big data and cognitive computing

Article

A Multi-Modality Deep Network for Cold-Start Recommendation

Mingxuan Sun *, Fei Li and Jian Zhang

Division of Computer Science and Engineering, Louisiana State University, Baton Rouge, LA 70803, USA; fli6@lsu.edu (F.L.); zhang@csc.lsu.edu (J.Z.)

* Correspondence: msun@csc.lsu.edu

Received: 1 February 2018; Accepted: 27 February 2018; Published: 5 March 2018

Abstract: Collaborative filtering (CF) approaches, which provide recommendations based on ratings or purchase history, perform well for users and items with sufficient interactions. However, CF approaches suffer from the cold-start problem for users and items with few ratings. Hybrid recommender systems that combine collaborative filtering and content-based approaches have been proved as an effective way to alleviate the cold-start issue. Integrating contents from multiple heterogeneous data sources such as reviews and product images is challenging for two reasons. Firstly, mapping contents in different modalities from the original feature space to a joint lower-dimensional space is difficult since they have intrinsically different characteristics and statistical properties, such as sparse texts and dense images. Secondly, most algorithms only use content features as the prior knowledge to improve the estimation of user and item profiles but the ratings do not directly provide feedback to guide feature extraction. To tackle these challenges, we propose a tightly-coupled deep network model for fusing heterogeneous modalities, to avoid tedious feature extraction in specific domains, and to enable two-way information propagation from both content and rating information. Experiments on large-scale Amazon product data in book and movie domains demonstrate the effectiveness of the proposed model for cold-start recommendation.

Keywords: recommender system; deep learning; multimodal learning

1. Introduction

Recommender systems have been important tools for many business applications with broad economic impact [1–4]. Successful systems span various platforms, including Netflix's movie recommendations, Amazon's book recommendations, and Pandora's music recommendations. Constructing more accurate and personalized recommendation algorithms can help users to find items that they truly like from thousands or millions of items, which is important for the benefit of both users and businesses.

Collaborative filtering (CF), one of the most popular approaches to recommender systems, can recommend top items favored by the like-mined based on a collection of user ratings or purchase history [1,5–8]. A famous example is Netflix, where a group of users rate a set of movies to indicate their preferences and the system recommends movies tailored to individuals based on rating patterns. The success of collaborative filtering models largely depends on sufficient interaction history, and hence the prediction accuracy dramatically degrades for users and items with fewer interactions, which is known as the cold-start problem. In fact, a large portion of users and items are "cold" in many applications. For example, in the Netflix movie rating dataset of 480 k users and 17 k items, most users only rate a few movies.

On the other hand, content-based approaches extract content features such as the demographic information of users and the textual descriptions of items and recommend items with similar contents, which can be used to alleviate the cold-start problem. A prominent content model extracting text features [9] has been successfully applied to news recommendations such as Yahoo!'s Today module, where lots of daily news emerge and there are little historical data per user. In fact, content-based models and collaborative filtering approaches complement each other, which motivates us to adopt hybrid systems integrating both contents and collaborative filtering in a unified way.

To construct a hybrid collaborative filtering model with contents more effectively, it is highly desirable to exploit correlation or complementary information from different resources and learn more expressive representations than single-view learning. For example, in Amazon's product recommender system, millions of items in women's clothing emerge everyday with both textual and visual descriptions. A customer likes a dress because of the visual cues such as color and style and non-visual cues such as material and fitting inferred from text descriptions. Fusing content features from different input modalities and mapping them to similar users can further improve recommendation accuracy.

How to efficiently integrate heterogeneous contents into collaborative filtering remains a challenge for two main reasons. First, in classic matrix-factorization based collaborative filtering methods, both users and items are mapped into a lower-dimensional Euclidean space, so that a rating is approximated as a similarity function between a user and an item. It is challenging to map item contents in different modalities from original feature space to a joint lower-dimensional space, since they have intrinsically different characteristics and statistical properties, such as sparse texts and dense images. Generally, feature mapping algorithms tailored to unimodal data cannot be directly applied to multimodal data. Secondly, most algorithms only use content features as the prior knowledge to improve the estimation of user and item profiles. However, the ratings do not directly provide feedback to guide feature extraction. A tightly coupled framework is therefore needed so that more effective user and items embeddings can be automatically learned from both content features and rating information.

In this paper, we propose to tackle the above challenges by learning effective feature representations from multi-modal data through a unified deep network, to avoid tedious feature extraction in specific domains, and to enable two-way information propagation from both content and rating information. Specifically, we integrate feature learning and rating prediction into a unified deep learning architecture, where item embeddings are learned through auto-encoders to extract item semantic representations from heterogeneous contents, and the correlation of items and user embeddings is used to predict ratings. The objective function is to minimize both feature reconstruction error and rating prediction error.

To summarize, our unified deep network model tends to couple the collaborative filtering models with content information more effectively. In particular, the content correlation between different modalities is captured by the shared embeddings, and the embeddings of users and items adapt to each other in a way that the prediction of ratings can be largely improved. Experimental results on large-scale benchmark recommendation datasets such as Amazon Movies and Books demonstrate that our proposed algorithm significantly outperforms existing methods.

2. Related Work

There are two major categories of recommendation algorithms: content-based filtering and collaborative filtering. The former measures a user's satisfaction with an item based on the user information and item features [10–12]. For example, item features include textural descriptions such as genres and synopses, and visual cues from posters. User features may include demographic information, location, activity context, and device capability. Collaborative filtering goes beyond content-based methods to correlate users and items based on the assumption that users prefer items favored by the like-minded [1,5,13–19].

One challenge of collaborative filtering is that the performance is largely affected by several factors such as the number of items, the number of users, and the density of observed ratings. Hybrid models have been proposed in [20–22] by incorporating content features such as item genres and information extracted from user social network as prior knowledge to improve the estimation of user and item profiles and thus improve the recommendation accuracy for cold-start cases. However, in most cases, features are usually extracted from a single domain and the processes of representation learning and rating prediction are totally independent. Some extensions such as [23] explore tightly coupled approaches to integrate topic modeling and collaborative filtering for more accurate document recommendations. Some other recent studies [24,25] deal with the content-based, collaborative-based, and hybrid mechanisms in multimedia information retrieval.

