Engineering Applications of Artificial Intelligence 75 (2018) 114–124

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A MapReduce implementation of posterior probability clustering and relevance models for recommendation

Daniel Valcarce [*], Javier Parapar, Álvaro Barreiro

Information Retrieval Lab, Computer Science Department, University of A Coruña, Campus de Elviña s/n, 15071 A Coruña, Spain

ARTICLE INFO

Keywords:
Recommender systems
Collaborative filtering
MapReduce
Relevance-based language models
Posterior probability clustering
Matrix factorisation

ABSTRACT

Relevance-Based Language Models are a formal probabilistic approach for explicitly introducing the concept of relevance in the Statistical Language Modelling framework. Recently, they have been determined as a very effective way of computing recommendations. When combining this new recommendation approach with Posterior Probabilistic Clustering for computing neighbourhoods, the item ranking is further improved, radically surpassing rating prediction recommendation techniques. Nevertheless, in the current landscape where the number of recommendation scenarios reaching the big data scale is increasing day after day, high figures of effectiveness are not enough. In this paper, we address one urging and common need of recommendation systems which is algorithm scalability. Particularly, we adapted those highly effective algorithms to the functional MapReduce paradigm, that has been previously proved as an adequate tool for enabling recommenders scalability. We evaluated the performance of our approach under realistic circumstances, showing a good scalability behaviour on the number of nodes in the MapReduce cluster. Additionally, as a result of being able to execute our algorithms distributively, we can show measures in a much bigger collection supporting the results presented on the seminal paper.

1. Introduction

Users are demanding more and more personalised and immediate items of information instead of browsing and explicitly querying the web. In this landscape, recommender systems have been growing in importance in parallel to the increasing amount of information since they offer tailored suggestions using the available data about the users and the items.

There exist multiple recommendation approaches which are commonly classified into three main categories: content-based, collaborative filtering and hybrid recommenders (Ricci et al., 2011). Content-based techniques use the properties of the items rated by the target user to provide personalised suggestions (Lops et al., 2011). Thus, they require rich metadata about the items in the system. In contrast, collaborative filtering methods solely exploit the historical interaction between users and items for generating recommendations (Desrosiers and Karypis, 2011). They do not require any data about the items which are considered as black boxes; however, it is necessary to have a community of users. Finally, hybrid recommendation approaches compute recommendations taking into account both types of evidence.

Usually, the main concern of the designer of a recommender system is to deliver useful recommendations, but, as the number of users and items increases, scalability becomes a crucial issue. Processing large amounts of data give the opportunity of satisfying the information needs of the users with better recommendations at the cost of a much heavier computation load. Particularly, collaborative filtering techniques are very effective in these situations where a lot of data is available. Nevertheless, several scalability issues may arise since recommendation algorithms are usually designed for single-node computing.

New paradigms have been developed (Dean and Ghemawat, 2004) to tackle problems involving large-scale datasets. These frameworks propose solutions to distribute the computation across multiples machines in a fault-tolerant manner. Since they employ multiple nodes, these paradigms should ensure that the programme execution will finalise correctly and efficiently in spite of machine failures. Additionally, they must be scalable, that is, the cluster performance should improve proportionally when more computing nodes are added.

Recent efforts have resulted in the development of distributed implementations of numerous recommendation algorithms (Das et al., 2007; Liu et al., 2010; Gemulla et al., 2011; Yu et al., 2012; Schelter et al., 2012, 2013; Chen et al., 2014; Valcarce et al., 2015a). In this paper, we intend to accomplish the same task for a recent recommendation technique based on Posterior Probability Clustering

[*] Corresponding author.
E-mail addresses: daniel.valcarce@udc.es (D. Valcarce), javierparapar@udc.es (J. Parapar), barreiro@udc.es (Á. Barreiro).

https://doi.org/10.1016/j.engappai.2018.08.006
Received 2 March 2017; Received in revised form 18 April 2018; Accepted 14 August 2018
Available online 5 September 2018

(PPC) and Relevance-Based Language Models (RM2). We refer to this algorithm as PPC+RM2 (Parapar et al., 2013). This recommendation technique, which has demonstrated superior accuracy than other state-of-the-art rating prediction algorithms, is divided into two phases. First, PPC, a matrix factorisation method, is used for clustering the user space. Second, an adaptation of Relevance-Based Language Models for Recommender Systems is utilised for generating tailored suggestions using the neighbourhoods calculated in the previous stage.

