Коллаборативная фильтрация на MapReduce

Григорьев Илья, 317 группа

15 сентября 2021 г.

Описание каждой стадии выполнения программы и каждой map-reduce задачи

Первая map-reduce задача

Маппер выполняет разделение на пары ключ-значение, где в качестве ключа – user_id, а в качестве значения – пара film_id, rating.

Редьюсер агрегирует по user_id и собирает пары film_id, rating в список для каждого user_id.

Вторая map-reduce задача

Маппер составляет пары фильмов, которые посмотрел каждый пользователь. В качестве ключа – пара (film_id1, film_id2), в качестве значения – user_id, который посмотрел оба этих фильма; средний рейтинг этого пользователя и две его оценки для данной пары фильмов.

Редьюсер агрегирует по film_id. Для каждого film_id строится список из фильмов, которые были с ним в паре. В свою очередь, для каждого фильма из списка вычисляется их similarity и хранится список пользователей, которые посмотрели оба этих фильма. Вместе с каждым user_id хранится его рейтинг фильму из ключа.

Третья map-reduce задача

Маппер выполняет разделение на пары ключ-значение, где в качестве ключа – user_id, а в качестве значения – список фильмов, которые пользователь еще не оценил (только фильмы, для которых можно посчитать similarity с тем фильмом, который был в ключе на входе данного маппера). Вместе с каждым неоцененным фильмом хранятся similarity (с тем фильмом, который был в ключе) и рейтинг данного пользователя (для того фильма, который был в ключе).

Редьюсер агрегирует по user_id и для каждого пользователя составляет список из неоцененных фильмов вместе с предсказаниями рейтинга данного пользователя для каждого неоцененного фильма.

Четвертая map-only задача

Маппер для каждого пользователя сортирует пары (film_id, predicted_rating) по невозрастанию предсказанного рейтинга и оставляет топ 100 фильмов.

Post-processor

Заменяет film_id на название фильма, сортирует фильмы в лексико-графическом порядке для фильмов с одинаковым рейтингом, форматирует вывод, сортирует строки по возрастанию user_id.

Сложность по числу операций и по количеству памяти

Mapper1: сложность по числу операций – $O(U * I * \alpha)$, сложность по памяти – $O(1)$.

Reducer1: сложность по числу операций – $O(U * I * \alpha)$, сложность по памяти – $O(I * \alpha)$.

Mapper2: сложность по числу операций – $O(U * I^2 * \alpha^2)$, сложность по памяти – $O(I * \alpha)$.

Reducer2: сложность по числу операций – $O(I^2)$, сложность по памяти – $O(I * U * \alpha^2)$.

Mapper3: сложность по числу операций – $O(U * I^2 * \alpha^2)$, сложность по памяти – $O(I * U * (1 + \alpha^2 - \alpha))$.

Reducer3: сложность по числу операций – $O(U * I^2 * (1 - \alpha))$, сложность по памяти – $O(I * (1 - \alpha))$.

Mapper4: сложность по числу операций – $O(U * I * (1 - \alpha) * \log(I * (1 - \alpha)))$, сложность по памяти – $O(I * (1 - \alpha))$.

Данные рассчеты представленны для всех данных целиком. На первом шаге было 2 маппера и 16 редьюсеров, на втором шаге – 16 мапперов и 16 редьюсеров, на третьем шаге – 16 мапперов и 16 редьюсеров, на четвертом шаге – 16 мапперов и 1 редьюсер (которого не было).

Суммарное время работы программы

Суммарное время работы программы составило 15 минут.