Deep learning [26,27] has emerged recently to be one of the most powerful approaches to learn robust and complex features directly from data and has shown great success in various fields such as computer vision [28,29]. Multi-modal deep learning has been applied in both traditional unsupervised and supervised learning tasks [30–32] to fuse multiple data resources with novel applications such as medical diagnosis [33]. Earlier attempts of deep learning applications in recommender systems include [34], which introduces restricted Boltzmann machines to model user-item correlations. However, these models are typically simple neural network models and do not incorporate any content information. Recent work integrating deep learning with collaborative filtering mostly focuses on extracting content features from single modality such as texts [35–37] or images [38–40]. A few exceptions such as [41] extract latent features from multi-modality contents including both texts and images through auto-encoders to improve rating prediction accuracy. Specifically, the embedding vectors are learned through auto-encoders in each modality separately and then the summation of those vectors are used to predict final ratings. However, the latent spaces learned from different modalities are not necessarily well aligned. Our framework differs from others in that it learns a shared embedding from all modalities simultaneously, which can be generally applied to incorporating heterogeneous contents such as texts and images.

3. Methods

3.1. Problem Definition and Model Overview

We consider the problem of rating prediction for recommendations, where a recommendation model is learned to predict the users' ratings on unseen items given previous ratings. As an example, one can imagine a commercial website that stores user rating history and use that information to predict the users' ratings (preferences) for their future visits. The ratings can be explicit, e.g., users give 1 to 5 stars, or implicit, e.g., users take an action or not, corresponding to a rating of 0 or 1. Let $U = \{u_1, u_2, \ldots, u_m\}$ be the set of users and $V = \{v_1, v_2, \ldots, v_n\}$ be the set of items. The collection of past ratings is a set of 3-tuples $R = \{(i, j, r_{ij})\}$, where $i \in U$, $j \in V$, and r_{ij} is user i's rating on item j.

One type of state-of-the-art recommendation models is based on matrix factorization (MF) [1,16]. The past ratings can be represented as a (sparse) matrix R. Through matrix factorization, one can learn a low-dimensional latent vector $\mathbf{u}$ for each user and a low-dimensional latent vector $\mathbf{v}$ for each item. User i's rating on item j can be predicted as $\mathbf{u}_i^\top \mathbf{v}_j$, where $\mathbf{u}_i$ and $\mathbf{v}_j$ are the low-dimensional vectors associated with user i and item j, respectively.

In order to improve rating predictions, it is helpful to incorporate content information from multiple domains, e.g., a poster (visual domain) and reviews (text domain) of a movie. Figure 1 shows the model of our approach. At a high level, the model can be divided into two main parts: (1) an autoencoder-based, multimodal feature extraction and fusion framework, which consists of three components including domain-specific encoding networks $E(k)$ for each domain k, domain-specific decoding networks $D(k)$ for each domain k, and a fusion network F. Taking content data from multiple domains as inputs, the framework generates an embedding (feature) vector for each item that fuses multi-domain information; (2) rating prediction with the fused embedding vector. Let $\mathbf{x}_{e(j)}$ be the embedding (feature) vector of item j. The new rating prediction becomes the following:

$$\mathbf{u}_i^\top (\mathbf{v}_j + \mathbf{x}_{e(j)}). \tag{1}$$

In next subsections, we give details on our deep fusion framework and the rating prediction learning.

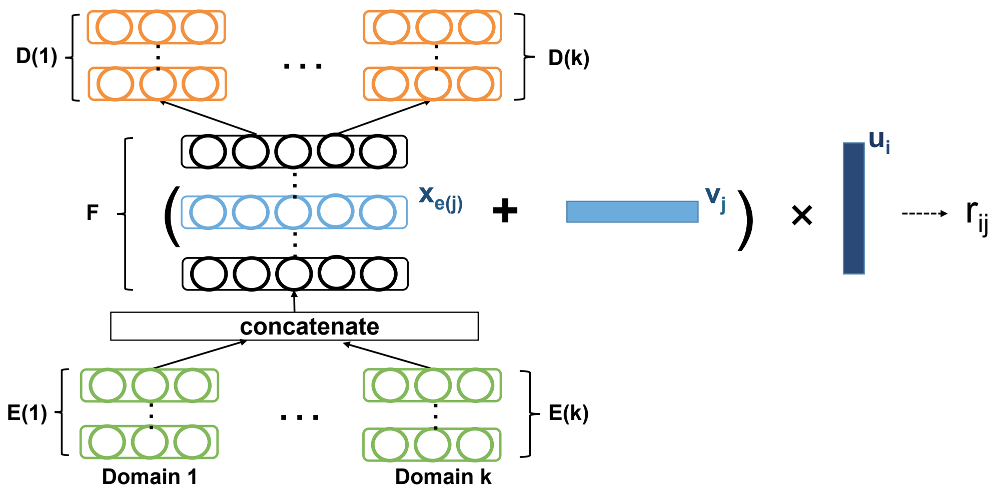

Figure 1. Rating prediction with deep fused embedding.

3.2. Deep Fusion for Multimodal Embedding

We propose a general deep fusion framework for multimodal embedding (feature extraction). (Note that we use embedding and features/feature extraction interchangeably since they all refer to finding a representation for the data). In multimodal embedding, data from multiple domains are available to describe an object. We seek an embedding vector (feature vector) that combines information from different domains to represent the item and achieve better performance than single-view learning by using this representation. In this sense, our embedding is also task related because different tasks may involve different aspects of the item. In this paper, we target our multimodal embedding for rating prediction.

An intuitive approach to using data from multiple domains is to compute an embedding independently for each domain. Then one can obtain an overall embedding by summing/averaging the domain-specific embedding vectors. However, such embedding (feature vectors) cannot capture correlations across different domains.

Consider a movie, a poster of the movie, and a text summary of the movie. One may take a hierarchical generative view on how to derive the poster and the summary. Let z be a variable describing the characteristics of the movie. z determines the distribution of two other variables, z_p and z_s. z_p is associated with the characteristics of the poster and determines a distribution from which we can sample the poster. z_s is the same for the summary. This simple generative view shows that some features of the movie (z) may be connected to both the poster and the summary. The alternative approach described above treats the two domains independently and cannot extract these features well.

We design our deep fusion network based on this generative view. As shown in Figure 1, for each domain, our framework has a sub-network that extracts domain-specific features (e.g., those corresponding to z_p or z_s) from the domain input. These features are then fed to (the first half of) the central fusion network that combines the features from multiple domains and further extracts fused high-level features (e.g., those corresponding to z). In such a fashion, the fused high-level features can capture correlations between different domains and provide a better description of the characteristics of the item represented by multi-domain inputs. We utilize these fused high-level features in rating prediction later.