In this paper, we describe the development of a scalable version of this recommendation technique. Specifically, we employed MapReduce which is one of the most popular large-scale data processing frameworks, and it was proposed by Google (Dean and Ghemawat, 2004). In fact, the development of recommender systems using MapReduce has become a fertile area of interest as the growing body of literature shows (Das et al., 2007; Liu et al., 2010; Gemulla et al., 2011; Schelter et al., 2012, 2013; Chen et al., 2014). As we will explain in the following sections, the distribution of the computation of this recommendation algorithm was complicated and we needed to seek compromise solutions. We employed different join strategies depending on the properties of the data to optimise the efficiency. We also discovered that PPC generates very unbalanced clusters which demote the efficiency of the RM2-based recommendation severely. Therefore, we designed a cluster balancing algorithm that leverages the probabilistic interpretation of PPC. This strategy proved successful in our experiments regarding both efficiency and effectiveness.

We decided to employ the Netflix dataset in our large-scale experiments. The Netflix Prize, an open competition held by Netflix from 2006 to 2009, represented a major milestone in the field of Recommender Systems (Bennett and Lanning, 2007). This competition consisted in improving the rating prediction algorithm that Netflix was using in that time by 10%. Towards this end, the company released a massive dataset containing 100 million ratings from 480 thousand users to almost 18 thousand films. In this paper, we employ this collection for testing the scalability of the proposed implementation of the recommendation algorithm based on PPC and RM2. We discovered that PPC scalability is comparable to other related methods such as NMF (Liu et al., 2010). Moreover, we identified RM2 as a highly scalable technique. We obtained a nearly linear scalability with the number of computing nodes. Additionally, we also present the first accuracy figures of this recommendation technique for this large-scale dataset because the original paper only studied a single-node implementation on the MovieLens 100k and 1M datasets. Our proposal showed good ranking accuracy figures compared to the standard baselines.

In summary, the contributions of this paper are (1) the development of a MapReduce implementation of PPC (Ding et al., 2008) and the recommendation algorithm based on RM2 (Parapar et al., 2013), (2) a cluster balancing algorithm to distribute the users across the clusters computed by PPC in order to be able of scaling up the recommendation process, (3) a study of the efficiency and scalability of the proposed MapReduce implementation and (4) a comparison of the PPC+RM2 algorithm against several baselines using the Netflix Prize dataset, a large-scale collection.

2. Recommendation algorithm

Given a set of users U, the goal of a recommender system is to find elements, from the set of items I, that may be of interest to each particular user $u \in U$. Since we are following a collaborative filtering approach, we utilise the interaction between users and items. Thus, we denote the ratings from the user u to the item i by $r_{u,i}$. Also, I_u is defined as the set of items rated by user u. The output of the recommender is a list L_u^k which represents the ranking of k suggested items for the user u.

Next, we describe the recommendation process whereas its distributed implementation is detailed in Section 3. As it was noted, the recommendation method proposed in Parapar et al. (2013) is composed of two phases: the generation of the users' neighbourhoods using Posterior Probability clustering and the computation of recommendations with Relevance Modelling.

2.1. Posterior Probabilistic Clustering

Posterior Probabilistic Clustering (PPC) (Ding et al., 2008) is a technique that provides a posterior probability interpretation for Non-negative Matrix Factorisation (NMF) (Lee and Seung, 2000). For this reason, we will outline NMF first, and then we will describe PPC and its advantages.

