Information Fusion 60 (2020) 25–40

Contents lists available at ScienceDirect

Information Fusion

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

Predicting and ranking box office revenue of movies based on big data

Zhaoyuan Wang[a], Junbo Zhang[a,b,c], Shenggong Ji[a], Chuishi Meng[b,c], Tianrui Li[a,*], Yu Zheng[a,b,c]

[a] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China
[b] JD Intelligent Cities Research, JD.com, Beijing 100176, China
[c] JD Intelligent Cities Business Unit, JD Digits, Beijing 100176, China

A R T I C L E I N F O

Keywords:
Feature learning
Data fusion
Dynamic network embedding
Heterogeneous information network
Big data

A B S T R A C T

Predicting box office revenue (BOR) of movies before releasing on big screens successfully becomes an emerging need, as it informs investment decisions on the stock market, the design of promotion strategies by advertisement companies, movie scheduling by cinemas, etc. However, the task is very challenging as it is affected by a lot of complex factors. In this paper, we first provide a strategic investigation of these influential factors. Then, we put forward a novel framework to predict a movie's BOR by modeling these factors using big data. Specifically, the framework consists of a series of feature learning models and a prediction and ranking model. In particular, there are two models devised for learning features: (1) a novel dynamic heterogeneous network embedding model to simultaneously learn latent representations of actors, directors, and companies, capable of capturing their cooperation relationship collectively; (2) a deep neural network-based model designed to uncover high-level representations of movie quality from trailers. Based on the learned features, we train a mutually-enhanced prediction and ranking model to obtain the BOR prediction results. Finally, we apply the framework to the Chinese film market and conduct a comprehensive performance evaluation using real-world data. Experimental results demonstrate the superior performance of both extracted knowledge and the prediction results.

1. Introduction

Investing in companies from the film industry on the stock market is harder than investing other companies in traditional industries because these film-related companies' stocks are strongly affected by the financial performance of their upcoming movies in the short term. Hence, accurate and reliable prediction of the Box Office Revenue (BOR) of a movie before releasing is an in pressing need, as it greatly informs the tough investment decision-making process when facing such a high-risk and high-yield opportunity. In addition, the prediction is highly important for advertisement companies that seek to embed their ads in popular movies. Such a prediction can also assist cinemas in scheduling movies and help people choose movies to watch.

However, the task is incredibly challenging as a movie's BOR is affected by a lot of complex aspects. Here, we first investigate these influential aspects, coming up with five factors falling under two major stages of the movie-making lifecycle, as illustrated in Fig. 1:

(1) *Production stage*. A movie is made in this stage from an initial story, through screenwriting, casting, and shooting:
   - **Screenwriting**. Most film experts believe that a movie's story is highly predictive of its ultimate BOR performance because the premise of an amazing movie is that there is a wonderful story [1,2].
   - **Casting**. Casting a movie requires teamwork between three types of main participants, i.e., actor/actress (denoted as actor hereafter for simplicity), director, and production company. Their performance is considered to be the most important driver of the movie's BOR [3,4].
   - **Shooting**. A movie is communicated to audiences through the lens. Therefore, the quality of shooting affect the audiences' perception, which impacts the BOR eventually [5,6].

(2) *Distribution stage*. After the production, the movie prepares for releasing in this stage:
   - **Promotion**. As we know, the more people see the movie the higher the BOR. Thus, movie promotion generally involves an advertising campaign by distribution companies to let audiences' know about and to further pique their curiosity about the movie [7].
   - **Schedule**. Selecting the right time to release is also vital for the BOR of movies, e.g., releasing to a period with more holidays and avoiding competition from the same type of movies [8].

* Corresponding author.
E-mail addresses: wang_zhaoyuan@foxmail.com (Z. Wang), msjunbozhang@outlook.com (J. Zhang), shenggongji@163.com (S. Ji), meng.chuishi@jd.com (C. Meng), trli@swjtu.edu.cn (T. Li), msyuzheng@outlook.com (Y. Zheng).

https://doi.org/10.1016/j.inffus.2020.02.002
Received 26 June 2019; Received in revised form 11 December 2019; Accepted 23 February 2020
Available online 24 February 2020

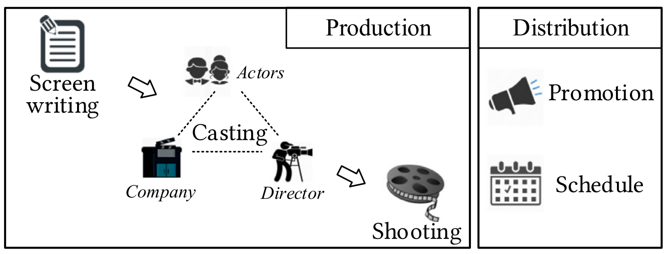

Fig. 1. Influence factors of the BOR.

Initially, the task was considered impossible. Jack Valenti, former CEO of the Motion Picture Association of America, claimed in 1979, “No one can tell you how a movie is going to do in the marketplace, not until the movie opens in darkened theatre and sparks fly up between the screen and the audience” Nevertheless, driven by the apparent financial benefit, researches have been attempting to predict BOR prediction since 1983 [9]. Since then, the issue was broadly studied by econometricians and operations researchers [10–15]. However, these models are usually based on empirical assumptions with hyper-parameters that are not easy to set. Recently, big data in the film domain have become widely available, enabling researchers to solve this challenging problem from a data perspective [7,16–24]. Nevertheless, the performance of these previous methods is fairly limited as they suffer from the following issues.

(1) In general, each method takes into account some of the influencing factors, and there is no method that has considered all the above factors simultaneously (see Section 2 for more details). Thus, it is a challenge and opportunity to consider all of these factors simultaneously, and we may expect a better result.
(2) Casting participants, i.e., actor, director, and production company, are already extensively used in the prediction task. However, most of the prior work handled participants independently by simple hand-crafted features and thus fail to capture dependencies among them [8,25–27]. Instead of an individual participant’s expertise, the cooperation between different participants is essential to determining the casting quality of movies. Therefore, it is important but difficult to measure their cooperation especially when these participants have never worked together.
(3) The existing studies have not fully exploited the potential of trailers due to the high complexity of visual data [5,6,28]. However, as a snippet of the movie, it is very valuable before the movie is released. Proper investigation providing information about the shooting factor from movie trailers is required.

To tackle these issues, in this paper, we propose a novel framework to predict a movie’s BOR, considering and modeling all of the above factors from a wide variety of data. Specifically, the framework consists of a series of feature learning models and a prediction and ranking model. First, we propose a Multi-task dynamIc heterogeneous Network Embedding model (MINE) that learns embeddings for the three types of participants in a unified latent space, capable of capturing their cooperation knowledge in a collective environment. Second, a Deep Neural Network (DNN) is designed to uncover knowledge of the movie shooting factor from trailers. Besides, we propose to employ a topic model to extract knowledge about screenwriting from textual data. Then, we concatenate the above knowledge with several hand-crafted features representing the movie distribution factor as a movie’s feature vector. Finally, a mutually-enhanced prediction and ranking model is trained based on the movie feature vectors to provide ultimate prediction results.

In summary, our contributions include the following:

- We comprehensively investigate influential factors for a movie’s BOR. To the best of our knowledge, this is the first work which takes full consideration of these factors when predicting movies’ BOR.
- We study a novel dynamic heterogeneous network embedding method, i.e., MINE, to handle the casting factor. By introducing dynamic models that capture the temporal correlations of participants, MINE can learn all participants’ representations in different periods collectively with a multi-task paradigm.
- In contrast to only considering simple features about trailers such as the number of plays and audience ratings, we propose to explore the shooting factor from trailer contents by leveraging the power of deep neural networks. Experimental results show that the prediction results using the learned features are significantly better than these simple features.
- We firstly build a set of comprehensive film-related datasets about one of the biggest film markets in the world (i.e., the Chinese film market). These datasets collect movies’ profiles, trailers, stories, as well as the actors’ profiles from many different websites, and we will release them to the general public to encourage future studies. Moreover, the results demonstrate the effectiveness of our framework. We have also deployed the solution as a service for a business group to assist their investment.

The rest of the paper is organized as follows: a brief review of related work is demonstrated in Section 2. Section 3 presents an overview of our data and solution. Section 4 and Section 5 details two specified feature learning models, respectively. The prediction and ranking model will be introduced in Section 6. We evaluate our method in Section 7 followed by the conclusion of this paper in Section 8.

2. Related work

We study several categories of related works, positioning our work in the research community.

2.1. Box office revenue prediction

In recent years, big data reflecting movies have become widely available, enabling us to solve this challenging problem from a data-driven perspective. In this section, we mainly focus on these data-driven studies.

Specifically, these studies can be classified into two groups based on the time point when conducting the prediction, i.e., pre-release and post-release. For the latter, approaches to integrate data about BOR performance, ratings, and word-of-mouth (e.g., reviews from social media) of the initial several days could achieve quite accurate results [16,18–21,23]. Unfortunately, knowing the prediction results at such a late stage is of little value to business people whose money has been spent and who have no time to adjust their investment strategy. The rest of the research was interested in predicting the BOR prior to releasing, and it is in this group that our study falls. As aforementioned, making prediction at this time point is more commercially valuable. Next, we review papers of this group in terms of modeling the influential factors.

(1) *Production factors.*
  - **Screenwriting**. Most film experts believe that a movie’s story is highly predictive of its ultimate BOR performance [1,2]. Motivated by this, literature took advantage of scripts to predict the BOR and confirmed the correlation, e.g., [29,30]. Thus, in this paper, we leverage a topic model to uncover the power of the story data to depict this factor.
  - **Casting**. Participated actors, directors, and companies had been exploited by related work [8,25–27]. Nevertheless, they used independent hand-crafted features, i.e., properties of the three types of participants. This way fails to capture dependencies among them. Clearly, instead of an individual participant’s expertise, the cooperation between different participants determines the casting quality of movies. Later, Parimi et al. introduced a graph-based

method to capture relations between movies, such as sharing with the same actors or directors, alleviating the movie independence assumption [31]. However, the employed homogeneous graph is too simple to host such a sophisticated relationship. Hence, in this paper, we introduce a tri-partite graph to handle the issue and develop a novel embedding method, i.e., MINE to preserve the casting factor into embeddings of participants.

- **Shooting**. The impact of movie trailers on the BOR was investigated by previous research [5,6,28]. However, feature extraction in these methods is typically done by hand. Thanks to the big data era, multimedia data about movies, such as trailers and posters, are publicly available on the Internet. In addition, DNNs have demonstrated their power in understanding these multimedia data. In light of this, Zhou et al. [24] used movie posters as a supplement to predict the BOR employing a CNN configuration. Trailers contain far more information than posters. Hence, to uncover the knowledge to assist in the prediction task, in this paper, we design a DNN to deal with trailers.

(2) *Distribution factors*. Extracting hand-crafted features about the **promotion** factor and the **schedule** factor has also been studied and showed to be fairly accurate [8,22]. Following the work, we also adopt these useful features in our paper.

It is observed that all of these previous studies just considered a part of the influential factors. Instead, in order to get more accurate prediction results, our method properly considers and models all these factors.

2.2. Network embedding

(1) *Static network*. Inspired by the setting of word embeddings [32], DeepWalk [33] pioneered network embedding by considering the node paths traversed by random walks over networks as the sentences and leveraging a skip-gram model for learning node embeddings. With the advent of this work, many network embedding models have been developed, such as LINE [34], node2vec [35], and SDNE [36]. However, these models all paid attention to homogeneous networks. As a newly emerging direction, HIN [37,38] can naturally model complex entities and their rich relations, which brings a new challenge to unify the heterogeneous types of nodes and edges in an embedding space. Recently, several works attempted to analyze HINs via embedding methods [39–44]. However, because these methods are designed for static heterogeneous networks and difficult to generalize to dynamic networks, they fail in our participants learning scenario that is in fact a dynamic heterogeneous network embedding problem. Like these methods, PTE [45] is an effective model in static heterogeneous networks. But, in contrast to them, PTE can be unified into a matrix factorization framework with closed forms [46]. Taking advantage of this characteristic, we propose a novel dynamic heterogeneous network embedding method, i.e., MINE, making PTE work in a dynamic environment.

