Санкт-Петербургский Государственный Университет

Математико-механический факультет

Кафедра системного программирования

Лебедев Дмитрий Юрьевич

«Агрегирование из различных источников рекомендаций для аудио контента»

Выпускная работа бакалавра

Допущен к защите

Зав. кафедрой:

д.ф.-м.н., проф. А.Н. Терехов

Научный руководитель:

к.ф.-м.н., Д.Ю. Бугайченко

Рецензент:

к.ф.-м.н., доцент И.П. Соловьев

Санкт-Петербург

2012

Saint-Petersburg State University

Mathematics and Mechanics Faculty

Software Engineering Department

Dmitry Lebedev

Aggregation from different sources of recommendations for audio content

Bachelor's thesis

"Approved by"

Head of Department

Professor A. N. Terekhov

Scientific advisor

PhD, Dmitry Bugaychenko

Reviewer

PhD, associate professor Igor Soloviev

Saint-Petersburg

2012

Оглавление

Введение

В последнее время, активный рост объема информации оставляет пользователя наедине с проблемой выбора. Объем данных становится настолько велик, что пользователь не в силах сам определить среди массы объектов те, в которых он был бы заинтересован. Эту проблему призваны решить рекомендательные системы, которые, учитывая предпочтения и прошлый выбор пользователя, рекомендуют объекты, которые могут его заинтересовать.

Данная работа сфокусирована на рекомендательных системах для аудиозаписей. В рамках этой работы исследуется возможность предоставлять не просто обезличенные рекомендации, а учитывающие социальные связи для более точного результата.

Область рекомендательных систем активно развивается, так как рекомендательные системы представляют немалый коммерческий интерес для компаний, владеющих ресурсами, продающими мультимедиа (книги, фильмы, аудиозаписи, и т.д.), и имеющих данные об интересах и совершенных покупках покупателей. Но главная проблема, заключающаяся в точном предсказании предпочтений пользователя, до сих пор является открытой. Существующие решения предсказывают возможную оценку объекта пользователем с немалой погрешностью. Некоторые реализации предсказывают достаточно точную оценку, но могут плохо работать с большими объемами данных. Некоторые системы плохо решают проблему «нового пользователя» (ситуация, когда в системе появляется новый пользователь, о предпочтениях которого еще ничего не известно) и «нового объекта» (когда в систем появляется новый объект, о котором неизвестно, кому он может понравится). Некоторые системы требуют больших вычислительных ресурсов, так как количество пользователей и объектов в системе может достигать нескольких десятков миллионов. Таким образом получатся, что при добавлении новых объектов или пользователей в систему, рост, количества данных происходит очень быстро. Соответственно,

целью разработок в данной области является улучшение точности, масштабируемости и производительности систем.

Эта работа нацелена на создание и анализ гибридной системы рекомендаций (агрегирования нескольких существующих подходов к рекомендациям), использующей преимущества и достоинства этих подходов. В работе рассматривается агрегирование данных о схожести пользователей, полученных с помощью рекомендательной системы, основанной на нейронной сети с данными о схожести пользователей, полученными с помощью графа социальных связей пользователей.

Системы рекомендаций

Необходимость в системах рекомендаций возникает в тех областях, где общее количество данных настолько велико, что пользователь не может полностью изучить данные и выбрать объекты для использования. Примерами таких областей могут служить книжная и музыкальная индустрии, с большим объемом книг и музыкальный записей соответственно. Пользователи не хотят, и не могут тратить время на изучения множества объектов, например, прослушивания множества аудиозаписей. Пользователи хотят, чтобы им предлагали те объекты, которые могут их заинтересовать с большой вероятностью. Для этой цели системы рекомендаций используют все возможные источники информации о пользователе в том числе: прошедшую активность пользователя, связи с другими пользователями и др.

Особенно, такие системы интересны компаниям, предоставляющим некоторый контент через социальные сети, так как они заинтересованы в том, чтобы пользователь был удовлетворен своим выбором, и потреблял как можно больше контента. Так, например, такие компании как Amazon[1], Apple[2] и многие другие в своих интеренет-магазинах используют системы рекомендаций товаров, на основе данных о прошлой активности пользователя. В 2006 году компания NetFlix[4], предоставляющая подписку на онлайн просмотр фильмов, проводила соревнование между командами, разрабатывающими системы рекомендаций. Команда, предоставившая систему, которая была бы на 10% более точной, чем существующая в сервисе на тот момент, становилась победителем и получала главный приз 1 000 000 долларов. Условием получения для приза являлось предоставление исходного кода алгоритма, который предсказывает рейтинг на наборе тестовых данных на 10% процентов точнее, чем текущая система рекомендаций NetFlix. Участвовало в этом соревновании множество команд, однако, первыми предоставившими систему, превосходящую на тот момент систему рекомендаций NetFlix, стали участники

из команды «BellKor’s Pragmatic Chaos». Они достигли улучшения на 10,09 % и получили главный приз. Соревнования такого рода дают большой толчок в развитии этой области, так как привлекают внимание ученых со всего мира.