Our deep fusion network is modified from stacked denoise autoencoder (SDAE) [42]. The embedding (feature extraction) component that we describe above can be viewed as an multi-domain fusion encoder. Different from probabilistic models, with autoencoder, the encoder is coupled with a decoder for training. Our deep fusion framework contains a decoding structure that consists of several components too. First, the fused high-level features go through several layers of (joined) decoding (the second half of the central fusion network). Then, for each individual domain, we have a domain-specific decoder that reconstructs the data for that domain.

A deep neural network consists of multiple layers of neurons. Given an input data point $\mathbf{z}$ (a row vector) and a layer l, we have the following computation within a layer that generates output $\mathbf{x}_l$:

$$\mathbf{x}_l = \sigma(\mathbf{z}\mathbf{W}_l + \mathbf{b}_l), \tag{2}$$

where $\mathbf{W}_l$ and $\mathbf{b}_l$ are the weights and the bias of the layer respectively. Function σ is a nonlinear function. To simplify the notation, for layer l, we denote the above computation by $\mathbf{Net}^l$ and have $\mathbf{x}_l = \mathbf{Net}^l(\mathbf{z})$. A sequence of layers can be stacked such that the output of layer $l-1$ serves as the input to layer l, i.e., $\mathbf{x}_l = \mathbf{Net}^l(\mathbf{x}_{l-1})$ for all $l > 0$ in the sequence and x_0 is the input to the stack. For a stack S of $|S|$ layers (we use $|\cdot|$ as the operator that returns the number of layers), we denote by $\mathbf{Net}^S$ this computation and have

$$\mathbf{x}_{|S|} = \mathbf{Net}^S(\mathbf{x}_0), \tag{3}$$

where $\mathbf{x}_{|S|}$ is the output of the last layer (layer $|S|$). (Note that if $|S| = 0$, the output of the stack is the same as the input).

For each domain d, let $\mathbf{x}^d$ be an original input data point from the domain and $\tilde{\mathbf{x}}^d$ be the noise-corrupted input. Specifically, the input in text domain is represented as a bag-of-words vector and we randomly mask some entries of the input by making them zero. For image domain, the inputs are corrupted with Gaussian noise. We have a stack of domain-specific encoding layers (encoder) for each domain d. We call this stack of layers $E(d)$, and denote its computation by $\mathbf{Net}^{E(d)}$. Suppose that there are k domains. The encoders produce a set of k output vectors $\{\mathbf{x}_c^1, \mathbf{x}_c^2 \ldots, \mathbf{x}_c^k\}$, where $\mathbf{x}_c^i = \mathbf{Net}^{E(i)}(\tilde{\mathbf{x}}^i)$. We denote by $\mathbf{x}_t = \mathbf{x}_c^1 || \mathbf{x}_c^2 || \cdots || \mathbf{x}_c^k$ the concatenation of these vectors.

At the center of our network is the stack of layers (F) that fuses features from multiple domains. (We call it the fusion network.) It takes $\mathbf{x}_t$ as input and generates output $\mathbf{x}_p = \mathbf{Net}^F(\mathbf{x}_t)$. The first half of the stack serves as (fuse) encoder and the second half of the stack as (fuse) decoder. The embedding computed by our framework thus is the output of the layer $|F|/2$ in the network F, i.e., $\mathbf{x}_e = \mathbf{Net}^{F(1/2)}(\mathbf{x}_t)$, where $F(1/2)$ is the first half of the network F. We refer to the second half as $F(2/2)$.

During the training, the model is given corrupted domain inputs, and is trained to predict the original inputs as a result of decoding. The final decoding involves a set of domain-specific decoders ($D(d)$ for each domain d). Each decoder takes $\mathbf{x}_p$ as input and output a reconstruction $\mathbf{Net}^{D(d)}(\mathbf{x}_p)$ for the original input $\mathbf{x}^d$. We call the reconstruction $\mathbf{x}_r^d$.

A summary of notations we introduced is given in Table 1.

The process to compute $\mathbf{x}_e$ and $\mathbf{x}_r$ is summarized in Algorithm 1.

In an unsupervised setting, one may minimize the reconstruction loss:

$$\mathcal{L}_{rec} = \sum_d ||\mathbf{x}_r^d - \mathbf{x}^d||_2^2 \tag{4}$$

to train the model and use the trained model to obtain multimodal embedding for the data. However, as we have discussed earlier, embedding is task related. Therefore, we consider a semi-supervised model where the training involves both the reconstruction loss and a task-specific loss.

Table 1. Summary of Notations.

$\mathbf{u}, \mathbf{v}$	user and item latent vector
r_{ij}	user i's rating on item j
d	data domain
$\mathbf{x}^d, \tilde{\mathbf{x}}^d$	input and corrupted input from d
$E(d)$	domain-specific encoding network for d
$D(d)$	domain-specific decoding network for d
F	fusion network
$\mathbf{Net}^S(\cdot)$	computation done by network S
$\mathbf{x}_r^d$	reconstruction for domain d input
$\mathbf{x}_e$	the embedding vector
Θ	parameters of network (E, D and F)

Algorithm 1: Computing $\mathbf{x}_e$ and $\mathbf{x}_r$ for a single item.

```
Input: Inputs $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^k$ from $k$ domains
Output: $\mathbf{x}_e$ and $\{\mathbf{x}_r^i\}_{i=1}^k$
foreach $i$ in $1, 2, \ldots, k$ do
    Generate corrupted input $\tilde{\mathbf{x}}^i$ by adding noise to $\mathbf{x}^i$
    $\mathbf{x}_c^i = \mathbf{Net}^{E(i)}(\tilde{\mathbf{x}}^i)$
$\mathbf{x}_t = \text{concat}(\{\mathbf{x}_c^i\}_{i=1}^k)$;
$\mathbf{x}_e = \mathbf{Net}^{F(1/2)}(\mathbf{x}_t)$;
$\mathbf{x}_p = \mathbf{Net}^{F(2/2)}(\mathbf{x}_e)$;
foreach $i$ in $1, 2, \ldots, k$ do
    $\mathbf{x}_r^i = \mathbf{Net}^{D(i)}(\mathbf{x}_p)$
return $\mathbf{x}_e$, $\{\mathbf{x}_r^i\}_{i=1}^k$
```

3.3. Heterogeneous Domain-Specific Encoders and Decoders

Although we present the domain-specific encoders and decoders as regular (fully connected) neural networks in the description of our deep fusion framework, there is no technical constraint that limits the type of the neural networks being used. One can use convolutional networks or other types of neural networks. Our deep fusion is an unified framework for multimodal embedding that can incorporate heterogeneous domain encoders (decoders).