NMF is a particular matrix factorisation method. Given a non-negative matrix $\vec{A} \in \mathbb{R}^{+m\times n}$ and a positive integer $k \in \mathbb{Z}^+$, NMF tries to find $\vec{W} \in \mathbb{R}^{+m\times k}$ and $\vec{H} \in \mathbb{R}^{+k\times n}$ such that $\vec{A} \approx WH$. In practice, $k \ll \min(m, n)$ in order to obtain a low-rank approximation of $\vec{A}$.

We can apply NMF to the recommendation task as a clustering algorithm where k establishes the number of clusters. Given a collection of n users and m items, let $\vec{A} = (\vec{A}_{i,j})$ be the item-to-user matrix where $\vec{A}_{i,j}$ is the score of the item i rated by the user j (i.e., $\vec{A}_{i,j} = r_{j,i}$). Once the solution (W^*, H^*) is obtained, each user j is assigned to the cluster c_l such that $l = \arg\max_z(\vec{H}^*_{z,j})$.

Since NMF is a minimisation problem, there exist a plethora of optimisation methods for computing such factorisation such as multiplicative rules, gradient descent approaches or alternating least squares (Lee and Seung, 2000; Berry et al., 2007). These methods are intended to minimise the factorisation error between the original matrix and the approximation. It is typical that these algorithms utilise the Euclidean distance or the generalised Kullback–Leibler divergence as similarity metrics. Nevertheless, it is important to emphasise that these algorithms offer a suboptimal solution: they find a local optimum because NMF is a non-convex problem. In spite of this fact, the suboptimal solution is usually suitable for data mining purposes (Berry et al., 2007).

One of the main disadvantages of NMF is that its solution is not unique, and neither is the cluster assignment (Berry et al., 2007; Ding et al., 2008). Consider that a solution given by the matrices $\vec{W}$ and $\vec{H}$ can generate infinite alternative solutions in the form of $\vec{W}\vec{Q}$ and $\vec{Q}^{-1}\vec{H}$ where $\vec{Q}$ is an invertible matrix such that $\vec{Q} \in \mathbb{R}^{+k\times k}$. The lack of uniqueness motivates the use of other variants, of the many ones that have been developed.

One variant that has been proposed in the context of document clustering is Posterior Probability Clustering (PPC). This method guarantees a unique solution and removes the uncertainty in the clustering assignment enforcing the posterior probability normalisation on $\vec{H}$ while minimising the Euclidean distance (Ding et al., 2008). This method considers the columns of $\vec{H}^*$ as the posterior probability that the target user j belongs to each cluster (i.e., $p(j|c_l) = \vec{H}^*_{l,j}$). PPC can be expressed as the following optimisation problem:

$$\min_{\vec{W}\geq 0,\, \vec{H}\geq 0} \|\vec{A} - \vec{W}\vec{H}\|_2^2, \quad s.t. \sum_{l=1}^{k} \vec{H}_{lj} = 1 \tag{1}$$

Ding et al. derived the following update rules (Ding et al., 2008), using Lagrange multipliers, to calculate $\vec{W}$ and $\vec{H}$:

$$\vec{H} \leftarrow \vec{H} \,.*\, \frac{\vec{W}^T\vec{A} + \mathrm{diag}(\vec{H}^T\vec{W}^T\vec{W}\vec{H})}{\vec{W}^T\vec{W}\vec{H} + \mathrm{diag}(\vec{H}^T\vec{W}^T\vec{A})} \tag{2}$$

$$\vec{W} \leftarrow \vec{W} \,.*\, \frac{\vec{A}\vec{H}^T}{\vec{W}\vec{H}\vec{H}^T} \tag{3}$$

where diag refers to the diagonal elements of the matrix it is referencing. Throughout this paper, we denote element-wise matrix multiplication and division with $.*$ and —, respectively. The MapReduce implementation of these rules is detailed in Section 3.2.

2.2. Relevance modelling

The second phase of the recommendation algorithm employs Relevance-based Language Models (frequently abbreviated as Relevance Models or RM) which are a state-of-the-art technique for improving retrieval performance in Information Retrieval (IR) (Lavrenko and Croft, 2001). In the text retrieval task, users express their information needs