(2) *Dynamic network*. In real-world applications, it is well recognized that edges between nodes evolve over time in a network [47]. To handle the issue in the field of network embedding, Chen and Tong [48] proposed a fast eigenvalue-tracking algorithm, called TRIP, which can be used to update the solution of the SVD problem when preserving the 1st-order proximity. Goyal et al. used a dynamically expanding deep autoencoder to capture highly nonlinear 1st- and 2nd- order proximities of nodes [49]. Zhu et al. presented a model, denoted as DHPE, which develops an incremental updating method to transform a static high-order network embedding method being dynamic [50]. As previously discussed, the proposed MINE also devote to tackle the dynamic issue, which can be distinguished from the existing methods as the following two aspects. First, these methods are all designed for homogenous networks and incredible to extend to HINs, whereas MINE is studied based on a HIN. Second, these methods incrementally handle the dynamic. In specific, suppose that we have learned the node embeddings before time t, they then investigate solutions to update these node embeddings at time $t + \Delta t$, such that the updated embeddings can reflect the changed network structure. However, we jointly learn all snapshots simultaneously, which is thus better to capture the temporal information globally.

2.3. Video learning

Videos provide more information to the image recognition task by adding a temporal component through which motion and other information can be additionally used. Encouraged by the success of deep CNNs on image classification [51], researches extend the deep networks to video understanding tasks, such as action recognition [52–54] and video classification [55,56]. However, our task differs from these conventional tasks in two aspects. First, trailers' content is more complicated than the used videos in these extensively studied tasks. Second, there is a limited number of movie trailers and no obvious label data for trailers to support us training a monolithic end-to-end network. In light of these issues, we decompose trailers to a series of keyframes, design a concise deep configuration by stacking several pre-trained components to leverage knowledge from these well-studied network structures, and propose to use movie ratings as our training label to assist in understanding these trailers.

2.4. Big data and data fusion

Big data is a general concept, covering a broad range of topics, from big data analytics to data-intensive computing and applications of big data research [57,58]. In this paper, we study an application of big data. That is, predicting BOR from large datasets consisting of multiple modalities, in which trailers are visual data, stories are textual data, and participants are network data.

More specifically, such a multi-modality application can be viewed as a typical data fusion task. Data fusion has been regarded as a new challenge in the big data application research [43,59–61]. According to Zheng [59], data fusion methods can be classified into three categories: stage-based, feature level-based, and semantic meaning-based. Following this line, our framework, from a macroscopic perspective, abides by the stage-based fusion manner, i.e., using different datasets at the different stages. From a mesoscopic perspective, the feature extraction process obtains embeddings for each influential factor and fuses them by the concatenation, which is a feature level-based fusion method. Moreover, the proposed embedding methods in the feature extraction process, such as MINE and the designed deep network, fuse data with the semantic meaning-based way from a microscopic perspective, because we carefully design these methods by understanding the insight of each dataset. Note that the proposed embedding methods also belong to the category of technique research in terms of the big data analytics [47,62–64].

3. Overview

Fig. 2 presents the architecture of our framework. First, data representing different influential factors of a movie are fed to distinct feature learning models to extract features of the corresponding movie. Then, we use these feature vectors to learn a prediction model. Concretely:

Feature extraction. We concatenate the following features as a movie's feature vector.

(1) Production stage. Data reflecting factors in this stage are multimodality. To avoid heavy and tedious manual feature extraction

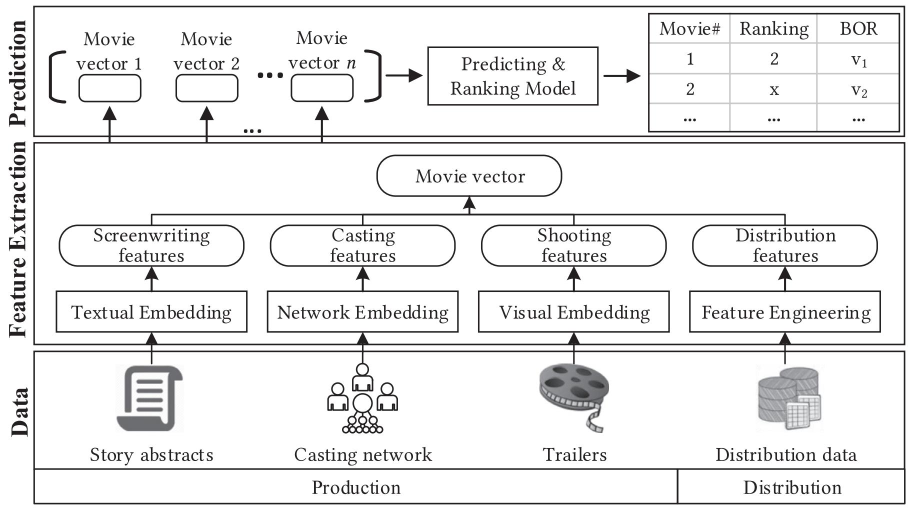

Fig. 2. The architecture of our framework.

and better capture knowledge, we design three learning components, which automatically extract knowledge from the corresponding data.

- *Screenwriting*. It is impossible to get the full script of a movie before it releases. In this paper, we use story abstracts, which introduce the movie's story in a few words, to model the screenwriting factor of a movie. Topic models have shown their effectiveness in document understanding tasks [65]. Hence, we aim to use a topic model to uncover knowledge about the movie's screenwriting from the story abstracts. However, the story abstract data is too short to train a conventional topic model because of the sparsity of word concurrence in a short length document. To address the issue, we propose employing a bi-term-based topic model, called BTM [66]. It learns topics by directly modeling the generation of bi-terms in the corpus, making the inference effective with the rich corpus-level information on short texts. Taking advantage of this model, we denote the topic distribution vector of a story abstract as its movie's screenwriting feature.
- *Casting*. As a movie is cast by the three types of participants, i.e., actor, director, and production company, we construct a participant network to assist us in uncovering the casting factor. On this basis, the goal here is to model the cooperation between participants instead of using hand-crafted features of participants independently. To this end, we aim to investigate a network embedding model to learn participants' embedding collectively. Then, we use these embeddings to generate the casting features. In particular, using these embeddings makes it easy to measure cooperation that has never happened before. However, the task is very challenging. (a) The network of the three types of participants is heterogeneous, which brings difficulties for the embedding learning. (b) The network is dynamic over time with evolution of participants. E.g., an actor may develop herself from a nobody to a rising star, and further to an Oscar award winner gradually by his production. Though their name does not change, they have already become "different persons". Hence, it is reasonable, but very challenging, to assign an embedding in different periods for each participant. Existing work about embedding dynamic network [48,50] or heterogeneous network [39–41] fail to address the two issues simultaneously. To this end, we propose a novel embedding model, i.e., MINE. We will introduce the model in Section 4 specifically.
- *Shooting*. A trailer is a recombination of clips from its original movie, which is available before the movie is released. Thus, in order to decode the shooting factor, we gather trailers of movies and extract the knowledge from the visual content. The DNNs have shown promising results in many computer vision tasks [51,56,67], which motivates us to leverage the technology identifying the shooting factor by projecting a trailer to a vector (i.e., shooting feature). However, we face two main difficulties when using deep networks: (a) trailers' content is more complicated than the videos in traditional video classification tasks [56]; (b) We have a limited number of movie trailers that is far from enough to train monolithic architectures. To overcome the two issues, we design a two-stage network configuration. More specifically, to address the data limitation issue, we first transfer knowledge from a well-known video recognition network to pre-process raw trailers, which thus reduces model complexity dramatically. On this basis, in order to extract higher-level knowledge, we leverage the movie rating to supervise a pairwise paradigm. The network is described in more detail in Section 5.

(2) Distribution stage. Data corresponding to factors in this stage is comprised of a collection of structured data. To include the knowledge in our prediction models, we extract hand-crafted features from the data.

- *Promotion*. For a movie, we first collect the number of fans in social media, historical BORs, ratings, and awards of its participants as well as the movie's IP (Intelligent Property) information, genre, and the sequel, because they are promotion materials revealing the original appeal of the movie. Moreover, the properties of distribution companies for the movie, such as maximum, minimum, and average BOR of movies distributed by the company, are used to identify the ability of the company. Besides, the number of news about the movie in professional film portal websites before released is counted as an indicator of the current promotion effectiveness.
- *Schedule*. We use holiday type, the number of holidays, holiday BOR capacity, capacity growth rate, and other related data to represent a schedule period. In addition, we overlay this with competitor data, e.g., the number of

Table 1
Notations used in the MINE section.

Notations	Descriptions
m	The length of the embedding
n	The length of the sparse embedding
t	The number of snapshots
$\mathcal{A}$	A set of actors
$\mathcal{D}$	A set of directors
$\mathcal{C}$	A set of companies
$\mathbf{A}_i \in \mathbb{R}^{m\times\lvert\mathcal{A}\rvert}$	Actor embedding matrix in the snapshot i
$\mathbf{D}_i \in \mathbb{R}^{m\times\lvert\mathcal{D}\rvert}$	Director embedding matrix in the snapshot i
$\mathbf{C}_i \in \mathbb{R}^{m\times\lvert\mathcal{C}\rvert}$	Company embedding matrix in the snapshot i
$\mathbf{A}_i^{'} \in \mathbb{R}^{m\times\lvert\mathcal{A}\rvert}$	Actor sparse embedding matrix in the snapshot i
$\mathbf{D}_i^{'} \in \mathbb{R}^{m\times\lvert\mathcal{D}\rvert}$	Director sparse embedding matrix in the snapshot i
$\mathbf{C}_i^{'} \in \mathbb{R}^{m\times\lvert\mathcal{C}\rvert}$	Company sparse embedding matrix in the snapshot i
$\mathbf{B}^{\mathcal{A}} \in \mathbb{R}^{m\times n}$	Base (dictionary) matrix for the actor
$\mathbf{B}^{\mathcal{D}} \in \mathbb{R}^{m\times n}$	Base (dictionary) matrix for the director
$\mathbf{B}^{\mathcal{C}} \in \mathbb{R}^{m\times n}$	Base (dictionary) matrix for the company

competitors and the number of competitors with the same genre.

Prediction model. We propose learning a prediction and ranking model to take advantage of these movie feature vectors and uncover the BOR of movies. We employ this model since: (1) a linear predictor is equipped in the model requires very few parameters and has shown its effectiveness in many learning-to-rank algorithms [68]. (2) This model utilizes the ranking information. On the one hand, the ranking information helps make decisions for business people; on the other hand, the predicting and ranking objectives can also be enhanced mutually. (3) A feature selection mechanism is also conducted simultaneously in this model to eliminate redundant features. Details of the model can be found in Section 6.

4. Dynamic heterogeneous network embedding model

To model the relationship among participants' cooperation, i.e., actors, directors, and production companies, we propose a novel embedding model, i.e., MINE. In this section, we first formalize the dynamic heterogeneous network embedding problem. Then, we describe the MINE for tackling the problem in detail.

4.1. Preliminaries

Table 1 lists frequently-used notations in the section. Note that we use lowercase bold letters, such as $\mathbf{a}$, to represent vectors. Data matrices are written in uppercase bold letters, such as $\mathbf{X}$. Calligraphic letters, such as $\mathcal{A}$, are used to represent sets. In addition, $\mathbf{X}_i$ indicates the matrix of the time slot i, $\mathbf{X}_{(jd)}$ represents the jth row and kth column of the matrix, and $\mathbf{B}^{\mathcal{A}}$ denotes the matrix for the type of set $\mathcal{A}$. Finally, we adopt $\mathbb{R}$ and $\mathbb{R}_+$ to denote the set of real numbers and non-negative real numbers, respectively.

4.1.1. Graph construction

As pervious mentioned, the relationship between participants can be represented by a network. Here, we show how we construct this network.

We propose to use a tripartite graph [37,38] to host participants' cooperation relationship, which is composed of two sub-networks, i.e., an actor-director network and an actor-company network, as shown in Fig. 3 (a). This is because the cooperation between the actor and the director and the cooperation between the actor and the company are two main interactions, whereas the director and the company have no frequent cooperation patterns. In addition, the graph is dynamic as their cooperation relationship is time-evolving. In order to easily generate different embeddings at different timepoints for a participant, we denote the dynamic network as a sequence of snapshots. Thus, we further demonstrate how to model the dynamics. In other words, the calculated edge weights of these snapshots vary according to newly added movies.