The capability of combining different types of encoders (decoders) is particularly pertinent when we fuse image domain with other domains. Many studies have shown that convolutional neural networks render much better performance with images than regular fully connected networks [28,43]. As a result, it is more suitable to first apply a convolutional network that extracts domain-specific features from the images, and then use these features in multimodal fusion, where they will be combined with features from other domains to generate high-level features that involve all the domains.

Since it is a general understanding that deep models require large datasets to train, a deep convolutional network can be a good choice for the domain-specific encoder (decoder) if there is a large amount of data. On the other hand, in a situation where the number of images in the dataset is quite limited (comparing to that of the ImageNet), it would not be ideal to use such data to train both a deep convolutional network and a deep fusion model.

One may employ a less-deep convolutional network as the domain-specific encoder (decoder). But a less-deep network (e.g., a convolutional network of 2–3 layers) has some drawbacks. For large images (several hundred by several hundred pixels), the feature vector obtained by unrolling the feature map produced by the convolutional network may have a very high dimensionality. In order to get lower dimensionality, one may apply a very large filter size, stride, or pooling window. These are not the best practices for convolutional networks. Oftentimes, people choose to shrink the image to a small size instead. However, shrinking causes loss of details in the image (e.g., a movie poster of 50×50 pixels will not be elaborate).

Given these considerations, there is a third solution in our framework to deal with image domain when data are limited. Borrowing the idea from transfer learning, one could employ a pretrained existing model as image domain feature extractor. There are quite a few existing models such as AlexNet and VGG_net that are already trained and available. It is one of the benefits from our unified fusion framework to incorporate existing trained models for multi-domain fused embedding.

3.4. Learning Rating Prediction with Deep Fusion

We apply our deep fusion framework to extract information from multiple domains and integrate such information in rating prediction. For our model presented in Figure 1, the variable values that we need to learn are the low-dimensional latent vector for users $\{\mathbf{u}_i\}$, the low-dimensional latent vector for items $\{\mathbf{v}_j\}$, and the parameters of the whole neural network Θ. (Note that once we have the network parameters, each item's fused embedding vector, i.e., $\mathbf{x}_e$, can be computed by using the procedure in Algorithym 1.)

We use a training process that minimizes a combination of reconstruction loss and rating loss to obtain the values for the parameters. Equation (4) provides the reconstruction loss for a single item. The rating loss for a single user-item pair (i, j) is given by:

$$\mathcal{L}_{rating} = (r_{ij} - \mathbf{u}_i^\top(\mathbf{v}_j + \mathbf{x}_{e(j)}))^2. \tag{5}$$

Putting both errors together and adding regularizations, we have the following overall loss:

$$\begin{aligned} \mathcal{L} = &\textstyle\sum_{(i,j,*)\in R}\left(r_{ij} - \mathbf{u}_i^\top(\mathbf{v}_j + \mathbf{x}_{e(j)})\right)^2 \\ &+\lambda_1 \textstyle\sum_j \sum_d ||\mathbf{x}^d_{r(j)} - \mathbf{x}^d_j||^2_2 + \lambda_2 \sum_t ||\mathbf{u}_t||^2_2 + \lambda_3 \sum_t ||\mathbf{v}_t||^2_2, \end{aligned} \tag{6}$$

where λ_1, λ_2, and λ_3 are model parameters. λ_1 controls the trade-off between rating loss and reconstruction loss while λ_2 and λ_3 control the regularization. To solve the minimization problem, we used stochastic gradient descending (SGD).

4. Experiments

We conducted a collection of experiments to evaluate our proposed model and its variations. We also compared the proposed models to some other state-of-the-art models. Our experiments used two datasets: the book dataset and the movie dataset, both from Amazon product dataset [38]. In the following, we first describe the datasets, the models used in comparison and the experiment configuration. Then we discuss the experimental results.

4.1. Datasets

The **Movie dataset** contains reviews of movies, movie images, and movie ratings. We use movies' and TVs' review data and their metadata from Amazon product dataset (http://jmcauley.ucsd.edu/data/amazon). The reviews contain 4,607,047 pieces of review information including user ID, movie ID, rating, and textual review. The metadata contains 208,321 pieces of image information including movie ID and image link. The ratings range 1–5. Traditional benchmark datasets such as Movielens1M are relatively dense. The density is around 4% and each user has at least 20 ratings. We would like to process datasets to better simulate cold-start scenarios. Specifically, we get 5921 users after removing users with fewer than 40 ratings, and then we get 10,520 movies after removing movies with fewer than 80 ratings or without image. This results in 394,980 valid ratings between these users and movies and the rating density is around 0.63%. Around 20% users will have fewer than 20 ratings among the selected movies. **Book dataset**: We use books' review data and metadata from Amazon product dataset. From 8,898,041 pieces of raw review information and 2,370,585 pieces of raw metadata, we get

9085 users after removing users with fewer than 100 ratings, and then we get 13,333 books after removing books with fewer than 100 ratings. This results in 553,840 valid ratings between these users and books and the rating density is around 0.46%. Around 30% users will have fewer than 20 ratings among the selected books. In comparison with traditional benchmark datasets, our processed datasets better simulate cold-start scenarios.

For the review data, we concatenate all reviews for each movie into a single review file, and then use Bag-of-Words model to convert the review file into a 5000-dimension vector. In terms of image domain, some models use raw images as domain input. For the other models, we apply pretrained AlexNet [28] to the images and obtain image features. These features are then used as image domain inputs. We resize our image data to 227×227 and feed them to AlexNet. The outputs from the neurons before the softmax operation in AlexNet are used as image domain inputs. This type of inputs has a dimensionality of 1000.

4.2. Models Used in Experiments

In our deep fusion network, for both the text and image domains, the inputs are directly concatenated and fed to the fusion autoencoder. (Note that the image domain inputs are the features derived from trained AlexNet.) For each domain, there is a one-layer domain-specific decoder to reconstruct the input. A 5-layered network is used as our central fusion network (network F). The output of the 3rd layer is the fused embedding. All neurons in the network are rectified linear units.