Для чего нужны системы рекомендаций

Увеличение продаж. Пожалуй, самая важная функция коммерческих систем рекомендаций. Например, чтобы продать дополнительные вещи, которые чаще всего покупаются вместе с теми, что продаются без всяких рекомендаций. Эта цель достигается, так как рекомендуемые вещи, как правило, соответствуют потребностям и желаниям пользователя. Например, если человек покупает телефон в интернет-магазине, то ему можно порекомендовать также купить аксессуары к этому телефону, которые отдельно покупают редко. Некоммерческие приложения преследуют схожие цели, даже если пользователь не платит за выбранные объекты. Например, новостной сайт нацелен на то, чтобы пользователь читал как можно больше новостей на сайте, и старается предложить ему новости, связанные с текущими.

Продажа более разнообразных объектов. Другая важная функция систем рекомендаций - предоставить пользователю выбор среди объектов, которые было бы тяжело найти без точных рекомендаций. Например, в системе рекомендаций подобной Netflix, поставщик услуг заинтересован в аренде всех DVD из каталога, а не только самых популярных. Это было бы трудно воплотить в жизнь без системы рекомендаций, так как поставщик услуг не может рисковать широко рекламировать фильмы, которые вероятно не придутся по вкусу большей части пользователей. Гораздо лучшим решением будет реклама фильмов, ориентированная на каждого конкретного пользователя. Таким образом, система рекомендаций предлагает непопулярные фильмы тем пользователям, которым они могут понравится.

Увеличение удовлетворенности работой с системой. Правильно спроектированные системы рекомендаций могут улучшить восприятие пользователем приложения или сайта, где используется система рекомендаций. Если взаимодействие с системой удобное, и она предоставляет интересные и подходящие рекомендации, то пользователь будет удовлетворен работой с ней. Комбинирование эффективных и точных рекомендаций с удобным интерфейсом увеличит ценность системы для пользователя, и это в свою очередь увеличит время использования системы.

Увеличение лояльности пользователей. Пользователь будет лоялен к системе, которая распознает его вкусы и работает с ним с как известным посетителем. Это является характерной особенностью систем рекомендаций, так вычисление рекомендаций производится на основе информации, полученной из предыдущей деятельности пользователя, например, пользовательских рейтингов объектов. Следовательно, чем дольше пользователь взаимодействует с системой, тем более качественней становится его модель предпочтений.

Улучшение понимания пользовательских потребностей и желаний. Еще одна важная функция систем рекомендаций, которая может быть использована другими приложениями - создание описания пользовательских предпочтений, будь то собранные в явном виде или предсказанные системой. Поставщик услуг может решить использовать эти знания для других цели, таких как улучшение управления хранением или производством.

Определения

Системы рекомендаций обрабатывают и активно собирают информацию различного рода для того чтобы построить рекомендации. В общем случае - это объекты, пользователи и взаимоотношения между ними[6].

Объекты. Объекты характеризуются своей структурой и значимостью. Значимость может быть положительной, если объект полезен пользователю, или отрицательной, если объект неподходящий для пользователя, и он совершил неправильное решение, выбрав этот объект. Структура объектов зависит от предметной области, в которой реализована система рекомендаций. Например в системе рекомендаций фильмов объекты имеют такие атрибуты как название, год выпуска, жанр, режиссер, список актеров и др.

Пользователи. Пользователи предоставляют информацию для систем рекомендаций в явном и неявном виде. Например, для систем коллаборативной фильтрации, система рекомендаций получает данные пользователя о рейтингах объектов. В социально-ориентированных системах рекомендаций пользователь предоставляет данные о себе, такие как пол, возраст, профессия, образование, страна и город проживания и др.

Взаимоотношения. Как правило, взаимоотношения между пользователями и объектами выражаются в виде рейтингов, которые могут быть различного типа.

- Числовые рейтинги: от 1 до 5, как, например, для аудиозаписей в iTunes.
- Упорядоченные рейтинги: такие как “полностью согласен”, “согласен”, “нейтрален”, “не согласен”, “полностью не согласен”, в ситуации, когда у пользователя спрашивают мнение о высказывании.
- Двоичные рейтинги: предусматривает, что пользователь говорит про объект, что он либо “хороший”, либо “плохой”.
- Абсолютные рейтинги: предоставляющие информацию о том, что пользователь просмотрел объект. Как правило, при предусматривается подсчет количества просмотров.

Технологии и подходы

Существует несколько довольно популярных подходов к решению задачи предсказания предпочтений. Наиболее популярны из них следующие:

- Content-based (Ориентированный на контент)
- Collaborative filtering (Коллаборативная фильтрация)

Ориентированный на контент подход основан на том, что рекомендации рассчитываются исходя из информации, имеющейся о пользователе и о том, что ему нравится. Для этого, рекомендательная система ищет объекты похожи на те, что понравились пользователю в прошлом. В случае аудиозаписей рекомендательная система сравнивает их по таким параметрам как исполнитель, жанр, год и др. что хранится в так называемых тэгах.

