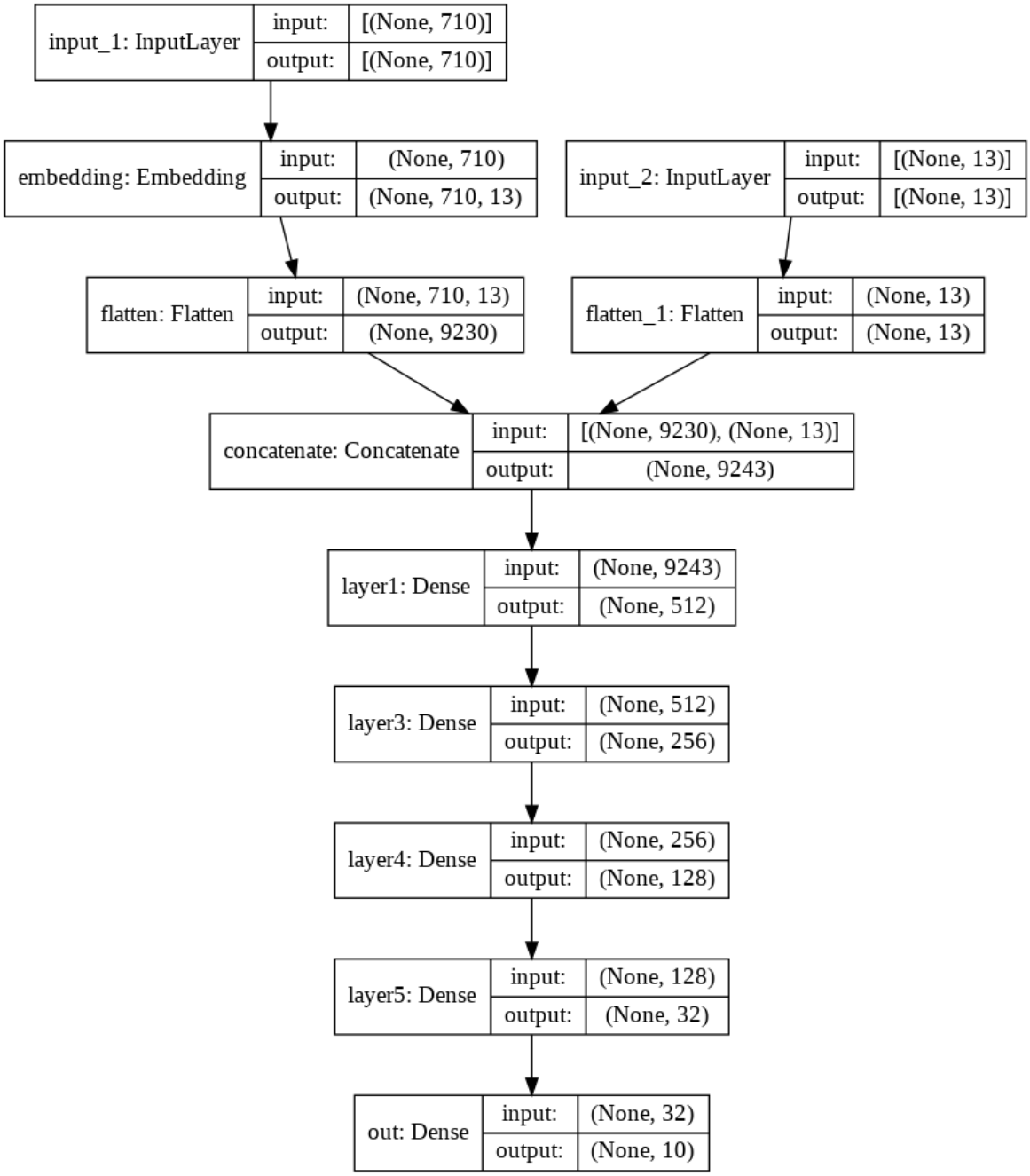

Рисунок 9. Измененная архитектура нейронной сети.

Обучение этой нейронной сети также проходило в 100 эпох. Графики обучения представлены далее на рисунках 10 и 11.