We compare our deep fusion rating prediction with several other models. These models utilize two other types of networks: standard autoencoder with fully connected layers (FCAE) and standard autoencoder with convolutional and fully connected layers (SCAE). Compressed features with dimension m extracted from the last layer of encoder are used as the embedding of the data. FCAE is a 6-layer fully-connected neural network with the first 3 layers as an encoder and the last 3 layers as a decoder. The number of neurons used in each layer of FCEA is: 1000, 500, m, 500, 1000, 5000, respectively, and Sigmoid function is used as the activation function for all layers. SCAE, mainly for images, is a 6-layer neural network, in which the first two layers and the last two layers are convolutional, and the third and fourth layers are fully-connected. The number of convolutional filters and window size in each layer of SCAE are: (20, 5, 5), (20, 5, 5), m, 81,920, (20, 5, 5), (3, 5, 5), respectively. We use rectified linear unit (ReLU) as the activation function for each layer, and RMSProp as the optimizer for SCAE.

We now list the models used in our experiments and a quick comparison of the model properties is given in Table 2.

Table 2. Comparison of Models

Method	Fused Embedding	Embedding Coupled Rating	Pretrained
MF	✗	✗	✗
MF + T	✗	✗	✗
CTR	✗	✓	✗
MF + I	✗	✗	✗
MF + IT	✗	✗	✗
MF * IT	✗	✓	✗
MF + I$_p$	✗	✗	✓
MF + I$_p$T	✗	✗	✓
MFUI$_p$T	✓	✓	✓

MF: This is the traditional matrix factorization model [1], no review or image information is added. The rating of user i on item j is simply predicted as $\mathbf{u}_i^\top \mathbf{v}_j$. **MF + T**: Matrix factorization with embedding vector from the reviews. Note that, different from our main model, embedding vector is derived through unsupervised training in this model. A FCAE is trained on the review data (text domain) in an unsupervised fashion at first to generate an embedding for the review data ($\mathbf{e}_T$). The embedding vector is incorporated in the matrix factorization for rating prediction, i.e., the rating of user i on item j is predicted as $\mathbf{u}_i^\top(\mathbf{v}_j + \mathbf{e}_T)$. In this scenario, we say that learning of the embedding vector and learning of the rating are *not coupled*. **CTR**: Collaborative topic regression [23] integrates topic modeling (bag of words) and CF simultaneously. The learning of embeddings and the rating predictions are coupled. A pair-wise loss function is used to measure the differences of relative preferences of pairs of items. **MF + I**: Same as **MF + T** with a SCAE in place of the FCAE to learn image features.

MF + IT: Same as **MF + T** and **MF + I** but includes embedding from both the reviews ($\mathbf{e}_T$) and the images ($\mathbf{e}_I$), using a FCAE for the reviews and a SCAE for the images. The rating of user i on item j is predicted as $\mathbf{u}_i^\top(\mathbf{v}_j + \mathbf{e}_T + \mathbf{e}_I)$. Again, the embedding learning and the rating learning are not coupled.

MF * IT: This is a simplified version of the model proposed in [41], where the input contains text and visual domains. A FCAE is used for text domain data and a SCAE for image domain data. Different from **MF + T**, **MF + I**, and **MF + IT**, the embedding learning and the rating learning are *coupled* in this model. The loss function used in training includes both a reconstruction loss (for embedding) and a rating loss. On the other hand, the text domain and the image domain embeddings are extracted using two separate autoencoders. In this case, we say that the embedding is *not fused embedding*.

MF + I$_p$: This is a slight variation of the model proposed in [39], which integrates only contents in image domain to improve recommendation. The difference is in that we use item-wise loss function while they use pair-wise loss function. The reason that we use item-wise loss function is to make the comparison consistent with the strongest baseline [41]. **MF + I$_p$** is the same as **MF + I** except that the image domain inputs are features extracted by AlexNet rather than the raw images. Accordingly, this model uses a FCAE instead of SCAE because we view AlexNet features as a feature vector, rather than a feature map. Same as **MF + I**, the embedding learning is *not coupled* with the rating learning. **MF + I$_p$T**: Same as **MF + I$_p$** with text domain inputs added. We use a FCAE for the text domain embedding. The embedding learning and the rating learning are *not coupled*. **MFUI$_p$T**: This is our main model (model in Figure 1). It uses features extracted by AlexNet as image domain inputs and generates a *fused embedding* for both image and text features. The training for the embedding and the rating is *coupled*.

4.3. Evaluation Scheme

We measure the performance of a rating prediction model by computing the error between the predicting ratings and the real rating of items. In particular, we use mean squared error (MSE) and mean absolute error (MAE) to evaluate the performance. In addition, to compare with rank-based models, we rank the test items by rating scores and adopt normalized discounted cumulative gain (NDCG) to measure ranking quality. We use 5-fold cross-validation technique to evaluate both Book and Movie datasets. To optimize the user latent u_i and item latent v_j in matrix factorization, we implement the stochastic gradient descent (SGD) algorithm. The parameters include the dimension of latent vector m, the learning rate lr, the regularization factor λ, the batch size *batch*, and the number of iterations *iter*. For Movie dataset, we use $lr = 0.015$ and $\lambda = 0.15$. For Book dataset, we use $lr = 0.003$ and $\lambda = 0.2$. For both datasets, latent factors m changes from 50, 100, 150, 200 to 300. The parameter settings are shown in Table 3. m denotes the dimension of latent vector, lr is the learning rate, λ is the regularization factor, *batch* is the batch size, and *iter* is the number of iterations.

Table 3. Parameter Settings of Different Parts.

	Movie Dataset	**Book Dataset**
MF	m = 50, lr = 0.01, λ = 0.15	m = 50, lr = 0.006, λ = 0.2
	m = 100, lr = 0.01, λ = 0.15	m = 100, lr = 0.006, λ = 0.2
	m = 150, lr = 0.015, λ = 0.15	m = 150, lr = 0.003, λ = 0.2
	m = 200, lr = 0.015, λ = 0.15	m = 200, lr = 0.003, λ = 0.2
	m = 300, lr = 0.015, λ = 0.15	m = 300, lr = 0.003, λ = 0.2
FCAE	lr = 0.05, batch = 32, iter = 500	lr = 0.05, batch = 32, iter = 500
SCAE	lr = 0.01, batch = 32, iter = 100	lr = 0.01, batch = 32, iter = 100

5. Results and Discussion

Figure 2 shows the MSE (top) and MAE (bottom) of different rating prediction models across different embedding dimensions for the Movie dataset (left) and the Book dataset (right). For both datasets, the hybrid models (models that include content information such as images and reviews) outperform the pure matrix factorization model (**MF**), across all embedding dimensions. As an example, consider MSE for the movies dataset. In average (across embedding dimensions), the **MF + T** model achieves 1.3% improvement over the pure matrix factorization model (**MF**) and the **MF + I** achieves 2.3% improvement. For the books dataset, the **MF + T** model achieves 0.73% improvement and the **MF + I** achieves 4.3% improvement. This agrees with the well-known view in recommendation research that hybrid systems often give better performance.