More specifically, to identify the cooperation between participants, we define a score, as shown in Eq. (1), to determine the edge weight of participants according to the quality of their cooperated movies before the timestamp of this snapshot. Take the first snapshot of $\mathcal{G}$ as an example. Suppose $\mathcal{M}_{jk} = \{m_1, m_2, \dots, m_{|\mathcal{M}_{jk}|}\}$ is a set of movies cooperated by participants j and k before this snapshot, their edge weight e_{jk} can be calculated by Eq. (1):

$$e_{jk} \equiv w_1 \sigma\left(\frac{1}{2}\sum_{z=1}^{|\mathcal{M}_{jk}|}(b_z + r_z)\right) + w_2 \max\left(\mathcal{B}_{jk}\right) + w_3 \max\left(\mathcal{R}_{jk}\right)$$
$$s.t.\ w_1 + w_2 + w_3 = 1 \tag{1}$$

where b_z and r_z is the BOR ranking ratio and rating of movie m_z, respectively, $\mathcal{B}_{jk} = \{b_1, b_2, \dots, b_{|\mathcal{M}_{jk}|}\}$, and $\mathcal{R}_{jk} = \{r_1, r_2, \dots, r_{|\mathcal{M}_{jk}|}\}$. Besides, $\sigma()$ is the Sigmoid function, function $\max()$ returns the maximum value among the given set, and parameters w_1, w_2, and w_3 are used to adjust the importance of each part (we set all of them equal to 1/3). Concretely, as the real BOR value is impacted by the market environment, we use the ranking of the BOR instead of the value directly, $b_z = \exp^{-\rho(q-u)} * (1 - (rank_z/tot_u))$ where u is the snapshot when m_z is released, tot_u is the total number of movies released around the snapshot, $rank_z$ is the BOR ranking of m_z among tot_u movies. $\exp^{-\rho(q-u)}$ is a time-aware weighted function where q is the current snapshot to weaken old cooperation and highlight new cooperation since the effect of a movie decays as time goes on.

In summary, the intuition of the score is to consider both long-term cooperation (the first part) and greatest cooperation (the last two parts) since cooperation quantity is vital, but cooperation quality is more important where moviegoers always remember those excellent movies. For the next snapshot, as the cooperated movies are updated, we recalculate each edge weight according to Eq. (1) using a new movie set. At last, we generate t snapshots of $\mathcal{G}$ in this way.

4.1.2. Problem definition of the participant embedding

Definition 1 Participant Embedding Problem. Given a dynamic graph $\mathcal{G}$ including t snapshots, the task is to learn the m-dimensional latent embeddings $\{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_t\}$, $\{\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_t\}$, and $\{\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_t\}$ for all snapshots collectively and simultaneously, preserving cooperation relationships between the three types of participants.

4.2. MINE

4.2.1. Model overview

In order to tackle the defined problem, we propose a novel model, named MINE. The main idea of MINE is demonstrated in Fig. 3 (b), in which each plate is represented as a snapshot.

In each snapshot, a static module captures the cooperation relationship into participants' embeddings. Meanwhile, participants also have temporal dependencies. This is captured by a global correlation module among snapshots, which jointly extracts the common characteristics for each specific type of participant. We also learn a local smoothness module for two arbitrary adjacent snapshots, capturing the local dynamic information. Concretely, participants who have no new movies between two adjacent snapshots should have their representations change smoothly. Notably, the two modules mean our model essentially follows a multi-task learning paradigm such that participants from all snapshots are projected into a unified latent space, which is capable of helping us better capture the cooperation relationship by utilizing the knowledge between tasks.

In summary, we give a brief formulation of the MINE as Eq. (2), including three parts according to the above three modules, respectively. We will detail them subsequently.

$$\mathcal{L} = \mathcal{L}^{static} + \mathcal{L}^{dynamic}_{global} + \mathcal{L}^{dynamic}_{local} \tag{2}$$

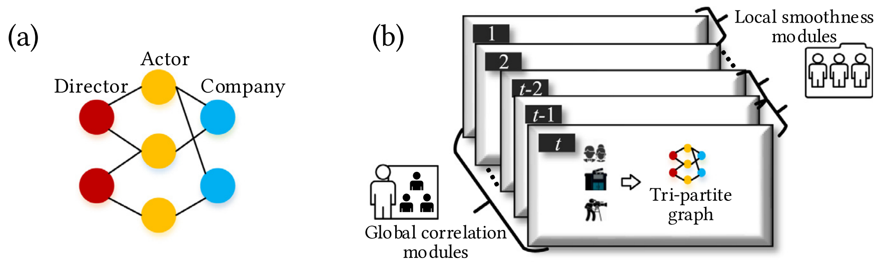

Fig. 3. (a) Tri-partite graph of participants; (b) The main idea of MINE model .

4.2.2. *Static module*

For a task (snapshot) i, having such a tripartite graph with rich information about participants' cooperation relationship, we aim to preserve the information in participants' embedding. To this end, we can adopt PTE [45], an effective extension of LINE (2nd) in heterogeneous networks that can capture both network structures and semantic information in embeddings. In fact, the cooperation relationship can be represented by the two-fold information. This is because the structures represent whether they have cooperated and the semantic information stands for the quality of the cooperation.

In addition, in order to apply this algorithm to our framework, we adopt a matrix factorization based PTE model as [46] proved that the algorithm can be unified into a matrix factorization framework with closed forms. Eq. (3) shows the matrix that implicitly approximates PTE in our scenario.

$$\mathbf{X}_i = \log\left(\begin{bmatrix} \alpha \mathrm{vol}\left(\mathbf{G}_i^{\mathcal{DA}}\right)\left(\mathbf{M}_{i_{row}}^{\mathcal{DA}}\right)^{-1}\mathbf{S}_i^{\mathcal{DA}}\left(\mathbf{M}_{i_{col}}^{\mathcal{DA}}\right)^{-1} \\ \beta \mathrm{vol}\left(\mathbf{G}_i^{\mathcal{CA}}\right)\left(\mathbf{M}_{i_{row}}^{\mathcal{CA}}\right)^{-1}\mathbf{S}_i^{\mathcal{CA}}\left(\mathbf{M}_{i_{col}}^{\mathcal{CA}}\right)^{-1} \end{bmatrix}\right) - \log b \tag{3}$$

Take bipartite graph $\mathbf{G}_i^{\mathcal{DA}}$ as an example, we use $\mathbf{S}_i^{\mathcal{DA}} \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{A}|}$ to denote its adjacency matrix, and use $\mathbf{M}_{i_{row}}^{\mathcal{DA}}$ and $\mathbf{M}_{i_{col}}^{\mathcal{DA}}$ to denote its diagonal matrices with row and column sum, respectively. In addition, $\mathrm{vol}(\mathbf{G}_i^{\mathcal{DA}}) = \sum_j \sum_k \mathbf{S}_{i_{(jk)}}^{\mathcal{DA}}$ is the volume of graph $\mathbf{G}_i^{\mathcal{DA}}$.

As Eq. (3) is computationally complex, following Yang et al. [69], we factorize the matrix $\hat{\mathbf{X}}_i$ shown in Eq. (4) instead of $\mathbf{X}_i$. This is because $\mathbf{X}_i$ has much more non-zero entries than $\hat{\mathbf{X}}_i$ and the complexity of matrix factorization is proportional to the number of non-zero entries.

$$\hat{\mathbf{X}}_i = \begin{bmatrix} \alpha \mathrm{vol}\left(\mathbf{G}_i^{\mathcal{DA}}\right)\left(\mathbf{M}_{i_{row}}^{\mathcal{DA}}\right)^{-1}\mathbf{S}_i^{\mathcal{DA}}\left(\mathbf{M}_{i_{col}}^{\mathcal{DA}}\right)^{-1} \\ \beta \mathrm{vol}\left(\mathbf{G}_i^{\mathcal{CA}}\right)\left(\mathbf{M}_{i_{row}}^{\mathcal{CA}}\right)^{-1}\mathbf{S}_i^{\mathcal{CA}}\left(\mathbf{M}_{i_{col}}^{\mathcal{CA}}\right)^{-1} \end{bmatrix} \tag{4}$$

With this information in hand, we then formalize the static module for a single task as:

$$\mathcal{L}_{(i)}^{static} = \|\hat{\mathbf{X}}_i - [\mathbf{D}_i, \mathbf{C}_i]^{\mathrm{T}}\mathbf{A}_i\|_F^2 \tag{5}$$

where $\|\cdot\|_F^2$ denotes the Frobenius norm. By means of this, we encode the cooperation information of participants into their embeddings under a collective environment.

As a result, $\mathcal{L}^{static} = \sum_{i=1}^{t} \mathcal{L}_{(i)}^{static}$ for all tasks.

4.2.3. *Global correlation module*

A basic assumption of this module is that participants of the same type should have common characteristics. For instance, there may be a template for actors that includes their performance skills. An arbitrary actor is thus a combination of these skills from the template. Knowing the template can assist us in obtaining a more accurate embedding. Inspired by the sparse coding technology [70], the module intends to uncover type-based high-level latent features by a global dictionary (or base) overall tasks.

Thus, we model this module on task i as:

$$\mathcal{L}_{global_{(i)}}^{dynamic} = \left\|\mathbf{A}_i - \mathbf{B}^{\mathcal{A}}\mathbf{A}_i^{'}\right\|_F^2 + \left\|\mathbf{D}_i - \mathbf{B}^{\mathcal{D}}\mathbf{D}_i^{'}\right\|_F^2 + \left\|\mathbf{C}_i - \mathbf{B}^{\mathcal{C}}\mathbf{C}_i^{'}\right\|_F^2 \tag{6}$$

where $\mathbf{B}^{\mathcal{A}}$, $\mathbf{B}^{\mathcal{D}}$, and $\mathbf{B}^{\mathcal{C}}$ are high-level bases (i.e., templates) for each type of participants shared by all tasks, $\mathbf{A}_i^{'}$, $\mathbf{D}_i^{'}$, and $\mathbf{C}_i^{'}$ are the sparse representation for each participant. Finally, $\mathcal{L}_{global}^{dynamic} = \sum_{i=1}^{t} \mathcal{L}_{global_{(i)}}^{dynamic}$ for all tasks.

4.2.4. *Local smoothness module*

It is reasonable that the state of a participant who participates in new movies from the former snapshot to the latter should be changed. In contrast, for the rest of the participants, their state should be as stable as possible. Hence, this module aims to penalize disagreements for these participants. Supplemented by the local temporal knowledge, the embedding is able to become more reasonable. Following this idea, for two arbitrary adjacent tasks, the module is formulated as:

$$\begin{aligned}\mathcal{L}_{local_{(i)}}^{dynamic} &= \left\|\mathbf{I}_i^{\mathcal{A}} \odot \left(\mathbf{A}_i - \mathbf{A}_{i-1}\right)\right\|_F^2 + \left\|\mathbf{I}_i^{\mathcal{D}} \odot \left(\mathbf{D}_i - \mathbf{D}_{i-1}\right)\right\|_F^2 \\ &+ \left\|\mathbf{I}_i^{\mathcal{C}} \odot \left(\mathbf{C}_i - \mathbf{C}_{i-1}\right)\right\|_F^2\end{aligned} \tag{7}$$

where $\odot$ denotes the Hadamard product. $\mathbf{I}_i^{\mathcal{A}}$, $\mathbf{I}_i^{\mathcal{D}}$, and $\mathbf{I}_i^{\mathcal{C}}$ are identity matrices, indicating if a participant does not produce new movies in the snapshot, e.g., $\mathbf{I}_{i(j)}^{\mathcal{A}} = 1$, if actor j cast a new movie from snapshot $i-1$ to i; $\mathbf{I}_{i(j)}^{\mathcal{A}} = 0$, otherwise. As a result, $\mathcal{L}_{local}^{dynamic} = \sum_{i=2}^{t} \mathcal{L}_{local_{(i)}}^{dynamic}$ for any two adjacent tasks.

