AutoRec: Autoencoders Meet Collaborative Filtering

Suvash Sedhain†, Aditya Krishna Menon†, Scott Sanner†, Lexing Xie†
† NICTA, † Australian National University
suvash.sedhain@anu.edu.au, {aditya.menon, scott.sanner}@nicta.com.au,
lexing.xie@anu.edu.au

ABSTRACT
This paper proposes AutoRec, a novel autoencoder framework for collaborative filtering (CF). Empirically, AutoRec’s compact and efficiently trainable model outperforms state-of-the-art CF techniques (biased matrix factorization, RBM-CF and LLORMA) on the Movielens and Netflix datasets.

Categories and Subject Descriptors D.2.8 [Information Storage and Retrieval]Information Filtering

Keywords Recommender Systems; Collaborative Filtering; Autoencoders

1. INTRODUCTION
Collaborative filtering (CF) models aim to exploit information about users’ preferences for items (e.g., star ratings) to provide personalised recommendations. Owing to the Netflix challenge, a panoply of different CF models have been proposed, with popular choices being matrix factorisation [1, 2] and neighbourhood models [5]. This paper proposes AutoRec, a new CF model based on the autoencoder paradigm; our interest in this paradigm stems from the recent successes of (deep) neural network models for vision and speech tasks. We argue that AutoRec has representational and computational advantages over existing neural approaches to CF [4], and demonstrate empirically that it outperforms the current state-of-the-art methods.

2. THE AUTOREC MODEL
In rating-based collaborative filtering, we have m users, n items, and a partially observed user-item rating matrix \( R \in \mathbb{R}^{m \times n} \). Each user \( u \in U = \{1, \ldots, m\} \) can be represented by a partially observed vector \( r^{(u)} = (R_{u1}, \ldots, R_{un}) \in \mathbb{R}^n \). Similarly, each item \( i \in I = \{1, \ldots, n\} \) can be represented by a partially observed vector \( r^{(i)} = (R_{i1}, \ldots, R_{im}) \in \mathbb{R}^m \).

Our aim in this work is to design an item-based (user-based) autoencoder which can take as input each partially observed \( r^{(i)} \) \( r^{(u)} \), project it into a low-dimensional latent (hidden) space, and then reconstruct \( r^{(i)} \) \( r^{(u)} \) in the output space to predict missing ratings for purposes of recommendation.

Formally, given a set \( S \) of vectors in \( \mathbb{R}^d \), and some \( k \in \mathbb{N}_+ \), an autoencoder solves

\[
\min_\theta \sum_{r \in S} \| r - h(r; \theta) \|^2_2, \tag{1}
\]

where \( h(r; \theta) \) is the reconstruction of input \( r \in \mathbb{R}^d \),

\[
h(r; \theta) = f(W \cdot g(Vr + \mu) + b)
\]

for activation functions \( f(\cdot), g(\cdot) \). Here, \( \theta = \{W, V, \mu, b\} \) for transformations \( W \in \mathbb{R}^{d \times k}, V \in \mathbb{R}^{k \times d} \), and biases \( \mu \in \mathbb{R}^k, b \in \mathbb{R}^d \). This objective corresponds to an auto-associative neural network with a single, \( k \)-dimensional hidden layer. The parameters \( \theta \) are learned using backpropagation.

The item-based AutoRec model, shown in Figure 1, applies an autoencoder as per Equation 1 to the set of vectors \( \{r^{(i)}\}_{i=1}^m \), with two important changes. First, we account for the fact that each \( r^{(i)} \) is partially observed by only updating during backpropagation those weights that are associated with observed inputs, as is common in matrix factorisation and RBM approaches. Second, we regularise the learned parameters so as to prevent overfitting on the observed ratings. Formally, the objective function for the Item-based AutoRec (I-AutoRec) model is, for regularisation strength \( \lambda > 0 \),

\[
\min_\theta \sum_{i=1}^m \| r^{(i)} - h(r^{(i)}; \theta) \|^2_2 + \frac{\lambda}{2} (\|W\|^2_F + \|V\|^2_F), \tag{2}
\]

where \( \| \cdot \|^2_F \) means that we only consider the contribution of observed ratings. User-based AutoRec (U-AutoRec) is derived by working with \( \{r^{(u)}\}_{u=1}^m \). In total, I-AutoRec requires the estimation of \( 2nk + m + k \) parameters. Given learned parameters \( \theta \), I-AutoRec’s predicted rating for user \( u \) and item \( i \) is

\[
\hat{R}_{ui} = (h(r^{(i)}; \theta))_u \tag{3}
\]

Figure 1 illustrates the model, with shaded nodes corresponding to observed ratings, and solid connections corresponding to weights that are updated for the input \( r^{(i)} \).
AutoRec is distinct to existing CF approaches. Compared to the RBM-based CF model (RBM-CF) [4], there are several differences. First, RBM-CF proposes a generative, probabilistic model based on restricted Boltzmann machines, while AutoRec is a discriminative model based on autoencoders. Second, RBM-CF estimates parameters by maximising log likelihood, while AutoRec directly minimises RMSE, the canonical performance in rating prediction tasks. Third, training RBM-CF requires the use of contrastive divergence, whereas training AutoRec requires the comparatively faster gradient-based backpropagation. Finally, RBM-CF is only applicable for discrete ratings, and estimates a separate set of parameters for each rating value. For $r$ possible ratings, this implies $nkr$ or $(nkr)$ parameters for user- (item-) based RBM. AutoRec is agnostic to $r$ and hence requires fewer parameters. Fewer parameters enable AutoRec to have less memory footprint and less prone to overfitting. Compared to matrix factorisation (MF) approaches, which embed both users and items into a shared latent space, the item-based AutoRec model only embeds items into latent space. Further, while MF learns a linear latent representation, AutoRec can learn a nonlinear latent representation through activation function $g(\cdot)$.

3. EXPERIMENTAL EVALUATION

In this section, we evaluate and compare AutoRec with RBM-CF [4], Biased Matrix Factorisation [1] (BiasedMF), and Local Low-Rank Matrix Factorisation (LLORMA) [2] on the Movielens 1M, 10M and Netflix datasets. Following [2], we use a default rating of 3 for test users or items without training observations. We split the data into random 90%–10% train-test sets, and hold out 10% of the training observations. We split the data into random 90%–10% train-test sets, and hold out 10% of the training set for hyperparameter tuning. We repeat this splitting 5 times and report average RMSE.

We developed a deep version of I-AutoRec with three hidden layers of (500, 250, 500) units, each with a sigmoid activation. We used greedy pretraining and then fine-tuned by gradient descent. On Movielens 1M, RMSE reduces from 0.831 to 0.827 indicating potential for further improvement via deep AutoRec.

Do deep extensions of AutoRec help? We developed a deep version of I-AutoRec with three hidden layers of (500, 250, 500) units, each with a sigmoid activation. We used greedy pretraining and then fine-tuned by gradient descent. On Movielens 1M, RMSE reduces from 0.831 to 0.827 indicating potential for further improvement via deep AutoRec.

Acknowledgments: NICTA is funded by the Australian Government as represented by the Dept. of Communications and the ARC through the ICT Centre of Excellence program. This research was supported in part by ARC DP140002185.

References