A second observation from the result is that in all cases, coupling the learning of the embedding and that of the rating always provides some benefit. **MFUI**$_p$**T** (**MF * IT**) is the best (second best) performers across all embedding dimensions and datasets. As we have discussed earlier, embedding is task related. Different tasks may involve different aspects of an item. Features that are discriminative for one task may not be so for another task and general features derived from unsupervised learning may not be beneficial to a particular task. Our results show such a trend clearly. The performance of the decoupled models where the embeddings are learned through unsupervised learning cannot match that of the coupled models.

There is another separation between the decoupled models and the coupled ones. For the coupled models, within the range tested in our experiments, increasing embedding dimension leads to slightly better performance (our model **MFUI**$_p$**T** in most cases) or the performance stays roughly the same (**MF * IT** in most cases). On the other hand, for the decoupled models, the performance, in most cases, gets worse when the embedding dimension increases. We notice that the performance of the pure matrix factorization model, **MF**, decreases when the latent dimension increases. (Latent dimension is always the same as the embedding dimension if a model uses embedding.) It is likely that with a high latent dimension, **MF** reaches overfitting quickly. Hence, its performance deteriorates when dimension increases. Although the decoupled models still perform better than **MF**, the benefit from their embedding vectors is not large enough to reverse the trend (i.e., performance deteriorating due to overfitting with high dimension). On the other hand, although the coupled models use the same base model (**MF**), their embedding vectors are more beneficial and can reverse (or cancel) the overfitting trend.

The effect of the features from AlexNet is mixed. There are cases where using such features can benefit while there are also other cases where they give slightly worse performance than SCAE with raw images.

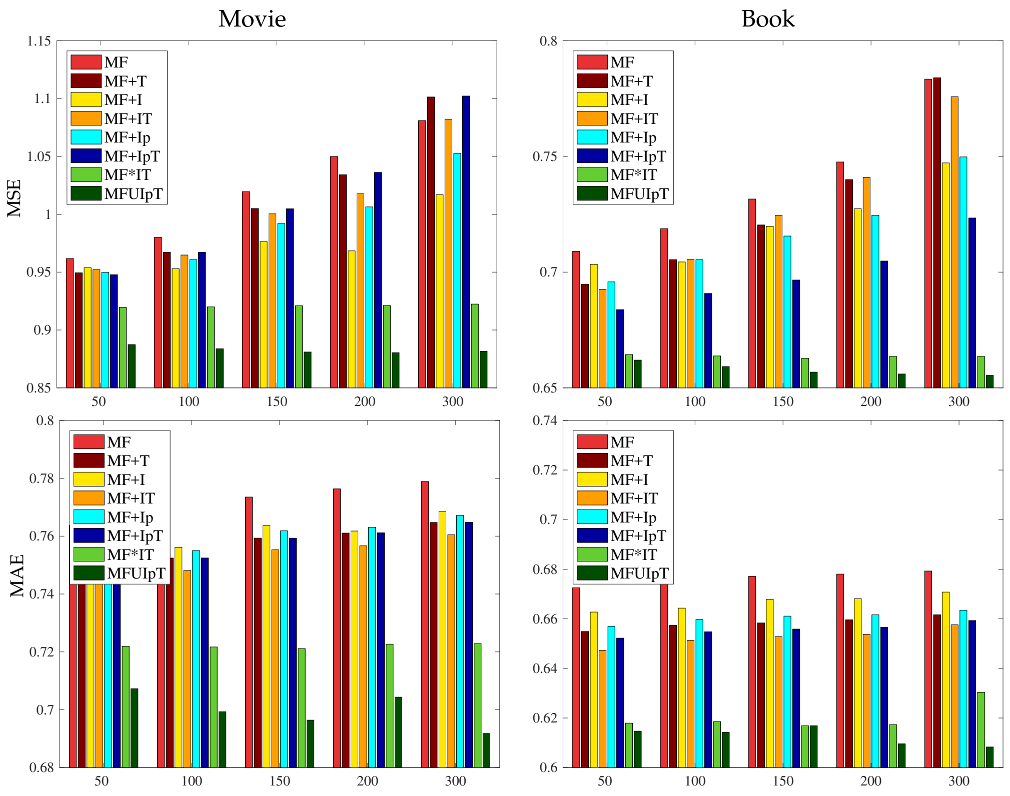

Figure 2. Rating prediction measured by mean squared error (MSE) and mean absolute error (MAE) with respect to latent dimension size on Movie (**left**) and Book (**right**).

Across all dimensions and all datasets, our main model, **MFUI$_p$T**, is constantly the best performer. As an example, we compare it with the second best model **MF * IT**. For the Movie dataset, the **MF * IT** model achieves improvement from 4.39% to 14.67% with respect to the base model **MF**, when the embedding dimension changes from 50 to 300. The average improvement is 9.43%. Our main model achieves improvement from 7.74% to 18.45% compared to **MF**. The average improvement is 13.15%. Furthermore, as we have discussed earlier, the base model **MF** overfits at high dimensions and its performance decreases. Our model can reverse such trends, even at the highest dimension. These results indicate that our model has a performance advantage over all other models. Given our model and **MF * IT** are both coupled models, a main part of our model's performance advantage is due to the fused embedding, which is capable of extracting features involving multiple domains.

In Table 4, we compare two of the previous rating-based recommendation models with rank-based recommendation model CTR on Movie dataset. The latent dimension for all models is set to 50 and all parameters are optimized through cross-validation. Since the first two models integrate both image and text features, they perform better (with higher NDCG) than single-modal model CTR (text).

Table 4. Movie recommendation performance using rank-based metric normalized discounted cumulative gain (NDCG) at top K positions.

Method	$k = 5$	$k = 10$	$k = 15$	$k = 20$
MF * IT	0.9320	0.9513	0.9587	0.9619
MFUI$_p$T	0.9370	0.9546	0.9614	0.9644
CTR	0.8978	0.9265	0.9376	0.9425

Figure 3 shows the illustrative examples of our recommendation model. The top row is the movie user case and the bottom row is the book user case. The three items on the left side of the bar are selected from users' favorite items with the highest ratings, and the three items on the right are top items recommended by our model. For each item, we show both image content such as a movie poster and a book cover, and text content that is the most frequent word in the corresponding reviews. First of all, we can see that features from different domains can complement each other and provide better descriptions of the characteristics of the items. For example, for most movies listed on the top, their images are mostly about people and it is difficult to identify the finer genres. By reading the text, we know that some movies are intense drama like "The Hunt" and some are relaxing comedy like "Nurse Jackie". In addition, some of the books are very good examples of "Don't judge a book by its cover". In particular, the book "Prayers for sale" is simply a blue background with white characters and it is hard to guess what the content is about, so text can help.