Note that, by means of the two dynamic modules, our framework can be seen as a multi-task model. Specifically, according to [71], multi-task learning is to learn these related tasks simultaneously by extracting and utilizing appropriate shared information across tasks. By leveraging the global correlation module and the local smoothness module, an embedding coordinate cross all snapshots is constructed, projecting participants from all snapshots into a unified latent space. Therefore, the two modules make our model essentially follow a multi-task learning paradigm. That is, we can treat each snapshot as a task and regard the coordinate as the shared information across tasks. By learning all these tasks simultaneously, we can better capture the cooperation relationship of participants as the utilization of the temporal knowledge between tasks.

4.2.5. *Model inference*

By combining all the aforementioned three modules, the objective of MINE in Eq. (2) can be formulated as Eq. (8), where $\mathcal{I}^{\mathcal{A}}$, $\mathcal{I}^{\mathcal{D}}$, and $\mathcal{I}^{\mathcal{C}}$ are three sets of identity matrices, e.g., $\mathcal{I}^{\mathcal{A}} = \{\mathbf{I}_2^{\mathcal{A}}, \mathbf{I}_3^{\mathcal{A}}, \dots, \mathbf{I}_t^{\mathcal{A}}\}$. δ, μ, ϵ, τ, and ρ are hyper-parameters. The first three components are the three modules above, respectively. The subsequent component is to prevent our embedding overfitting by the L2 penalty. Note that the next component is the L1 penalty for these sparse representations and the global bases are subjected to a small constant to ensure the effectiveness of the L1 penalty.

$$\begin{aligned}&\mathcal{L}\left(\mathcal{X}, \mathcal{I}^{\mathcal{A}}, \mathcal{I}^{\mathcal{D}}, \mathcal{I}^{\mathcal{C}}, \delta, \mu, \epsilon, \tau, \rho\right) \\ &= \frac{1}{2}\sum_{i=1}^{t}\left\|\hat{\mathbf{X}}_i - [\mathbf{D}_i, \mathbf{C}_i]^{\mathrm{T}}\mathbf{A}_i\right\|_F^2 \\ &+ \frac{\delta}{2}\sum_{i=1}^{t}\left(\left\|\mathbf{A}_i - \mathbf{B}^{\mathcal{A}}\mathbf{A}_i^{'}\right\|_F^2 + \left\|\mathbf{D}_i - \mathbf{B}^{\mathcal{D}}\mathbf{D}_i^{'}\right\|_F^2 + \left\|\mathbf{C}_i - \mathbf{B}^{\mathcal{C}}\mathbf{C}_i^{'}\right\|_F^2\right)\end{aligned}$$

$$+\frac{\mu}{2}\sum_{i=2}^{t}\left(\left\|\mathbf{I}_i^{\mathcal{A}}\odot(\mathbf{A}_i-\mathbf{A}_{i-1})\right\|_F^2+\left\|\mathbf{I}_i^{\mathcal{D}}\odot(\mathbf{D}_i-\mathbf{D}_{i-1})\right\|_F^2+\left\|\mathbf{I}_i^{\mathcal{C}}\odot(\mathbf{C}_i-\mathbf{C}_{i-1})\right\|_F^2\right)$$
$$+\frac{\epsilon}{2}\sum_{i=1}^{t}\left(\|\mathbf{A}_i\|_F^2+\|\mathbf{D}_i\|_F^2+\|\mathbf{C}_i\|_F^2\right)$$
$$+\tau\sum_{i=1}^{t}\left(\sum_{j=0}^{\mathcal{A}}\left\|\mathbf{a}'_{i(j)}\right\|_1+\sum_{j=0}^{\mathcal{D}}\left\|\mathbf{d}'_{i(j)}\right\|_1+\sum_{j=0}^{\mathcal{C}}\left\|\mathbf{c}'_{i(j)}\right\|_1\right)$$
$$s.t.\ \sum_j(\mathbf{B}^{\mathcal{A}})_{j,k}^2\le\rho,\ \sum_j(\mathbf{B}^{\mathcal{D}})_{j,k}^2\le\rho,\ \sum_j(\mathbf{B}^{\mathcal{C}})_{j,k}^2\le\rho;\ \forall k=1,\dots,n \tag{8}$$

In general, the objective function is not jointly convex to all the variables but is convex for every single variable. Thus, we separate the optimization to several subproblems and utilize the block coordinate descent approach [72] to solve this problem. Namely, we iteratively update the values of one set of the variable while fixing the values of others until convergence. The inference algorithm is presented in Algorithm 1. Specifically, **A, D** or **C** can be easily optimized by gradient descent. In addition, when solving the bases **B**, the problem is actually an L2-constrained least squares problem, so we adopt the well-known Lagrange dual algorithm [70] to address it in each round. Optimizing the sparse representation, i.e., $\mathbf{A}'$, $\mathbf{D}'$, and $\mathbf{C}'$, is a typical L1-regularized least squares problem. An efficient algorithm is used to tackle the problem in each iteration, denoted as the feature-sign search algorithm [70].

Algorithm 1: Inference algorithm for the MINE model.

```
Set      : $\mathcal{X}, \mathcal{I}^{\mathcal{A}}, \mathcal{I}^{\mathcal{D}}, \mathcal{I}^{\mathcal{C}}, \delta, \mu, \epsilon, \tau, \rho, \psi$
Initiate: Initiate $\{\mathbf{A}_1,\mathbf{A}_2,\dots,\mathbf{A}_t\}$, $\{\mathbf{D}_1,\mathbf{D}_2,\dots,\mathbf{D}_t\}$,
          $\{\mathbf{C}_1,\mathbf{C}_2,\dots,\mathbf{C}_t\}$, $\mathbf{B}^{\mathcal{A}}$, $\mathbf{B}^{\mathcal{D}}$, $\mathbf{B}^{\mathcal{C}}$, $\{\mathbf{A}'_1,\mathbf{A}'_2,\dots,\mathbf{A}'_t\}$,
          $\{\mathbf{D}'_1,\mathbf{D}'_2,\dots,\mathbf{D}'_t\}$, and $\{\mathbf{C}'_1,\mathbf{C}'_2,\dots,\mathbf{C}'_t\}$ with small
          random values
1 Let $i := 1$
2 do
3     for snapshot $i$ do
4         Update $\mathbf{A}_i$, $\mathbf{D}_i$, and $\mathbf{C}_i$ based on gradient descent,
          respectively
5         Update $\mathbf{A}'_i$, $\mathbf{D}'_i$, and $\mathbf{C}'_i$ based on the feature-sign
          search algorithm, respectively
6         Update $\mathbf{B}^{\mathcal{A}}$, $\mathbf{B}^{\mathcal{D}}$, and $\mathbf{B}^{\mathcal{C}}$ based on the Lagrange dual
          algorithm, respectively
7     $i = i + 1$
8 until $|Loss_g - Loss_{g+1}| > \psi$
Return : embeddings $\{\mathbf{A}_1,\mathbf{A}_2,\dots,\mathbf{A}_t\}$, $\{\mathbf{D}_1,\mathbf{D}_2,\dots,\mathbf{D}_t\}$,
         $\{\mathbf{C}_1,\mathbf{C}_2,\dots,\mathbf{C}_t\}$
```

Computational complexity. Here, we analyze the complexity of MINE. According to the above updating rules, for a static snapshot, the time complexity is roughly $O(mn)+O(ns)+O(mk)$ per vector, where s is the "sparsity" of the encoding (number of non-zero entries) and k is the average number of edges in the network. As a result, the efficiency of the dynamic model is about $O(tn(mn+ns+mk))$, where t is the number of snapshots and n is the number of nodes in a network.

5. Visual embedding model

In order to extract knowledge about the movie shooting, we design an emerging network configuration. In this section, we first introduce the trailer embedding problem. Then, the main idea of designing the network is described. Finally, we illustrate how to train the network.

5.1. Preliminaries

Table 2 lists frequently-used notations in this section.

Table 2
Notations used in the visual embedding section.

Notations	Descriptions
$\mathbf{K}$	a keyframe (image)
$\mathcal{T}^i=\{\mathbf{K}_1^i,\mathbf{K}_2^i,\dots,\mathbf{K}_l^i\}$	a trailer i containing l keyframes
$\mathbf{m}^i\in\mathbb{R}^l$	a l-dimension middle-level embeddings of trailer i
$\mathbf{x}^i\in\mathbb{R}^n$	a n-dimension high-level embeddings of trailer i

Definition 2 Trailer Embedding Problem. Given t trailers, $\{\mathcal{T}_1,\mathcal{T}_2,\dots,\mathcal{T}_t\}$, the task is to learn the n-dimensional latent embeddings $\{\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_t\}$ with high-level knowledge about the shooting factor.

5.2. Intuition of our method

We have identified two key challenges to tackle the problem, which motivate a two-stage network configuration design, as shown in Fig. 4(a). Specifically, processing raw trailers is computationally demanding since each video contains numerous frames. Meanwhile, we do not have enough training data (i.e., trailers) for this task. With the help of extensive data in the image and video domain, we first train a network designed for a video classification task [55], using stacked Convolutional Neural Networks (CNNs) [51,73] and Long-Short Term Memory networks (LSTMs) [74,75]. Then, we fix the network, entitled the trailer embedding extraction network, and feed raw trailers to the network to gain compact trailer embeddings. These embeddings about appearance and motion features can be seen as middle-level embeddings for trailers. Therefore, in order to further learn high-level features about the shooting factor (shooting quality and even plotline) over the middle-level features, a pairwise Siamese network [76] is learned by comparing two ratings of the two trailers' original movie. Our ratings are scored by audiences according to the picture quality and plotline of a movie and regardless of its shooting styles, capable of impacting movies' BOR [8,25,26,31]. Thus, to capture high-level features about the shooting quality and even plotline from the shooting styles (i.e., the middle-level knowledge), we propose to use the information to supervise the network. Notably, the network best suits our scenario where training data are limited [76]. Therefore, we address the two challenges above and obtain a concise vector with high-level knowledge for each trailer (i.e., the dense vector of the last fully connected layer, e.g., "FC3" in Fig. 4(a)).

5.3. Training and inference

As illustrated in Fig. 4(b), we first feed a trailer $\mathcal{T}^i$ to the trailer embedding extraction network. Features of keyframes are extracted by a weight shared CNN network, i.e. GoogLeNet [77], which the network is simple and with distinguished feature diversity characteristics. Then the frame-level information is combined, using the LSTM units, allowing the discovery of long-range temporal relationships. Following an LSTM layer, a Softmax classifier makes a classification at every frame in the original task. We obtain $\mathbf{m}^i$ by an element-wise multiplication among each hidden state $\mathbf{h}_j^i$ of the corresponding jth LSTM unit as the middle-level trailer embedding. The element-wise multiplication (a.k.a. Hadamard-Product) is a useful knowledge fusion operation [78,79]. In contrast to taking an average of multiple vectors (the most common fusion operation), the element-wise multiplication can make feature selection to some extent. Namely, it can amplify the redundant information. In our scenario, different keyframes bring much information. However, for a trailer, useful knowledge should be the information that often appears in pictures, e.g., symbols, colors, and characters. Consequently, we use the element-wise multiplication to identify such information. Note that the GoogLeNet is initialized from a pre-trained ImageNet [51] model and then can be fine-tuned over a large video dataset, such as UCF-101 [80]. Specifically, we can add two dense layers over the

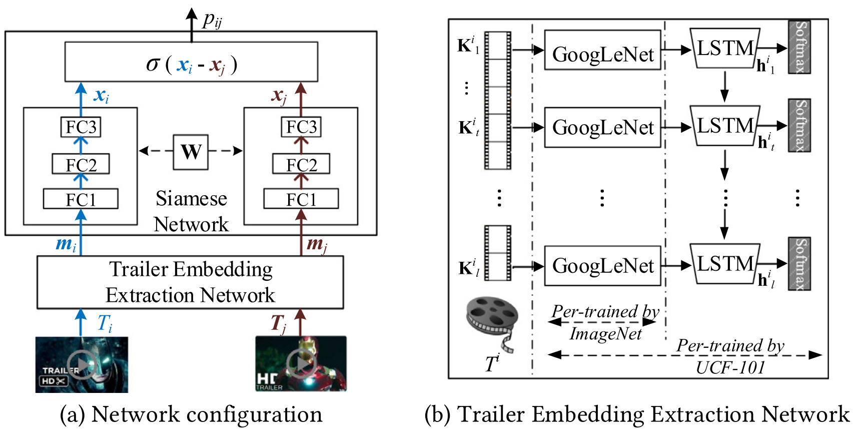

Fig. 4. The trailer embedding model.

trailer embedding extraction network, which becomes an effective network structure for visual recognition, as presented in [54]. Concretely, the last dense layer has 101 dimensions, providing the action classification results w.r.t. the 101 categories. In addition, the other dense layer with 256 dimensions is used to bridge the trailer embedding extraction network and the output layer. Based on the structure, we can fine-tune parameters of the trailer embedding extraction network (including these GoogLeNets) by the supervised visual recognition task using UCF-101 data. Through the way, we can leverage the existing big data to alleviate the issue that we do not have enough trailers to train a monolithic architecture end-to-end.