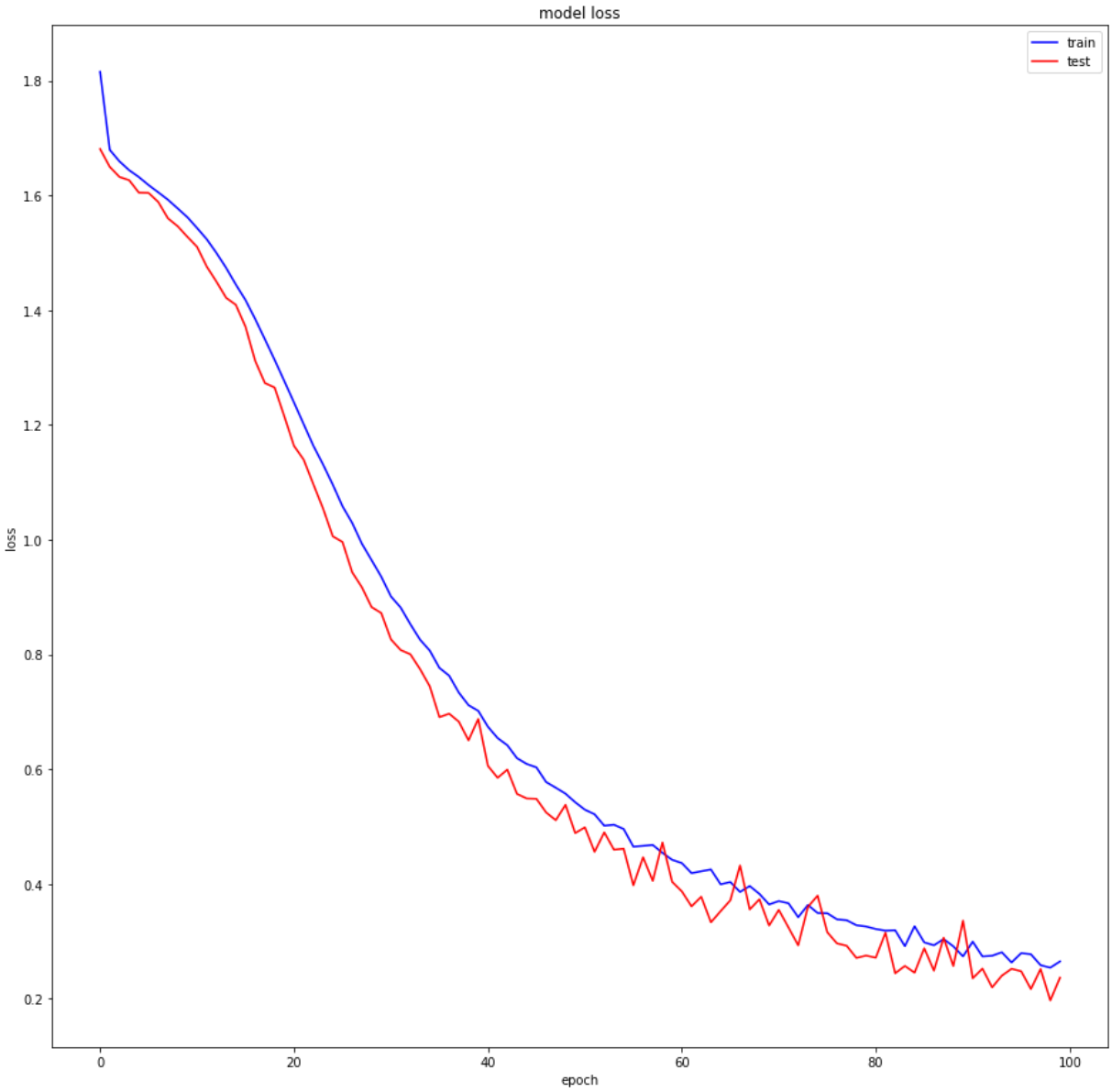

Рисунок 10. График функции ошибки.

В результате тренировки данной нейронной сети были получены следующие результаты:

- значение функции ошибки = 2.4
- значение метрики “Accuracy” = 0.92
- значение метрики “Precision” = 0.90
- значение метрики “Recall” = 0.92