Коллаборативная фильтрация является одним из наиболее развивающихся направлений, и основывается на идее поиска пользователей со схожими предпочтениями, и рекомендациях объектов интересных людям, похожим на пользователя. Выделяют два основных подхода в этом направлении:

- Memory-based
- Model-based

Memory-based подход основан на поиске схожести между пользователями или объектами. В свою очередь, этот подход подразделятся на два направления

- Item based (Объектно-ориентрованные)
- User based (Ориентированные на пользователя)

Целью обоих направлений является выделение схожих объектов в группы. Схожесть объектов, в первом, случае аудиозаписей, во втором пользователей, определятся с помощью матрицы рейтингов. Такой подход хорош тем, что при должной агрегации имеющихся данных получается неплохой предсказывающий механизм, однако с другой стороны этот подход достаточно трудно масштабировать, так как при большом объеме данных матрица рейтингов разрастается до огромных размеров.

Рассмотрим простой пример, где мы имеем данные о прослушиваемых пользователями треками без определения рейтингов, которые составляют следующую матрицу предпочтений:

Треки / Пользователи	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
U1	+		+			+				+
U2		+			+			+		+
U3	+				+		+		+	
U4			+			+		+		+
U5	+	+		+				+		
U6	+			+			+		+	
U7		+		+		+			+	

В этой таблице отражены предпочтения аудиозаписей пользователей следующим образом:

- Ячейки, в которых стоит «+», свидетельствуют о том, что пользователь слушает данную аудиозапись

- Пустые ячейки свидетельствуют о том, что пользователь не слушает данную аудиозапись

В первом направлении, необходимо определить похожесть аудиозаписей, что выполняется следующим образом. Считаем, что два трека похожи, если их слушает минимум n пользователей, где n – задаваемая граница похожести аудиозаписей. Для примера возьмем n = 2 и определим аудиозаписи, похожие на T1. В данном случае треками похожими на T1 окажутся T4 и T7. Рассмотрим пару треков T1 и T4. В нашем случае мы можем наблюдать, что оба трека слушают пользователи U5 и U6, а если смотреть по отдельности, то T1 слушает также пользователь U1 и U3, а T4 слушает пользователь U7. Отсюда можно заключить следующие рекомендации:

- Пользователю U1 и U3 мы можем рекомендовать послушать T4
- Пользователю U7 мы можем рекомендовать послушать T1

Рассмотрим второе направление. Определим похожесть пользователей по схожести их вкусов, следующим образом: считаем, что два пользователя похожи, если они слушают минимум n одинаковых треков, где m – задаваемая граница похожести пользователей. Рассмотрим случай с m = 2 и определим пользователей похожих на U1. В данном случае это пользователь U4. Рассмотрим пару пользователей U1 и U4. Оба они слушают аудиозаписи T3, T6, и T10. Пользователь U1 слушает также T1, а пользователь U4 слушает T8. Поэтому мы можем привести следующие рекомендации:

- Пользователю U1 мы можем рекомендовать послушать T8
- Пользователю U4 мы можем рекомендовать послушать T1

Model-based подход состоит в том, что имея данные о пользователях и треках, мы создаем некую модель, предварительно обучив на имеющемся наборе данных, и используем ее для предсказания рейтингов. С одной стороны, такой подход менее затратный в плане производительности и скорости при

работе, однако, с другой стороны подходящую модель достаточно тяжело разработать, и процесс обучения, как правило, занимает очень большое время.

Социальная фильтрация

Целью социально-ориентированного подхода является выделение социальных признаков, которые помогут системе рекомендаций предоставлять более точные предсказывания, по сравнению со стандартными подходами. Например в [5] было установлено, что схожесть пользователей в группе является характеристикой, коррелирующей с социальными взаимоотношениями в группе.

Социальная фильтрация, во многом напоминает коллаборативную фильтрацию, но в отличие от нее также использует информацию о социальных связях между пользователями. В подходах коллаборативной фильтрации наиболее используемыми являются два метода расчета схожести пользователей – расчет корреляции между рейтингами пользователей и расчет косинуса угла между векторами рейтингов. В первом случае формула примет вид:

$$w_{u,v} = \frac{\sum_{i=1}^{N}(r_{u,i} - \overline{r_u})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i=1}^{N}(r_{u,i} - \overline{r_u})^2}\sqrt{\sum_{i=1}^{N}(r_{v,i} - \overline{r_v})^2}} \quad (1)$$

Где $w_{u,v}$ – схожесть пользователей u и v, $r_{v,i}$ – рейтинг пользователя v для аудиозаписи i, а $\overline{r_v}$ - средний рейтинг по всем аудиозаписям пользователя v. В случае если схожесть определяется с помощью косинуса угла между векторами рейтингов, то формула примет следующий вид:

$$w_{u,v} = \cos(R_u, R_v) = \frac{R_u \bullet R_v}{\| R_u \| \| R_v \|} \quad (2)$$

Где R_u – вектор рейтингов пользователя u, и имеет следующий вид:

$R_u=(r_{u,1}, r_{u,2}, \dots, r_{u,N})$

После того как схожесть пользователей определена, мы будем рассчитывать предсказывания рейтингов по следующей формуле:

$$r_{u,i} = \overline{r_u} + \frac{\sum_{v \in N(u)} w_{u,v}(r_{v,i} - \overline{r_v})}{\sum_{v \in N(u)} |w_{u,v}|} \quad (3)$$

В нашем случае, так мы располагаем данными о социальных связях, мы будет рассчитывать схожесть пользователей по следующей формуле:

$$W_{u,v} = \alpha * connection(u,v) + \beta * w_{tracks}(u,v) \quad (4)$$

Где connection(u, v) = 1, если между пользователем u и v есть дружественная связь, 0 иначе. А $w_{u,v}$(tracks) – схожесть пользователей u и v по прослушиваемым аудиозаписям, которая вычисляется следующим образом:

$$w_{tracks}(u,v) = \cos(tracks_u, tracks_v) \quad (5)$$

Что имеет смысл схожести набора аудиозаписей двух пользователей.