The Siamese network is a weight shared pairwise paradigm. We feed the learned trailer embedding into the network in pairs. The network maps each input middle-level embedding $\mathbf{m}^i$ to a high-level $\mathbf{x}^i$. The two outputs of the model are mapped to a learned probability that rating of trailer i should be higher than that of trailer j via the Sigmoid function, i.e., $p_{ij} = 1/(1+\exp^{-(\mathbf{x}^i-\mathbf{x}^j)})$. We then apply the cross-entropy cost function, which penalizes the deviation of the model output probabilities from the desired probabilities. Let $\bar{p}_{ij}$ be the known probability that the rating of trailer i should be higher than that of trailer j (Suppose $\bar{r}^i$ and $\bar{r}^j$ are the ratings of trailers i and j respectively. The groundtruth $\bar{p}_{ij}$ is also calculated by the Sigmoid function, i.e., $\bar{p}_{ij} = 1/(1+\exp^{-(\bar{r}^i-\bar{r}^j)})$). Then the cost function is:

$$\mathcal{L}_{Siamese} = -\bar{p}_{ij}\log p_{ij} - \left(1-\bar{p}_{ij}\right)\log\left(1-p_{ij}\right) \tag{9}$$

The stochastic gradient descent algorithm is utilized to update the parameters of the network. More details about the network can be found in [81], which employs the Siamese paradigm for a ranking task, denoted as RankNet.

6. Prediction and ranking model

In this section, we introduce the prediction and ranking model, which uncovers the BOR of movies using the movie feature vectors.

6.1. Preliminaries

Through the feature extraction stage, we can obtain a feature vector for each movie, including casting features, shooting features, screenwriting features, and handcrafted features about distribution factors. Mathematically, let $\mathbf{z}_i \in \mathbb{R}^d$ denote a d-dimension feature vector of movie m_i. y_i and $\hat{y}_i$ represents real and predicted BOR value of m_i, respectively, while r_i and $\hat{r}_i$ represents real and predicted ranking of m_i in a set of movies, respectively.

Definition 3 (BOR Prediction and Ranking Problem)**.** The task is to train a model with movie feature vectors $\mathcal{Z}^{tr} = \{\mathbf{z}^{tr}_1, \mathbf{z}^{tr}_2, \ldots, \mathbf{z}^{tr}_m\}$, real BOR values $\mathcal{Y}^{tr} = \{y^{tr}_1, y^{tr}_2, \ldots, y^{tr}_n\}$, and rankings $\mathcal{R}^{tr} = \{r^{tr}_1, r^{tr}_2, \ldots, r^{tr}_m\}$, to learn a model. Then, it can be used to obtain predicted BOR values $\hat{\mathcal{Y}}^{te} = \{\hat{y}^{te}_1, \hat{y}^{te}_2, \ldots, \hat{y}^{te}_n\}$ and predicted ranking $\hat{\mathcal{R}}^{te} = \{\hat{r}^{te}_1, \hat{r}^{te}_2, \ldots, \hat{r}^{te}_n\}$, given n testing movies $\mathcal{Z}^{te} = \{\mathbf{z}^{te}_1, \mathbf{z}^{te}_2, \ldots, \mathbf{z}^{te}_n\}$.

In addition, a linear predictor is denoted as $f()$ projecting $\mathbf{z}_i$ to be close to the y_i. Namely, $f(\mathbf{z}_i;\mathbf{w}) = \mathbf{w}^T\mathbf{z}_i + \epsilon_i$ where ϵ_i is a zero-mean Gaussian bias with variance σ^2 and $\mathbf{w}$ is the weights of the vector. For simplicity, we denote $f(\mathbf{z}_i;\ \mathbf{w})$ as f_i. Note that, the predictor can be rewritten as $P(y_i|\mathbf{z}_i) = \mathcal{N}(y_i|f_i, \sigma^2) = \mathcal{N}(y_i|\mathbf{w}^T\mathbf{z}_i, \sigma^2)$ from a Bayesian perspective, where $\mathcal{N}$ represents the normal distribution.

6.2. Model details

In this paper, we propose using a prediction and ranking model, which combines a linear prediction objective, a pairwise ranking objective, and a sparsity regularizer in a unified probabilistic paradigm. Here, we detail how to train the model. Specifically:

Given $\mathcal{Z}^{training}$, by Bayesian inference, we have the posterior probability of the model as:

$$Pr(\Psi, \mathcal{D}, \Omega) = P(\mathcal{D}|\Psi, \Omega)P(\Psi|\Omega) \tag{10}$$

where $\Psi = \{\mathbf{v}, \beta^2\}$ denotes parameters of the model. $\Omega = \{a, b, \sigma^2\}$ are hyper-parameters of the sparsity regularization. $\mathcal{D} = \{\mathcal{Y}^{training}, \mathcal{R}^{training}\}$ represents the observed data.

The first term $P(\mathcal{D}|\Psi, \Omega)$ is the likelihood of the observed data D, i.e. the predictor and the ranker. It can be explained as:

$$\begin{aligned} P(\mathcal{D}|\Psi, \Omega) &= P(Y|\Psi, \Omega)P(\Pi|\Psi, \Omega) \\ &= \prod_{i=1}^{m}\mathcal{N}\left(y_i|f_i, \sigma^2\right)\prod_{i=1}^{m-1}\prod_{h=i+1}^{m}P(i\to h|\Psi, \Omega) \end{aligned} \tag{11}$$

where $P(Y|\Psi, \Omega)$ is the linear predictor. $P(\Pi|\Psi, \Omega)$ minimizes the ranking of movie m_i compared with each movie in the training set, in which $i \to h$ defines movie m_i is ranked higher than movie m_h measured by the Sigmoid function $P(i\to h) = 1/(1+\exp^{-(f_i - f_h)})$. As a result, we optimize the prediction and ranking objectives simultaneously by this term in a mutually enhanced manner.

The last term $P(\Psi|\Omega)$ is a sparse weight prior distribution to conduct the feature selection and avoiding overfitting:

$$P(\Psi|\Omega) = P\left(\mathbf{v}|0, \beta^2\right)P\left(\beta^2|a, b\right) = \prod_{i=1}^{m}\mathcal{N}\left(v_i|0, \beta^2\right)\prod_{i=1}^{m}\mathcal{IG}(\beta_i^2|a, b) \tag{12}$$

where β^2 is a separate variance parameter vector for $\mathbf{v}$, $\mathcal{IG}$ stands for the Inverse Gamma distribution that is a prior distribution for the β^2. Among that, the a and b are constants and usually set close to zero. It enforces sparse representations during learning by setting some feature weights to zero.

The model can be easily optimized by a stochastic gradient descent algorithm. For more details about optimization, please refer to [68].

Having the trained model, we can leverage it to obtain $\hat{\mathcal{Y}}^{te}$ and $\hat{\mathcal{R}}^{te}$ when given $\mathcal{Z}^{te}$.

7. Results and discussion

7.1. Settings

7.1.1. Dataset

As introduced, to evaluate our solution, we collected a set of comprehensive movie data about the Chinese film market from Mtime.com,[1] a Chinese version of IMDb.[2]

Specifically, since the Chinese film market was established in 2008, we collected about 2500 movies from 2008 to 2017, including domestic and Hollywood movies released in the market with their BORs and data about the aforementioned influential factors. Moreover, to facilitate the feature learning tasks, we also supplement the following data:

- A large movie participant database is built to have a real community of participants. Specifically, the database contains about 21,000 movie and TV entries, around 1,000,000 people entries who are actors and directors, and 85,000 company entries.
- Besides trailers of the 2500 movies, we collect about 2000 additional trailers. Note that we have ratings for all these movies, which were rated by Chinese audiences. In order to assist in learning these trailers, we draw support from a well-known large-scale video dataset UCF-101 [80].
- In addition to story abstracts of the 2500 movies, we gather another 50,000 story abstracts for training the BTM model.

7.1.2. Configuration

Features extracted by these feature learning models are configured as follows.

- *MINE*. We train two models for domestic and Hollywood movies respectively since they belong to two disclosed communities. The step between snapshots is set to one quarter. Embeddings in a snapshot are employed to generate the movie's participant features in the next snapshot. In addition, we use the mode of embedding and inner production between two embeddings (except the inner production between directors and companies, as they have no frequent cooperation patterns) to be our features, denoting their own and cooperation values. Note that we select three actors, one director, and two companies as participants for a movie (We will explain the reason and validate in Section 7.3). As a result, we have 19 dimensions of casting features for each movie.
- *Deep network*. We set the dimension of output embedding, i.e., "FC3", for the Siamese network as 16. Therefore, the shooting feature of a movie consists of 16 dimensions. (Other settings of the deep network will be determined by experiments, as shown in Section 7.3)
- *BTM model*. The story abstracts are in Chinese. We first handle these Chinese sentences, the string of characters without spaces, to word segmentation by using a wild open tool named Jieba.[3] We set the topic number to 20 for the BTM model. Thus, the screenwriting feature of a movie contains 20 dimensions.

For a movie, we concatenate the three features above with another 26 hand-crafted features about the distribution factor. As a consequence, the movie's feature vector has 81 dimensions in total.

7.1.3. Metric

The following metrics are used to measure results in our experiments:

- **Acc** (Accuracy). This metric is used to measure the performance of classification tasks. $Acc = (TP + TN)/(TP + FP + TN + FN)$, where *TP, TN, FP*, and *FN* are the number of true positives, false positives, true negatives, and false negatives respectively. The higher the Acc value, the higher the classification accuracy.
- **MAPE** (Mean Absolute Percentage Error). $MAPE = \frac{100\%}{T}\sum_{t=1}^{n}|\frac{y_t-\bar{y}_t}{y_t}|$, where y_t and $\bar{y}_t$ are the real BOR value and predicted real value, respectively, and *n* is the total number of testing movies. The smaller the MAPE value, the higher the prediction accuracy.
- **NDCG** (Normalized Discounted Cumulative Gain).
Firstly, the discounted cumulative gain (DCG) is given by: $DCG@n = \{ {r_1, if\ n = 1 \atop NCG@(n-1) + \frac{r_n}{log_2 n}, if\ n \geq 2}$.where r is the predicted ranking number. Later, given the ideal discounted cumulative gain $DCG^{'}@n$, *NDCG@n* can be computed as $NDCG@n = DCG@n/DCG^{'}@n$. The larger the *NDCG@n*, the higher the top *n* ranking accuracy

7.2. Training and testing data splits

In order to have more data for the feature extraction, we use movies released from 2015 to 2017 for prediction and use the rest of the data to learn the feature learning models. In this section, we show how to divide the training and testing data from the movies of 2015 to 2017 for the prediction and ranking model.

First, we demonstrate the BOR distribution of movies from 2015 to 2017 in Fig. 5(a). It is a long-tail distribution, the imbalance of which increases the difficulty of making predictions [82]. Considering the practicality, we should pay more focus on movies with higher BOR. Motivated by this and in order to alleviate the imbalance, we select 466 movies from the 1st quarter (Q1) of 2015 to the 3rd quarter (Q3) of 2017 as our training and testing data, belonging to six BOR ranges as evenly as possible, as reported in Table 3. Note that except for the first value range ($<$ 1.5M\$) where eight movies of each quarter are randomly sampled from a large number of movies with low BOR value, the number of movies of other value ranges is the actual number of released movies in the corresponding time range.