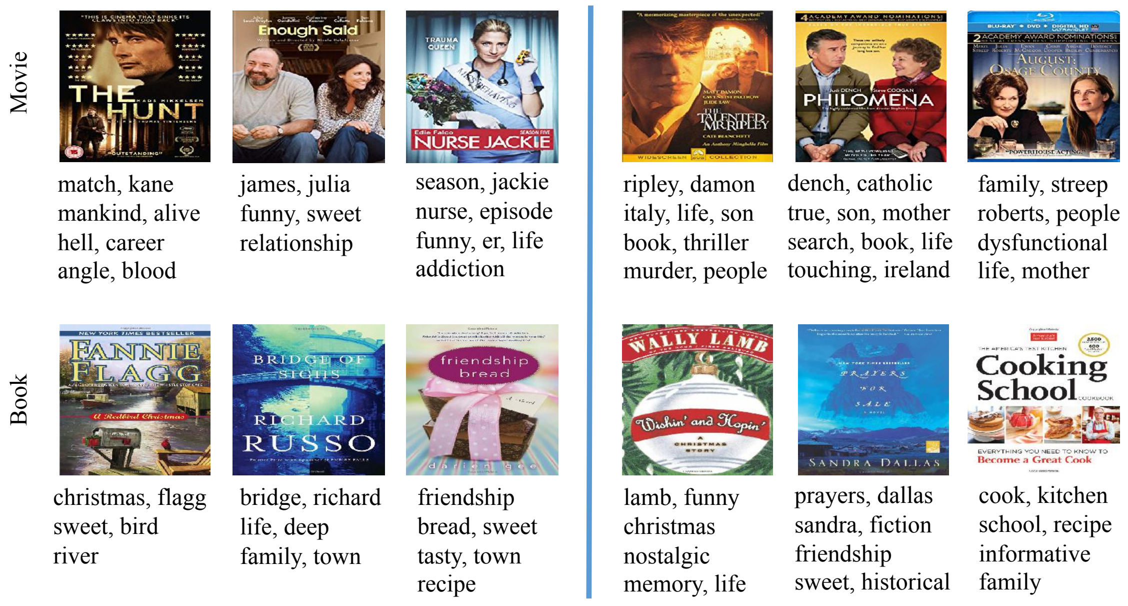
Movie
match, kane mankind, alive hell, career angle, blood
james, julia funny, sweet relationship
season, jackie nurse, episode funny, er, life addiction
ripley, damon italy, life, son book, thriller murder, people
dench, catholic true, son, mother search, book, life touching, ireland
family, streep roberts, people dysfunctional life, mother
Book
christmas, flagg sweet, bird river
bridge, richard life, deep family, town
friendship bread, sweet tasty, town recipe
lamb, funny christmas nostalgic memory, life
prayers, dallas sandra, fiction friendship sweet, historical
cook, kitchen school, recipe informative family

Figure 3. Case study for a Movie user (**top**) and a Book user (**bottom**): (**Left**) user's top 3 favorite items. (**Right**) top 3 items our model recommends.

For both cases, we can see that our model successfully captures the users' tastes and provides reasonable recommendations. Specifically, from the three liked movies "The Hunt", "Enough Said" and "Nurse Jackie", we can speculate that the movie user likes drama and romantic comedy, with a specific taste on movies discussing relationship management. Our model recommends "Mr. Ripley", "Philomena", and one Streep's movie, which aligns with the user's taste. Similar phenomena can be observed for the book user. It seems that the user likes books about religious, family, friendship, and cooking from the three training examples and the recommendations mirror the taste.

Finally, the case study perfectly illustrates the effectiveness of our hybrid modeling via deep embedding, which benefits from both collaborative filtering and content filtering. For very popular items such as "Nurse Jackie" and Streep's movie, they can be linked via rating patterns. lower-rated items such as "Friendship Bread" and "The cooking school", which may not be linked together by collaborative filtering methods, can be linked through text content similarities.

6. Conclusions

We have considered the rating prediction problem with content information in recommender systems. To utilize the information that often includes data from multiple domains, we have proposed a deep fusion framework for multimodal embedding. The key characteristic of our framework is the fused embedding that produces high-level features combining multiple domains tightly. We have trained the deep fusion embedding with rating prediction in a closely coupled fashion. To evaluate our model, we have conducted a set of experiments and compared it with other existing models and variations. The experimental results have shown that rating prediction with our deep fusion embedding gives the best performance in all experiments across all datasets. This demonstrates the effectiveness of our model. As future work, we would like to explore more network structures to generate fused embedding for data from vastly different domains, for recommendation tasks and other applications.

Acknowledgments: We would like to thank the reviewers for their constructive and insightful comments. This work was supported in part by the Louisiana Board of Regents under Grant LEQSF(2017-20)-RD-A-29.