α и β являются коэффициентами значимости каждой из метрик схожести и $\alpha + \beta = 1$. В нашем случае $\alpha = \beta = 0,5$.

Оценивание рекомендаций

Для оценивания точности рекомендаций используются метрики точности (precision) и охвата (recall) и F-measure.[8] Precision – качественная мера предоставляемых рекомендаций, которая вычисляется как отношение количества правильно предсказанных рейтингов, к общему количеству предсказанных рейтингов. N_{rel} – количество рейтингов предсказанных правильно, N_{ret} – общее количество предсказанных рейтингов.

$$\text{Precision} = \frac{N_{rel}}{N_{ret}} \quad (6)$$

Recall – количественная мера предоставляемых рекомендаций, которая определяется вероятностью того, что правильно предсказанный рейтинг будет выбран. N_{rel} – количество рейтингов предсказанных правильно, N_{tot_rel} – общее количество правильных рейтингов.

$$\text{Recall} = \frac{N_{rel}}{N_{tot\_rel}} \quad (7)$$

F-measure – мера, представляющая собой среднее гармоническое от Precision и Recall и высчитывается следующим образом:

$$F\text{-}measure = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (8)$$

Также для оценки точности предсказаний используется метрика называемая half-life utility. Идея заключается в том, что пользователь, при получении списка предсказанных рейтингов, редко смотрит объекты, которые находятся в конце списка. Метрика half-life utility вычисляет полезность списка предоставленных рейтингов. Полезность определяется как разность между предсказанным рейтингом и рейтингом по умолчанию. Вероятность того, что пользователь просмотрит каждый предложенный объект описывается убывающей экспоненциальной функцией, где сила убывания определяется параметром half-life.

$$R_a = \sum_j \frac{\max(r_{a,j} - d, 0)}{2^{(j-1)/(\alpha-1)}} \quad (9)$$

R_a –полезность предсказаний для пользователя a, $r_{a,j}$ –предсказанный рейтинг объекта j из списка рекомендаций для пользователя a, d – рейтинг по умолчанию, alpha – параметр half-life, который определяется как позиция объекта из списка рекомендаций, который понравится пользователю с вероятностью 0.5. Общая полезность предоставления рекомендаций определяется следующим образом:

$$R = 100 \frac{\sum_a R_a}{\sum_a R_a^{\max}} \quad (9)$$

Где R_a max – максимально возможная полезность, достигаемая в том случае, когда система расставит рейтинги для объектов в таком же порядке как это сделал бы пользователь.

Постановка задачи

- Реализовать алгоритм подсчета весов связей между пользователями на основе социальных связей и общих аудиозаписей
- Реализовать механизм агрегации предсказанных рейтингов
- Произвести оценку предоставленных рекомендаций

Реализация

Для реализации программы была использована библиотека проекта Apache Mahout[3]. Apache Mahout – проект, являющийся подпроектом Apache Hadoop, входящего в Apache Software Foundation, предоставляющим библиотеки для реализации масштабируемых алгоритмов машинного обучения, использующих большие объемы данных. Большинство алгоритмов в этой библиотеке реализованы на основе парадигмы MapReduce.

Данная программа расширяет существующую реализацию рекомендательной системы на основе нейронной сети[9].

Для работы программы необходимы следующие входные данные, на основе которых производится обработка:

- Users.csv – файл, содержащий данные о пользователе в виде следующих значений: идентификатор пользователя (id), имя пользователя (name).
- Friends.csv – файл, содержащий список пар пользователей, между которыми существует связь «друзья».
- Artists.csv – файл, содержащие данные об артистах в формате: идентификатор артиста (id), имя артиста (ArtistName).

Также, необходимые для работы файлы, получаемые после работы нейросетевого алгоритма, указываются в файле настроек program.properties. Он содержит следующие файлы:

- Файл, содержащий результат работы нейросети: группы, образованные пользователями (пользователи с идентификаторами групп, в которые они входят)
- Файл, содержащий рейтинги полученные на основе нейронной сети

- Файл, содержащий маску разбиения данных на данные для обучения и данные для проверки

Запуск программы производится с помощью командной строки. С помощью ключа –i и –o указывается расположение папок с входными и выходными данными соответственно. При указании ключа –verbose будет включен режим отчета с выводом сообщение о статусе работы.