Then we attempt to divide training and testing data from the 466 movies. Firstly, we use a quarter of movies as the testing set. Thus, the problem becomes how to determine a proper time range before the testing quarter where movies released in the range are our training data. To this end, a classification experiment in terms of the six BOR classes is designed, in which we use different periods of training data as listed in Table 4. Concretely, these training sets vary data from one quarter to eight quarters. For each training set, we use it to learn one-vs-rest logistic regression models. Results in terms of Acc according to each classifier are depicted in Fig. 5(b).

Results. As seen, classifiers trained by the 3rd, 4th, and 5th training sets can obtain better results. Thus, we select the length of the 4th training set (namely five quarters) to be the training data period hereafter. With this in hand, we employ a sliding window-based method to generate training and testing sets for subsequent experiments where the window size is six quarters (the first five quarters belong to a training set and movies in the last one quarter form the testing set), and the sliding step is set to one quarter. As a result, six sets are generated as reported in Table 5.

7.3. Parameter selection

In this section, we employ the classification task to determine appropriate parameters for the proposed models, i.e., MINE and the deep network. In each model, we firstly adjust one parameter with other parameters fixed by default to obtain different embeddings. Then, we use

1 http://www.mtime.com/ .
2 https://www.imdb.com/ .
3 https://github.com/fxsjy/jieba.

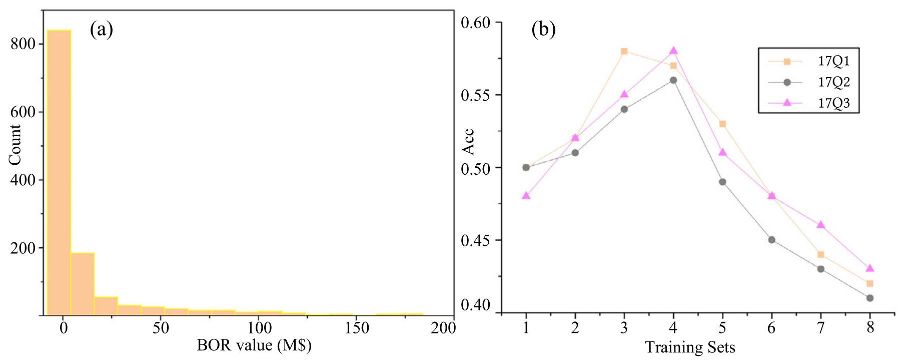

Fig. 5. (a) BOR value distribution for 2015–2017; (b) ACC results of each Training & Testing set.

Table 3
Movie distribution w.r.t BOR ranges and quarters (Q: quarter).

NO.	Time Range	1.5 <	1.5-4.6	4.6–12.3	12.3–46.2	46.2–153.8	> 153.8	Tot.
1	15Q1	8	9	6	8	10	0	41
2	15Q2	8	12	4	6	6	3	39
3	15Q3	8	10	10	5	9	4	46
4	15Q4	8	8	11	10	8	1	46
5	16Q1	8	10	8	7	4	3	40
6	16Q2	8	2	6	10	9	2	37
7	16Q3	8	6	9	12	10	2	47
8	16Q4	8	10	10	9	8	1	46
9	17Q1	8	4	3	8	7	3	33
10	17Q2	8	14	11	9	5	3	50
11	17Q3	8	2	7	9	13	2	41
Tot.	15Q1-17Q3	88	87	85	93	89	24	466

Table 4
Training and testing sets for validating the periods of training data.

Testing	Training Set [# (periods)]			
set	1(8Q)	2(7Q)	3(6Q)	4(5Q)
17Q1	15Q1-16Q4	15Q2-16Q4	15Q3-16Q4	15Q4-16Q4
17Q2	15Q2-17Q1	15Q3-17Q1	15Q4-17Q1	16Q1-17Q1
17Q3	15Q3-17Q2	15Q4-17Q2	16Q1-17Q2	16Q2-17Q2
	5(4Q)	6(3Q)	7(2Q)	8(1Q)
17Q1	16Q1-16Q4	16Q2-16Q4	16Q3-16Q4	16Q4
17Q2	16Q2-17Q1	16Q3-17Q1	16Q4-17Q1	17Q1
17Q3	16Q3-17Q2	16Q4-17Q2	17Q1-17Q2	17Q2

Table 5
Training and testing sets for subsequent experiments.

No.	Training set	Testing set	NO.	Training set	Testing set
I	15Q1-16Q1	16Q2	IV	15Q4-16Q4	17Q1
II	15Q2-16Q2	16Q3	V	16Q1-17Q1	17Q2
III	15Q3-16Q3	16Q4	VI	16Q2-17Q2	17Q3

features generated by these embeddings to train different one-vs-rest logistic regression models for the six classes of BOR. At last, we select the model with the best accuracy score and make the parameter of the model be our parameter. We adopt datasets of I, II, and III in the experiment.

7.3.1. Parameter selection of MINE

We firstly provide the default settings hereafter: the number of actors is set as 3, the number of directors equals to 1, and the number of production companies is 2. The embedding dimension m=32, the sparse embedding n=128; the coefficients $\delta = \epsilon = \tau = 1$, $\mu = \rho = 10$. Among them, as we treat each component of our model having the same effect, the coefficients are determined based on the scale of each component. Thus, in this section, we focus on discussing the settings of the number of participants and the dimension of embeddings.

The number of participants (1) According to Nelson and Glotfelty [83], many studies focused on the presence of a single star fail to account for the possible synergy of multi-actors and lead to overestimates of the impact of a single star on the BOR. Thus, we consider three actors in our model by following the setting of Nelson and Glotfelty [83]. To validate it, we conduct an experiment by choosing the number of actors from 1 to 4. The result is illustrated in Fig. 6(a), showing that the model with multiple actors is better than that of employing a single actor. Based on the result, we select to consider three actors for our MINE finally.

(2) In general, there is only one director for each film. Moreover, we use our data to verify this common sense. About 87% movies have only one director in our dataset. Therefore, we only select a director for each movie.

(3) Initially, domain knowledge motivate us to select two production companies for a movie. Specifically, there are usually two companies to cooperate in producing a movie, including a larger company that is responsible for overall planning and a smaller company that is in charge of specific implementations. Take Iron Man as an example, Paramount Pictures is the larger company, and Marvel Studios Inc. is the smaller company. In addition, we also conduct an experiment to validate the selection, in which we choose the number of companies from 1 to 4. As shown in Fig. 6(b), the result supports the knowledge. As a result, we set the number of companies as 2.

The dimension of embeddings As shown in Fig. 6(c) and (d), we obtain the classification results with $m \in \{32, 64, 128, 256\}$ and $n \in \{16, 32, 64, 128\}$ respectively. With fewer dimensions, performances of embeddings are limited as less information is captured. While having higher dimensions, the difficulty of training will increase to deteriorate the performance. Therefore, we select appropriate dimensions accordingly, i.e., $m = 128$ and $n = 32$.

7.3.2. Parameter selection of the deep network

Firstly, the default settings of the deep network are listed hereafter. We extract sixty keyframes for each trailer. Based on the setting of GoogLeNet [77], the output dimensions of the GoogLeNet are 1000.

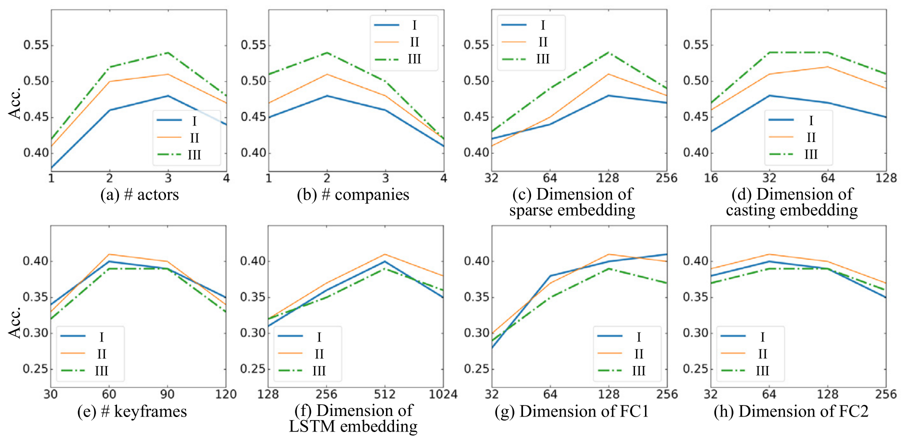

Fig. 6. Results of parameter selections.

Next, an LSTM layer follows, each of which has 512 memory cells. Then, for the Siamese network, we set dimensions of layers “FC1”, “FC2”, and “FC3” to 128, 64, and 16, respectively. In addition to the pre-determined parameters, i.e., the dimension of output embedding of GoogLeNet and “FC3”, these other parameters are selected based on the corresponding experimental results, as depicted in Fig. 6(e), (f), (g), and (h).

7.4. Effectiveness of features

In this section, we also use the classification task to evaluate the effectiveness of these learned features for each influential factor quantitatively. More specifically, we use different features to train a one-vs-rest logistic regression for the six classes of BOR and use the Acc score to evaluate the results. In addition, we provide case studies to demonstrate the knowledge encoded in each kind of feature visually.

7.4.1. Effectiveness of casting features

Settings. We evaluate the effectiveness of feature (F_n) obtained by embeddings of our MINE by comparing it with the hand-crafted features (F_n') of participants [8,25–27] and the features generated by embeddings learned from the following three groups of state-of-the-art methods.

(1) Features obtained by two static network-based embedding methods:
  - F_{LINE} [34]: LINE encodes the 2nd- order of node proximity for static homogeneous networks;
  - $F_{Metapath2vec}$ [39]: meta path random walk-based method based on static HINs;

(2) Features obtained by two incrementally updated-based dynamic network embedding models, which are based on homogeneous networks:
  - F_{TRIP} [48]: TRIP is an online algorithm to track the eigen functions of a dynamic graph, which preserves the first-order proximity between nodes;
  - F_{DHPE} [50]: DHPE is a high-order proximity preserving dynamic network embedding method.

(3) Features obtained by two variants of MINE:
  - F_{M1}: M1 is only the static module of MINE;
  - F_{M2}: M2 is M1 with the global correlation module;

Notably MINE is M2 with the local smoothness module, and we retain the setting of MINE for its variants. The embedding dimension of these embedding learning methods is also 32; the number of walks per node is 200; the walk length is 50; the neighborhood size is 5, and the size of the negative samples is 3. Other parameters of the baselines are set by grid search. The process of each of the following tasks is repeated 10 times, and the average results are reported.

Results. The results in terms of the Acc for the six-class classification problem are presented in Table 6. Overall, using MINE outperforms all the baselines. Specifically:

(1) MINE is better than the hand-crafted features, showing that features uncovered in a collective environment are more useful than independent individual ones since a movie is produced by the co-operation of different participants.
(2) MINE outperforms the *M1* and *M2*. First, results of M1 reveal that the semantic and structural information is preserved so as to direct this discriminative task. Second, the fact that M2 performs better than M1 and that MINE further improves upon M2 both indicate that the shared learning space can support capturing the cooperation information.
(3) MINE outperforms Mp2v, which itself performs better than LINE in the heterogeneous task. With the help of the shared learning space, the gain obtained by MINE over Mp2v is up to 4–25%.
(4) MINE outperforms TRIP and DHPE significantly. This is because both the two dynamic methods are designed for homogeneous networks, and are thus incapable of modeling the cooperation on such a complex heterogeneous environment.

7.4.2. Effectiveness of shooting features

Settings. We evaluate effectiveness of shooting features obtained by our deep network by comparing it with two kinds of lower-level features and conventional hand-crafted features.

- F_v': hand-crafted features extracted from trailers [5,6,28];
- F_l: low-level feature vector, i.e., element-wise multiplication among the GoogLeNet output of each keyframe;
- F_m: middle-level feature vector, i.e., the output of the trailer embedding extraction network;
- F_v: high-level feature vector, i.e., our shooting feature vector;

Results. The results in terms of the Acc for the six-class classification problem are presented in Table 7. Specifically, our visual feature vector performs better than hand-crafted features. In addition, with increasing depth of learning (i.e., from F_l to F_m, and further to F_v), the accuracy increases gradually, which demonstrates the effectiveness of our learning architecture.