Author Contributions: Mingxuan Sun and Jian Zhang conceived and designed the experiments; Mingxuan Sun, Fei Li, and Jian Zhang performed the experiments; Mingxuan Sun and Jian Zhang analyzed the data; Mingxuan Sun and Jian Zhang wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Koren, Y. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data* **2010**, *4*, 1–24, doi:10.1145/1644873.1644874.
2. Liu, B.; Fu, Y.; Yao, Z.; Xiong, H. Learning geographical preferences for point-of-interest recommendation. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 1043–1051.
3. Chen, X.; Zhang, Y.; Ai, Q.; Xu, H.; Yan, J.; Qin, Z. Personalized key frame recommendation. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 315–324.
4. Sun, M.; Li, C.; Zha, H. Inferring Private Demographics of New Users in Recommender Systems. In Proceedings of the ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Miami, FL, USA, 21–25 November 2017.
5. Hofmann, T. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.* **2004**, *22*, 89–115.
6. Pennock, D.M.; Horvitz, E.; Lawrence, S.; Giles, C.L. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, San Francisco, CA, USA, 30 June–3 July 2000; pp. 473–480.
7. Lee, J.; Sun, M.; Kim, S.; Lebanon, G. Automatic Feature Induction for Stagewise Collaborative Filtering. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012.
8. Lee, J.; Sun, M.; Lebanon, G. PREA: Personalized Recommendation Algorithms Toolkit. *J. Mach. Learn. Res.* **2012**, *13*, 2699–2703.
9. Li, L.; Chu, W.; Langford, J.; Schapire, R.E. A contextual-bandit approach to personalized news article recommendation. In Proceedings of the International Conference on World Wide Web, Raleigh, NC, USA, 26–30 April 2010; pp. 661–670.
10. Yang, Y.; Yoo, S.; Zhang, J.; Kisiel, B. Robustness of adaptive filtering methods in a cross-benchmark evaluation. In Proceedings of the ACM SIGIR Conference, Salvador, Brazil, 15–19 August 2005; pp. 98–105.
11. Gabrilovich, E.; Dumais, S.; Horvitz, E. Newsjunkie: Providing personalized newsfeeds via analysis of information novelty. In Proceedings of the International Conference on World Wide Web, New Yrok, NY, USA, 17–22 May 2004; pp. 482–490.
12. Chu, W.; Park, S.T. Personalized recommendation on dynamic content using predictive bilinear models. In Proceedings of the International Conference on World Wide Web, Madrid, Spain, 20–24 April 2009; pp. 691–700.
13. Breese, J.; Heckerman, D.; Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, Madison, WI, USA, 24–26 July 1998; pp. 43–52.

14. Herlocker, J.L.; Konstan, J.A.; Borchers, A.; Riedl, J. An algorithmic framework for performing collaborative filtering. In Proceedings of the ACM SIGIR Conference, Berkeley, CA, USA, 15–19 August 1999; pp. 230–237.
15. Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. Item-based collaborative filtering recommendation algorithms. In Proceedings of the International Conference on World Wide Web, Hong Kong, China, 1–5 May 2001; pp. 285–295.
16. Rennie, J.; Srebro, N. Fast maximum margin matrix factorization for collaborative prediction. In Proceedings of the International Conference on Machine Learning, Bonn, Germany, 7–11 August 2005; pp. 713–719.
17. Sun, M.; Lebanon, G.; Kidwell, P. Estimating Probabilities in Recommendation Systems. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 734–742.
18. Sun, M.; Lebanon, G.; Kidwell, P. Estimating Probabilities in Recommendation Systems. *J. R. Stat. Soc. Ser. C* **2012**, *61*, 471–492.
19. Sun, M.; Li, F.; Lee, J.; Zhou, K.; Lebanon, G.; Zha, H. Learning multiple-question decision trees for cold-start recommendation. In Proceedings of the Conference on Web Search and Data Mining, Rome, Italy, 4–8 February 2013.
20. Agarwal, D.; Chen, B. Regression-based latent factor models. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009; pp. 19–28.
21. Gunawardana, A.; Meek, C. Tied boltzmann machines for cold start recommendations. In Proceedings of the 2008 ACM Conference on Recommender Systems, Lausanne, Switzerland, 23–25 October 2008; pp. 19–26.
22. Schein, A.I.; Popescul, A.; Ungar, L.H.; Pennock, D.M. Methods and metrics for cold-start recommendations. In Proceedings of the ACM SIGIR Conference, Tampere, Finland, 11–15 August 2002; pp. 253–260.
23. Wang, C.; Blei, D.M. Collaborative topic modeling for recommending scientific articles. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 21–24 August 2011; pp. 448–456.
24. Stai, E.; Kafetzoglou, S.; Tsiropoulou, E.E.; Papavassiliou, S. A holistic approach for personalization, relevance feedback & recommendation in enriched multimedia content. *Multimedia Tools Appl.* **2018**, *77*, 283–326.
25. Pouli, V.; Kafetzoglou, S.; Tsiropoulou, E.E.; Dimitriou, A.; Papavassiliou, S. Personalized multimedia content retrieval through relevance feedback techniques for enhanced user experience. In Proceedings of the 2015 13th International Conference on IEEE Telecommunications (ConTEL), Graz, Austria, 13–15 July 2015; pp. 1–8.
26. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.
27. Hinton, G.E.; Salakhutdinov, R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507.
28. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
29. Hinton, G.; Deng, L.; Yu, D.; Dahl, G.; Mohamed, A.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T.; et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **2012**, *29*, 82–97.
30. Ngiam, J.; Khosla, A.; Kim, M.; Nam, J.; Lee, H.; Ng, A.Y. Multimodal deep learning. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 689–696.
31. Andrew, G.; Arora, R.; Bilmes, J.; Livescu, K. Deep canonical correlation analysis. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1247–1255.
32. Wang, W.; Arora, R.; Livescu, K.; Bilmes, J. On deep multi-view representation learning. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1083–1092.
33. Wang, Q.; Sun, M.; Zhan, L.; Thompson, P.; Ji, S.; Zhou, J. Multi-Modality Disease Modeling via Collective Deep Matrix Factorization. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 1155–1164.
34. Salakhutdinov, R.; Mnih, A.; Hinton, G. Restricted Boltzmann machines for collaborative filtering. In Proceedings of the International Conference on Machine Learning, Corvalis, OR, USA, 20–24 June 2007; pp. 791–798.

35. Li, S.; Kawale, J.; Fu, Y. Deep collaborative filtering via marginalized denoising auto-encoder. In Proceedings of the ACM International Conference on Information and Knowledge Management, Melbourne, Australia, 19–23 October 2015; pp. 811–820.
36. Wang, H.; Wang, N.; Yeung, D.Y. Collaborative deep learning for recommender systems. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Melbourne, Australia, 19–23 October 2015; pp. 1235–1244.
37. Elkahky, A.M.; Song, Y.; He, X. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In Proceedings of the International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 278–288.
38. McAuley, J.; Targett, C.; Shi, Q.; Van den Hengel, A. Image-based recommendations on styles and substitutes. In Proceedings of the ACM SIGIR Conference, Santiago, Chile, 9–13 August 2015; pp. 43–52.
39. He, R.; McAuley, J. VBPR: Visual Bayesian personalized ranking from implicit feedback. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 144–150.
40. Geng, X.; Zhang, H.; Bian, J.; Chua, T.S. Learning image and user features for recommendation in social networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4274–4282.
41. Zhang, F.; Yuan, N.J.; Lian, D.; Xie, X.; Ma, W.Y. Collaborative knowledge base embedding for recommender systems. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 353–362.
42. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* **2010**, *11*, 3371–3408.
43. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.