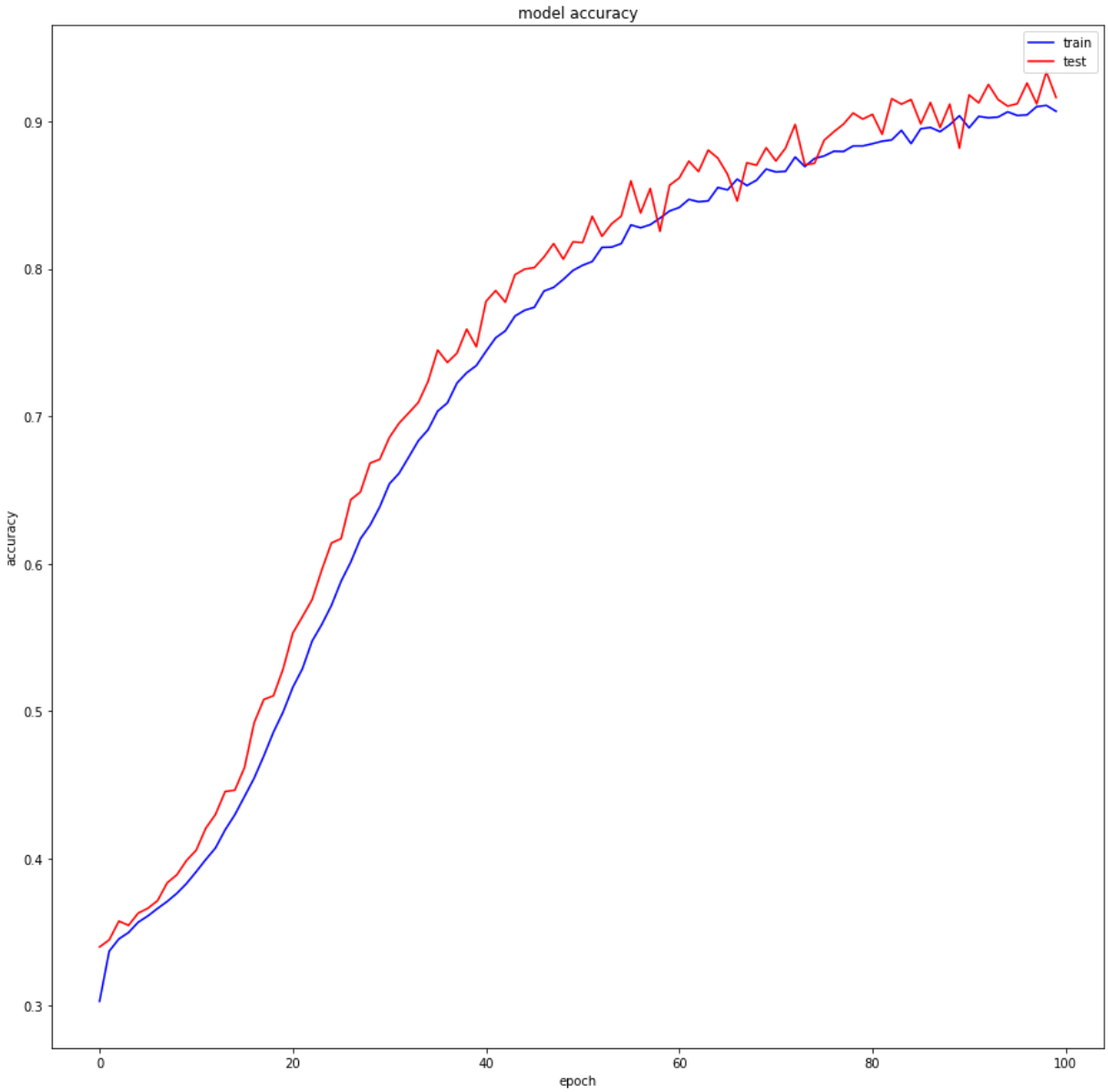

Рисунок 11. График метрики “Accuracy”.

Сравнивая данный результаты с полученными ранее, можно сказать, что использование шкалы оценок от 1 до 10 не сильно ухудшило ситуации, то есть созданная система может работать и м подобными данными.

Заключение

В рамках данной работы были изучены существующие и давно применяющиеся методы, а также появляющиеся с развитием машинного обучения концепции по реализации рекомендательных систем.

В рамках практического эксперимента были найдены необходимые для рекомендательной системы фильмов данные. Далее, были выбраны и изучены способы их обработки, а именно обработка естественного языка. После, для найденных данных о фильмах, пользователях и их оценках была реализована изученная обработка.

Затем, основываясь на изученных публикациях, была разработана архитектура нейронной сети для нейронной коллаборативной фильтрации. Описывая подробнее, был осуществлен подбор параметров сети, например количество нейронов на уровне, а также разработаны входные слои с учетом изменений в данных в связи с описанными в работе их преобразованиями, связанными с обработкой естественного языка.

Далее, полученная модель была реализована и обучена. После оценки качества получившейся обученной нейронной сети, была также произведено ее сравнение с уже существующей не преобразованной реализацией НКФ. В результате проделанной работы, можно утверждать, что получен прообраз рекомендательной системы фильмов, не уступающий существующим аналогам. Более того, привнесенные изменения его улучшили.