7.4.3. Effectiveness of feature combinations

In addition to the above two kinds of features, we also use a topic model to extract knowledge from the textual data and design several hand-crafted features from the distribution data. In this section, we firstly evaluate the effectiveness of the two kinds of features. Then, we show the performances of combinations of these features, including the learned features and conventional features. Note that as the methods of feature learning from the textual and distribution data are similar to

Table 6
Acc results for the BOR classification using various casting features.

Sets	$F_n^{'}$	F_{LINE}	F_{Mp2v}	F_{TRIP}	F_{DHPE}	F_{M1}	F_{M2}	$F_n(MINE)$
I	0.392	0.331	0.404	0.283	0.302	0.392	0.435	**0.481**
II	0.357	0.366	0.463	0.294	0.346	0.421	0.458	**0.514**
III	0.404	0.392	0.428	0.302	0.379	0.453	0.503	**0.536**
IV	0.394	0.406	0.431	0.365	0.406	0.440	0.511	**0.534**
V	0.406	0.375	0.476	0.374	0.386	0.408	0.480	**0.502**
VI	0.382	0.343	0.417	0.339	0.322	0.403	0.474	**0.519**
Aver.	0.393	0.369	0.437	0.326	0.357	0.420	0.477	**0.514**

Table 7
Acc results for the BOR classification using various shooting features.

Sets	$F_v^{'}$	F_l	F_m	F_v
I	0.318	0.272	0.347	**0.403**
II	0.344	0.291	0.363	**0.412**
III	0.331	0.303	0.338	**0.394**
IV	0.349	0.298	0.374	**0.408**
V	0.335	0.279	0.355	**0.406**
VI	0.313	0.266	0.344	**0.391**
Aver.	0.334	0.285	0.354	**0.402**

Table 8
Acc. results for the BOR classification using different feature combinations.

Features	I	II	III	IV	V	VI	Aver.
F_t	0.391	0.374	0.386	0.383	0.366	0.368	0.378
F_d	0.431	0.425	0.439	0.416	0.424	0.417	0.425
$F_t + F_d$	0.458	0.449	0.451	0.435	0.443	0.446	0.447
$F_t + F_d + F_n^{'}$	0.473	0.467	0.480	0.463	0.473	0.468	0.471
$F_t + F_d + F_v^{'}$	0.464	0.453	0.465	0.447	0.456	0.459	0.457
$F^{'}$	0.482	0.476	0.492	0.480	0.478	0.484	0.482
$F_t + F_d + F_n$	0.548	0.561	0.573	0.565	0.552	0.569	0.561
$F_t + F_d + F_v$	0.499	0.493	0.492	0.483	0.487	0.501	0.493
F	**0.576**	**0.585**	**0.594**	**0.574**	**0.570**	**0.582**	**0.580**

that of previous studies, we treat the textual and distribution features as shared features of both the learned features and conventional features. Specifically, we feed the following features to train classification models. The accuracy results of these models w.r.t. each testing set are reported in Table 8.

- F_t: textual features;
- F_d: distribution features;
- $F_t + F_d$: combining textual and distribution features;
- $F_t + F_d + F_n^{'}$: combining textual, distribution, and hand-crafted casting features;
- $F_t + F_d + F_v^{'}$: combining textual, distribution, and hand-crafted visual features;
- $F^{'}$: combination of all conventional hand-crafted features, i.e., $F_n^{'} + F_v^{'} + F_t + F_d$;
- $F_t + F_d + F_n$: combining textual, distribution, and the learned casting features;
- $F_t + F_d + F_v$: combining textual, distribution, and the learned visual features;
- F: the learned feature vector, namely, $F_t + F_d + F_v + F_n$.

Results. Comparing the performance using F_t, F_d, $F_t + F_d$ and $F_t + F_d + F_v$, it is evident that F_t and F_d can both contribute to the prediction accuracy. The results of feature combinations reveal the advantage of our strategy that takes consideration of these factors entirely. Moreover, the results of $F^{'}$ and F indicate that the latter brings a significant improvement to the classification precision, exhibiting the effectiveness of our feature extraction models further.

7.4.4. *Case studies of embedding vectors*

Casting features We select a snapshot and use t-SNE [84] to reduce our embeddings to 3D. They are shown in Fig. 7 type by type. From the 2D projection surfaces, we notice that they have a linear trend, which hints that the embeddings are equipped with a high distinguishability since the semantic and structural information has been embodied.

Shooting features For the DNN-based model, there are three levels of vectors can be used to represent a trailer, namely, the low-level vector, the middle-level vector, and the high-level vector (i.e., the shooting features), respectively. In order to reveal knowledge encoded in the three-level vectors, similar to the textual vectors, we map them into two dimensions using t-SNE [84] and plot them in Fig. 8(b), (c), and (d), respectively, in which we also mark out the location of the two Chinese and the ten Hollywood movies (their locations are at the far left of their captions).

As demonstrated, through preserving the knowledge of the shooting factor by the three-level learning process, movies with similar BOR values tend to cluster together. In particular, we see a clear trend in Fig. 8(d) where movies show up in descending order from left to right according to their BORs, which indicates there is a strong correlation between this knowledge and BOR. Therefore, the kind of embedding is significantly associated with the prediction task.

Screenwriting features We adopt t-SNE [84] to reduce the textual vectors to two dimensions and show them in Fig. 8(a). In these figures, each point represents a movie, while the size of the point is directly proportional to the BOR of the movie. In addition, we mark out the location of two Chinese and ten Hollywood hot movies in the figures as examples (their locations are at the far right of their captions). Movies with similar topics can be distributed in a nearby region, e.g. these superhero movies, which thus can assist in the ultimate prediction goal.

Distribution features We also show the relationship between BOR and five vital hand-crafted features in Fig. 9. Sequels and other IP movies are better received than other movies (Fig. 9(a)). For a movie, the fewer competitors, the higher the chance to get a higher BOR (top of Fig. 9(b)). The number of promotion news articles of a movie in professional film portal websites indicates the success of the promotion strategy (bottom of Fig. 9(b)). In addition, the average daily BOR of movies on vacations and holidays are shown in Fig. 9(c) and (d). They are brighter spots than ordinary days. As a result, these features contain the BOR-related knowledge that can thus help the BOR prediction.

7.5. *Effectiveness of prediction model*

In this section, we show the effectiveness of the prediction and ranking model by the following two experiments.

7.5.1. *Effectiveness of real-value prediction*

To support real-world applications, we evaluate the effectiveness of real-value prediction. To this end, we feed our movie features (F) and the traditional features ($F^{'}$) to the following algorithms to compare the MAPE results. Note that, the task is extremely difficult as the range of BOR values is very large ($0-about $1B).

- PRBO, the employed prediction and ranking model;
- LR, the linear regression model;

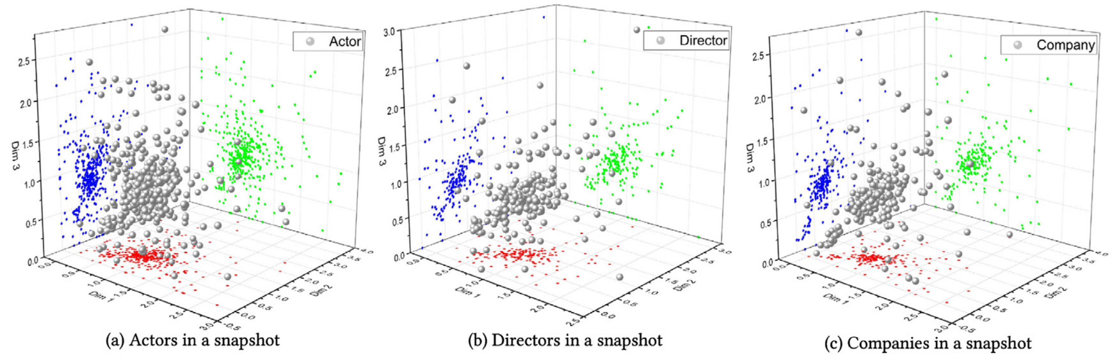

Fig. 7. Visualization of participant embeddings w.r.t. each type.

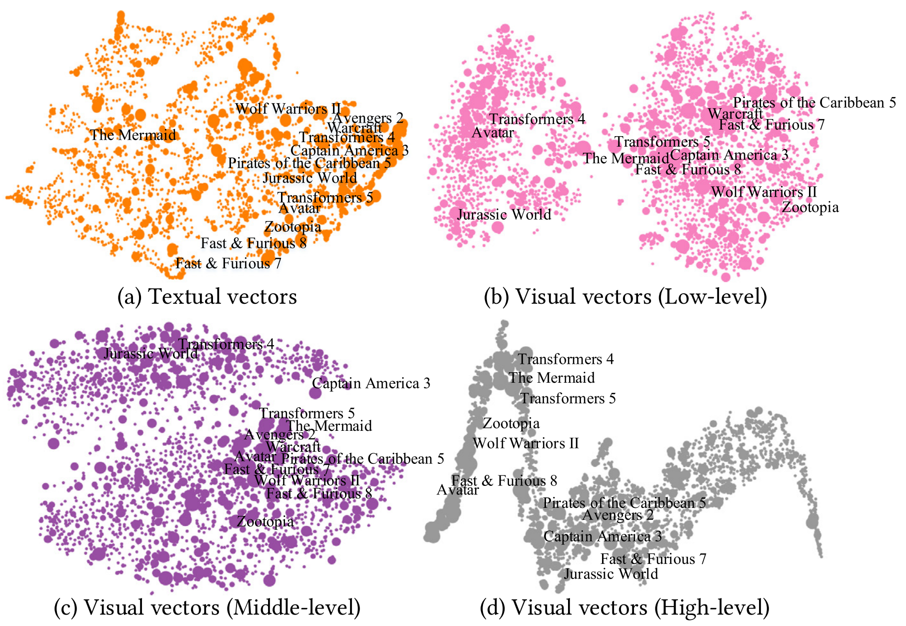

Fig. 8. t-SNE of screenwriting and casting features.

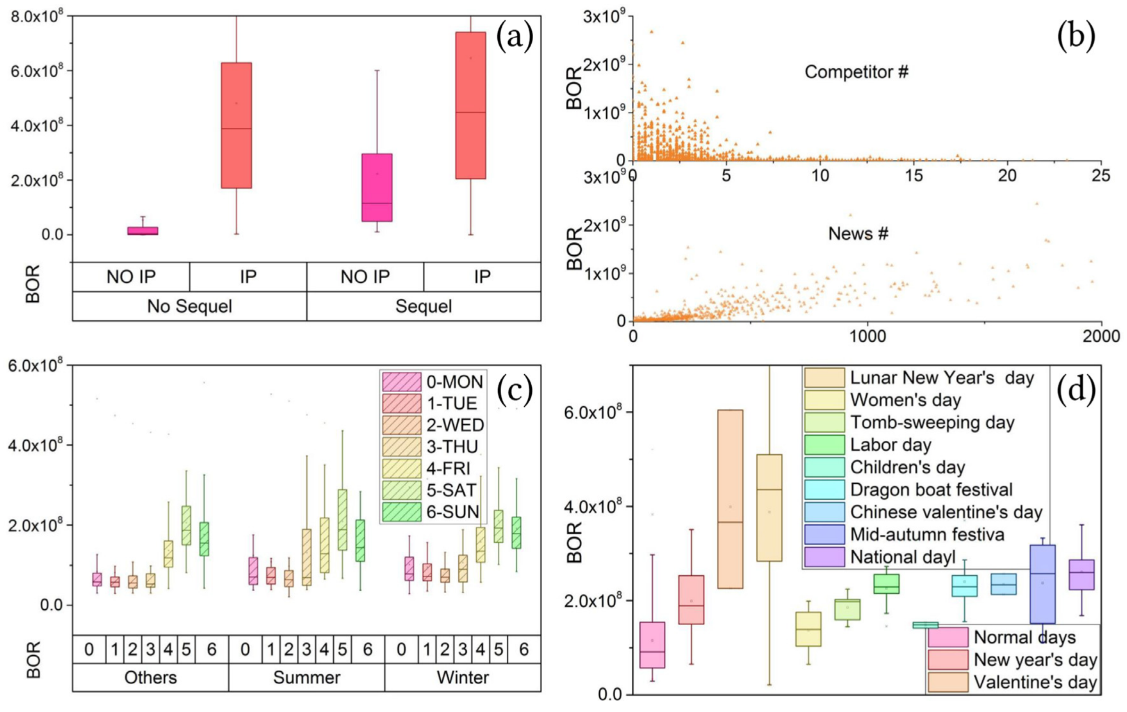

Fig. 9. Effectiveness of distribution features.