Общий алгоритм работы рекомендательной сети:

- Трансформация информации об артистах, пользователях и рейтингах во внутренне представление (переиндексация)
- Трансформация информации о дружественных связях между пользователями (переиндексация)
- Разделение данных на обучающее и тестовое множество
- Расчет рейтингов на основе пользовательских данных о прослушивании артистов
- Уменьшение размерности данных (подготовка к обработке нейронной сетью)
- Выделение групп схожих пользователей с помощью нейросетевого алгоритма
- Расчет рейтингов пользователей и групп на основе полученной группировки
- Расчет рейтингов на основе социального подхода (с использованием полученных ранее рейтингов с помощью нейросетевого подхода)
- Формирование списка рекомендаций на основе пересчитанных рейтингов
- Обработка полученных рейтингов и расчет интересующих нас метрик для анализа полученных результатов

Согласно предыдущей реализации, программа может быть запущена в разной стадии обработки данных с помощью ключа **-p**. Для интеграции алгоритма социальных рекомендаций в список фаз была добавлена еще одна фаза “social”. Таким образом для запуска алгоритма социальных рекомендаций необходимо при запуске программы указать ***-p social.***

Программа использует два файла настроек (при их отсутствии используются стандартные настройки).

Параметры алгоритмов, используемых в программе содержатся в файле *algorithm.properties*. В файле присутствуют следующие параметры:

- *initial_data* – тип расчета рейтингов:
  - *listen_count* – количество прослушиваний;
  - *inversed* – инверсированное количество прослушиваний.
- *normalization* - тип обработки рейтингов:
  - *average* – усредненный рейтинг;
  - *zscore* – стандартный рейтинг.
- *learning_set_percent* – размер обучающего множества от общих данных(от 0 до 1)

Так же в программе присутствует возможность настроить вывод промежуточных данных и результата. В файле *filenames.properties* можно задать имена для файлов, в которые будет производиться запись промежуточных данных и результата. Для реализации нашего алгоритма в этот файл были добавлены следующие настройки:

- friends_list - для настройки имени файла со списком друзей
- social_ratings - для настройки имени файла с пересчитанными рейтингами (на основе нашего алгоритма)

- social_top_list_ratings - для настройки имени файла со списком рекомендаций на основе социального подхода (список рейтингов, соответствующих каждому из мест в топ-листе рекомендаций)
- social_top_list_ratings - для настройки имени файла со списком рекомендаций на основе социального подхода (идентификаторы артистов, соответствующих каждому из мест в топ-листе рекомендаций)

Реализация алгоритма

Входные данные хранятся в экземпляре класса SocialInputStorage, к которому имеет доступ экземпляр класса SocialAlgorithm. Этот класс-хранилище имеет возможность считывать данные с диска.

Исходные данные хранятся в экземпляре класса SocialOutputStorage, который обладает возможностью сохранять данные на диск.

Сам алгоритм социальных рекомендаций реализован в классе SocialAlgorithm. Этот класс имеет следующие параметры:

- friendsWeight - модификатор веса рекомендации в случае, если пользователи состоят в друзьях
- neuroWeight - модификатор веса рекомендации в случае, если пользователи состоят в одной группе по результатам работы нейронной сети
- similarityWeight - модификатор веса рекомендации в случае, если пользователи похожи друг на друга
- similarityDistanceThreshold – порог расстояния между пользователями, определяемыми как похожие друг на друга

Эти параметры передаются из файла настроек параметров алгоритмов в необходимый экземпляр класса.

Шаги алгоритма:

- Расчет новых весов, обозначающих схожесть между пользователями, на основе параметров friendsWieght, similarityWeight, neuroWeigth в виде суммы.
    - Коэффициент friendsWeight рассчитанный по формуле влияет, если пользователи являются друзьями
    - Коэффициент neuroWeight влияет, если оба пользователя принадлежат одной и той же группе из списка групп, полученных после обработки данных нейронной сетью
    - Коэффициент similarityWeight, рассчитываемый по формуле (5), влияет, если расстояние между векторами рейтингов пользователей не превышает заданной границы similarityDistanceThreshold
- Расчет рейтингов
    - Окончательный рейтинг рассчитывается по формуле (3)

Ниже приведена диаграмма классов приложения:

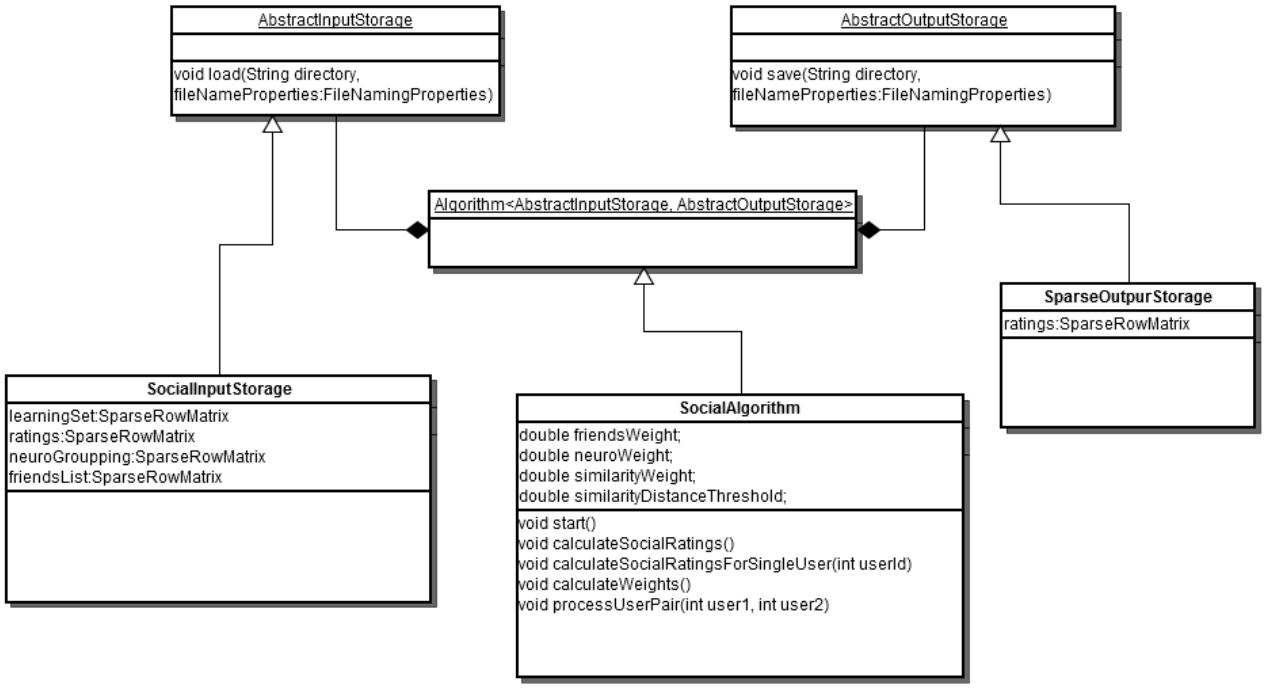

Рис. 1 Общая архитектура.

Эксперимент

В дополнение к эксперименту, проведенному в [9], был проведен эксперимент реализованного выше алгоритма на тех же данных, взятых из балы last.fm, которые представляют собой базу из 10 000 пользователей, 50 000 артистов, 500 000 рейтингов. Далее обработанные данные были оценены с помощью соответствующих метрик. Полученные результаты приведены ниже.

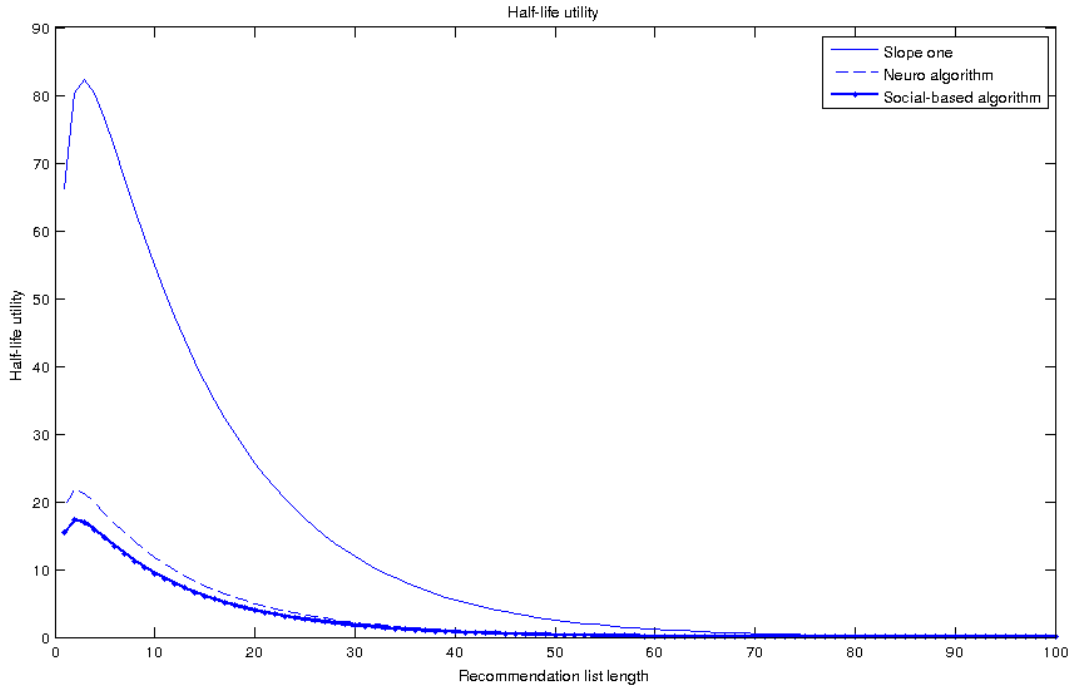

Рис. 2: Half-life utility для системы «Neuro», социально-ориентированного алгоритма и алгоритма Slope One. Расчет рейтингов – усреднение по количествам прослушиваний.