Затем, в модели были произведены дополнительные изменения, связанное со шкалой оценок фильмов. А именно, для нейронной сети были заново подобраны параметры, а также изменен выходной слой. После этого, новая модель, по аналогии с первой, была реализована, обучена, а ее качество оценено метриками. Показатели измененной полученной рекомендательной системы можно считать успешными, а эксперимент удачным.

Подводя итоги, можно заключить, что была проделана работа по изучению задачи о создании рекомендательных систем и об осуществлении рекомендаций в целом. Также были изучены как давно существующие подходы к созданию рекомендательных систем, так и новые исследования в этой области.

В рамках данной работы были изучены теоретические аспекты выбранного подхода для создания рекомендательной системы. Помимо этого, была предложена своя вариация обработки данных для реализуемой системы. Это было сделано на основе изученной обработке естественного языка.

Также, большую часть данной работы составляет создания своего варианта рекомендательной системы: выбранный подход и все модификации были реализованы и протестированы, а полученные результаты свидетельствуют об ожидаемом успехи при применении в реальных условиях созданного варианта нейронной коллаборативной фильтрации и её модификаций.

Список литературы

1. Jannach, D., M. Zanker, A. Felfernig and G. Friedrich, 2010. Recommender Systems: An Introduction. Cambridge: Cambridge University Press.

2. Su, X. and T.M. Khoshgoftaar, 2009. A survey of collaborative filtering techniques. Advances in Artificial Intelligence.

3. He, X., L. Liao, H. Zhang, L. Nie, X. Hu and T.S Chua, 2017. Neural Collaborative Filtering. Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee.

4. Гольдберг Й., Нейросетевые методы в обработке естественного языка. ДМК-Пресс, 2019 г., 282 с.

5. Cheng, H.T., J. Harmsen, L. Koc, T. Shaked, T. Chandra and others, 2016. Wide & deep learning for recommender systems. Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, ACM, pp: 7-10.

6. Covington, P., J. Covington and E. Sargin, 2016. Deep Neural Networks for YouTube Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems, ACM, pp: 191–198.

7. Zhang, F., N.J. Yuan, D. Lian, X. Xie and W.Y. Ma, 2016. Collaborative Knowledge Base Embedding for Recommender Systems. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery, KDD, pp: 353-362.

8. Wang, H., N. Wang and D.Y. Wang, 2015. Collaborative Deep Learning for Recommender Systems. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM.

9. Хайкин С., Нейронные сети. Полный курс. 2 изд. Вильямс, 2018. 1104 с.

10. Le, Q. and T. Mikolov, 2014. Distributed Representations of Sentences and Documents. Proceedings of the 31st International Conference on Machine Learning, PMLR, pp: 1188-1196.

11. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика. Москва, 2002 г., 382 с.

12. Паттерсон Дж., Гибсон А. Глубокое обучение с точки зрения практика. ДМК Пресс, 2018 г., 418 с.

13. Глебов Н. И., Ю.А. Кочетов, А. В. Плясунов. Методы оптимизации. Новосибирск: НГУ, 2000. 105с.

14. Goodfellow, I., Y. Bengio and A. Courville, 2016. Deep Learning. MIT Press.

15. Николенко С., Кадурин Е., Архангельская А., Глубокое обучение. Погружение в мир нейронных сетей. Питер, 2018 г., 480 с.

16. Theoretical Analysis of NDCG Type Ranking Measures. [Электронный ресурс]: URL: https://arxiv.org/pdf/1304.6480.pdf (дата обращения: 05.04.2021)

17. MovieLens // GroupLens [Электронный ресурс]: URL: https://grouplens.org/datasets/movielens/ (дата обращения: 13.10.20)

18. Шолле Ф., Глубокое обучение на Python. Питер, 2018 г., 400 с.

19. The Movies Dataset // Kaggle [Электронный ресурс]: URL: https://www.kaggle.com/rounakbanik/the-movies-dataset (дата обращения: 13.10.20)

20. Джулли А, Пал С. Библиотека Keras - инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и TensorFlow/ пер. с англ. М.: ДМК Пресс, 2018. - 294 с.

21. Gensim // Doc2vec paragraph embeddings [Электронный ресурс]: URL: https://radimrehurek.com/gensim/models/doc2vec.html (дата обращения: 17.11.20)

22. Keras // The Python deep learning API [Электронный ресурс]: URL: https://keras.io/ (дата обращения: 21.03.21)

23. Recommenders // Neural Collaborative Filtering on MovieLens dataset [Электронный ресурс]: URL: https://github.com/microsoft/recommenders (дата обращения: 10.04.21)