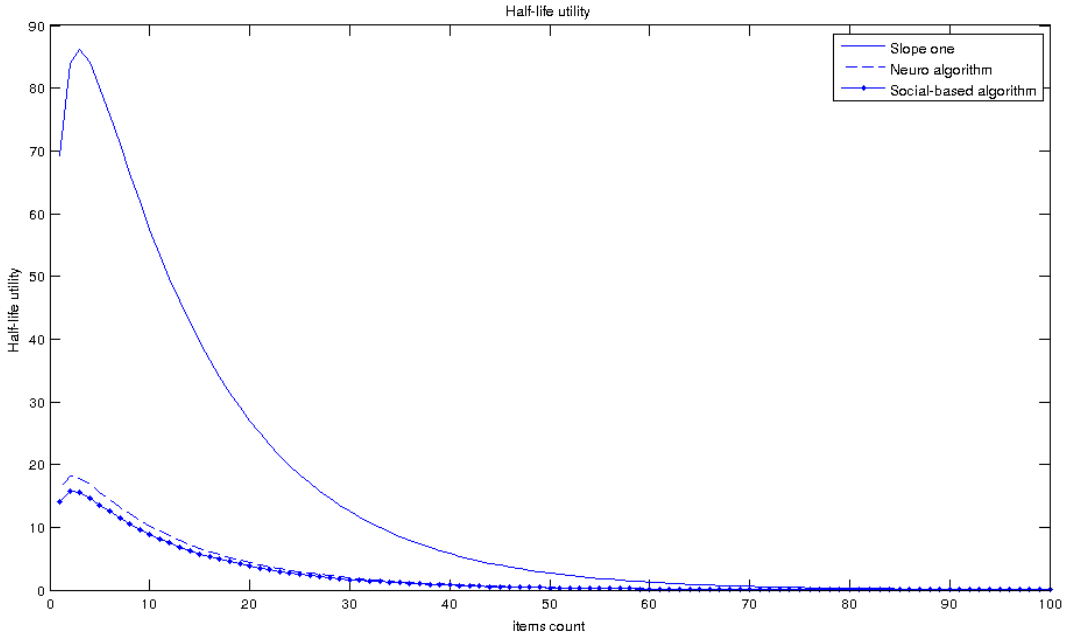

Рис. 3: Half-life utility для системы «Neuro», социально-ориентированного алгоритма и алгоритма Slope One. Расчет рейтингов – усреднение по инверсированным количествам прослушиваний.

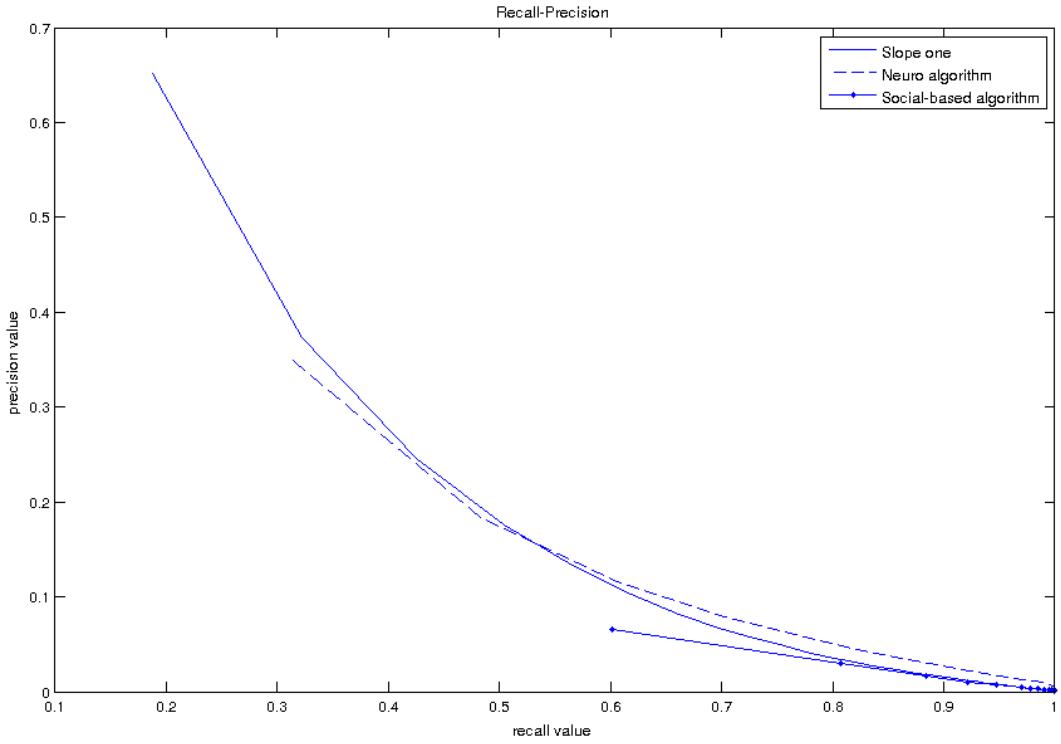

Рис. 4: График Recall-Precision для системы «Neuro», социально-ориентированного алгоритма и алгоритма Slope One. Расчет рейтингов – усреднение по инверсированным количествам прослушиваний.

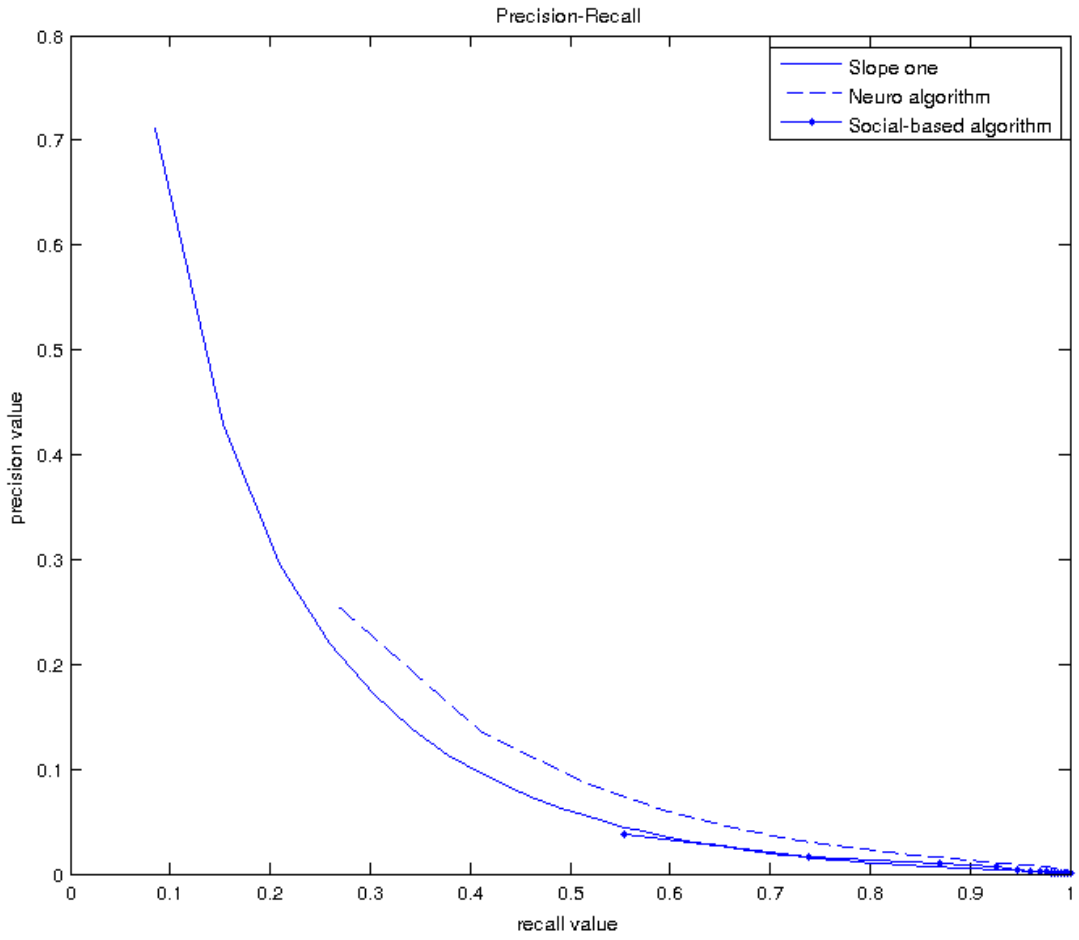

Рис. 5: График Recall-Precision для системы «Neuro», социально-ориентированного алгоритма и алгоритма Slope One. Расчет рейтингов – усреднение по количествам прослушиваний.

Вывод

В результате использования агрегирования результатов работы нейросетвого и социального подхода качество рекомендаций ухудшилось. Возможно, что это связано с недостаточно удачными выбранными коэффициентами для эксперимента, либо с большой разреженностью данных (в исходном наборе данных присутствует не очень много социальных связей).

Как видно из графиков, ухудшение рекомендаций произошло не настолько большое, чтобы полностью отказаться от этого метода. Возможно более тонкая настройка параметров, и выбор более подходящего набора данных для тестирования, позволит получить достойные результаты.

Результаты

- Реализован алгоритм подсчета весов связей между пользователями на основе социальных связей и общих аудиозаписей
- Реализован алгоритм агрегирования для предсказания рейтингов
- Произведена оценка предоставленных рекомендаций с помощью метрик Recall-Precision и Half-life utility.

Литература

1. Интернет магазин Amazon (15.05.12): http://www.amazon.com/.
2. Apple iTunes (15.05.12): http://www.apple.com/itunes/.
3. Apache Mahout (15.05.12): http://mahout.apache.org/
4. NetFlix (15.05.12): http://www.netflix.com/.
5. Georg Groh, Christian Ehmig: Recommendations in taste related domains collaborative filtering vs social filtering, 2007.
6. Gilles Louppe: Collaborative filtering. Scalable approaches using restricted Boltzmann machines, 2010.
7. Ioannis Konstas, Joemon M Jose, Vassilios Stathopoulos: On social networks and collaborative recommendation, 2009.
8. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl: Evaluating collaborative filtering recommender systems, 2004.
9. А.С. Солодкая. Предсказание предпочтений пользователя с использованием нейросетевого подхода, 2012.