Table 9
Comparisons of each predictor based on MAPE.

BOR range		< 1.5	1.5-4.6	4.6–12.3	12.3–46.2	46.2–153.8	> 153.8	Aver.
LR	F'	134%	139%	124%	175%	261%	296%	188%
	F	82%	74%	85%	65%	58%	54%	70%
LASSO	F'	125%	105%	99%	154%	247%	292%	170%
	F	75%	67%	82%	61%	53%	49%	65%
SVR-RBF	F'	223%	278%	282%	384%	447%	491%	351%
	F	162%	154%	144%	165%	159%	171%	160%
Ada-R2	F'	103%	91%	87%	127%	213%	254%	146%
	F	81%	65%	55%	56%	48%	51%	59%
RF	F'	86%	83%	88%	146%	206%	257%	144%
	F	66%	64%	58%	53%	51%	53%	58%
GBRT	F'	121%	104%	096%	139%	214%	265%	157%
	F	86%	68%	58%	52%	47%	43%	59%
PRBO	F'	108%	95%	87%	167%	237%	271%	161%
	F	**63%**	**61%**	**54%**	**47%**	**45%**	**39%**	**52%**

- LASSO [85]: the LR model with L1 sparsity normalization;
- Ada-R2 [86]: the decision tree regression with AdaBoost;
- SVR-RBF [87]: the support vector regression with the rational basis function kernel;
- RF [88]: the random forest;
- GBRT [89]: the gradient boosting regression tree.

We set a, b, and σ^2 as 0.01 0.01, and 1000 for the PRBO model, respectively. For Lasso, the constant that multiplies the L1 term is 0.1. For Ada-R2, the number of estimators is 50, the learning rate is 0.01, and the maximum depth of the decision tree is 3. For SVR-RBF, the penalty parameter is 0.1. For RF, the number of trees is 50, and the max depth is 4. For GBRT, the learning rate is 0.01, and the number of boosting stages is 100.

Results. Average MAPE results among six testing sets w.r.t. each revenue range are reported in Table 9. In general, the performance of methods using F significantly outperform that of F'. According to the results of using our representation, most of the methods can achieve good performance. We make two observations:

(1) *BOR range*. Clearly, for movies with BOR of more than \$12.3M, all methods obtain a promising result. However, the error is relatively high when the revenue is less than \$12.3M, because a part of these movies, especially in the \$1.5M-\$12.3M range are produced by renowned teams yet, are destroyed by bad reputations [90] that are not easy to be identified before releasing. Another part of movies are small productions without sufficient data for modeling them, leading to the relative inaccuracy.
(2) *Prediction method*. Because of the limitation of training samples, the model SVR-RBF shows the weakest performance, whereas our PRBO, with the help of the ranking information, outperforms all the baselines. Moreover, our method only deviates about 39% absolute error for these most bankable movies (> \$153.8M), which is of great value to direct the investment.

7.5.2. Effectiveness of ranking

In addition, as our PRBO method also provides ranking results, we further evaluate the ranking performance using the following baselines where we both feed our movie feature (F) and the traditional feature (F') to obtain ranking (NSGD) results.

- MART [89]: the boosted tree model;
- RankNet [81]: the neural network-based ranking method;
- RankBoost [91]: the model combined many “weak” rankers;
- Coordinate Ascent (CA) [92]: the model applies coordinate descent for optimization.

For MART, we set the number of trees, leaves, and threshold candidates as 50, 5, and 10, respectively, and the learning rate as 0.01. For RankNet, the number of epochs as 100, the number of hidden layers as 1, and the number of hidden nodes per layer as 20. For RankBoost, we set the number of iterations to 300, and the number of threshold candidates as 10. For CA, we set step base as 0.05, step scale as 2.0, tolerance as 0.001, and slack as 0.001.

Table 10
Comparisons of each ranker based on NDCG .

Methods	@3		@5		@7		@10	
	F'	F	F'	F	F'	F	F'	F
MART	0.51	0.78	0.54	0.91	0.59	0.90	0.68	0.88
RankNet	0.43	0.59	0.44	0.69	0.53	0.82	0.64	0.80
CA	0.32	0.55	0.34	0.64	0.46	0.61	0.50	0.62
RankBoost	0.53	0.74	0.61	0.86	0.62	0.88	0.71	0.89
PRBO	0.46	**0.94**	0.48	**0.95**	0.56	**0.96**	0.57	**0.96**

Results. Average NDCG results among all the six testing sets w.r.t. four n numbers are listed in Table 10. Supported by our movie feature vectors, each model indicates a promising ranking performance. As enhanced by the prediction ability, the PRBO demonstrates the superiority over other baselines.

As a consequence, experimental results reveal the applicability of the PRBO model in the real-world since the model can both provide two kinds of guaranteed results for investors.

8. Conclusion

In this paper, we proposed a two-stage machine learning method for predicting the BOR of a movie before releasing it. We propose utilizing various feature learning models for extracting different features from multi-modal data of the movie. In particular, we propose a novel dynamic heterogeneous network embedding model to collectively learn latent representations of actors, directors, and companies, capturing their cooperation relationship collectively. We design a deep neural network-based model to uncover high-level representations of movie quality from trailers. Then, a movie’s feature vector, consisting of these vectors, is fed into a prediction and ranking model to obtain the BOR prediction. The experimental results on the Chinese film market data show the advantage of both our movie feature vectors and the prediction results. In particular, the solution only has an error of about 40% absolute percentage error for bankable movies in a very large prediction space and also offers a promising ranking result. Finally, the real-world applicability of the solution drives us to deploy the solution as a business service to support investment decisions.

For future work, it is promising to explore NLP technologies in-depth for these textual data about movies. Second, in this paper, we specified an embedding method, i.e., MINE, for the participant embedding problem. In the future, it is interesting to investigate a general version of our model, such that our model can be employed to wide applications.

Author Contribution Statement

Zhaoyuan Wang, Junbo Zhang, and Yu Zheng conceived of the presented idea. Zhaoyuan Wang, Junbo Zhang, Shenggong Ji, and Chuishi Meng developed the theory and the program. Zhaoyuan Wang, Shenggong Ji, and Chuishi Meng verified the proposed methods. Tianrui Li and Yu Zheng encouraged Zhaoyuan Wang to investigate the data fusion aspect of this paper and supervised the findings of this work. All authors discussed the results and contributed to the final manuscript.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (No. 61773324) and the National Key R&D Program of China (No. 2019YFB2101800). Finally, thank Chentian Jin from Tsinghua University for polishing the paper.

References

[1] M. Hauge, Writing Screenplays that Sell, Collins Reference, New York, 1991.
[2] I.R. Blacker, Elements of Screenwriting: A Guide for Film and Television Writing, Pearson, Reissue, 1996.
[3] J. Steiff, The Complete Idiot's Guide to Independent Filmmaking, Alpha Books, 2005.
[4] B. Cleve, Film Production management, Focal Press, 2012.
[5] A. Tadimari, N. Kumar, T. Guha, S.S. Narayanan, Opening Big in Box Office? Trailer Content can Help, IEEE, 2016, pp. 2777–2781.
[6] H. Zhou, T. Hermans, A.V. Karandikar, J.M. Rehg, Movie genre classification via scene categorization, in: International Conference on Multimedea 2010, Firenze, Italy, October, 2010, pp. 747–750.
[7] B.D. Silva, R. Compton, Prediction of foreign box office revenues based on Wikipedia page activity, arXiv:1405.5924 (2014).
[8] R. Sharda, D. Delen, Predicting box-office success of motion pictures with neural networks, Expert Syst. Appl. 30 (2) (2006) 243–254.
[9] B.R. Litman, Predicting success of theatrical movies: an empirical study, J. Pop. Cult. 16 (4) (1983) 159–175.
[10] N.Z. Foutz, W. Jank, The Wisdom of Crowds: Pre-Release Forecasting via Functional Shape Analysis of the Online Virtual Stock Market, Social Science Electronic Publishing, 2009.
[11] J. Mckenzie, Predicting box office with and without markets: do internet users know anything? Inf. Econ. Policy 25 (2) (2013) 70–80.
[12] B. Somlo, K. Rajaram, R. Ahmadi, Distribution planning to optimize profits in the motion picture industry, Prod. Oper. Manag. 20 (4) (2011) 618–636.
[13] T. Hennig-Thurau, M.B. Houston, G. Walsh, Determinants of motion picture box office and profitability: an interrelationship approach, Rev. Manag. Sci. 1 (1) (2007) 65–92.
[14] D.A. Edwards, R. Buckmire, J. Ortegagingrich, A mathematical model of cinematic box-office dynamics with geographic effects, IMA J. Manage. Math. 25 (2) (2014) 233–257.
[15] R. Neelamegham, P. Chintagunta, A bayesian model to forecast new product performance in domestic and international markets, Mark. Sci. 18 (2) (1999) 115–136.
[16] T. Kim, J. Hong, P. Kang, Box office forecasting using machine learning algorithms based on SNS data, Int. J. Forecast. 31 (2) (2015) 364–390.
[17] J. Du, H. Xu, X. Huang, Box office prediction based on microblog, Expert Syst. Appl. 41 (4) (2014) 1680–1689.
[18] M. Hur, P. Kang, S. Cho, Box-office forecasting based on sentiments of movie reviews and independent subspace method, Inf. Sci. 372 (2016) 608–624.
[19] M. Arias, A. Arratia, R. Xuriguera, Forecasting with twitter data, ACM Trans. Intell. Syst. Technol. 5 (1) (2014) 1–24.
[20] E.V. Karniouchina, Impact of star and movie buzz on motion picture distribution and box office revenue, Int. J. Res. Mark. 28 (1) (2011) 62–74.
[21] T. Liu, X. Ding, Y. Chen, H. Chen, M. Guo, Predicting movie box-office revenues by exploiting large-scale social media content, Multimed. Tools Appl. 75 (3) (2016) 1509–1528.
[22] M. Ghiassi, D. Lio, B. Moon, Pre-production forecasting of movie revenues with a dynamic artificial neural network, Expert Syst. Appl. 42 (6) (2015) 3176–3193.
[23] H. Rui, Y. Liu, A. Whinston, Whose and What Chatter Matters? The Effect of Tweets on Movie Sales, 55, Social Science Electronic Publishing, 2012, pp. 863–870.
[24] Y. Zhou, L. Zhang, Z. Yi, Predicting movie box-office revenues using deep neural networks, Neural Comput. Appl. (3) (2017) 1–11.
[25] L. Zhang, J. Luo, S. Yang, Forecasting box office revenue of movies with bp neural network, Expert Syst. Appl. 36 (3) (2009) 6580–6587.
[26] D. Delen, R. Sharda, Predicting the financial success of hollywood movies using an information fusion approach, J. Ind. Eng. 21 (1) (2010) 30–37.
[27] W. Zhang, S. Skiena, Improving movie gross prediction through news analysis, in: IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies, 2009, pp. 301–304.
[28] S. Oh, J. Ahn, H. Baek, Viewer engagement in movie trailers and box office revenue, in: Hawaii International Conference on System Sciences, 2015, pp. 1724–1732.
[29] I. Hunter, S. David, S. Smith, S. Singh, Predicting box office from the screenplay: a text analytical approach, J. Screenwriting 7 (2) (2016) 135–154.
[30] J. Eliashberg, S.K. Hui, Z.J. Zhang, Assessing box office performance using movie scripts: a kernel-based approach, IEEE Trans. Knowl. Data Eng. 26 (11) (2014) 2639–2648.
[31] R. Parimi, D. Caragea, Pre-release box-office success prediction for motion pictures, in: International Conference on Machine Learning and Data Mining in Pattern Recognition, 2013, pp. 571–585.
[32] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv:1301.3781 (2013).
[33] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 701–710.
[34] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.
[35] A. Grover, J. Leskovec, node2vec: scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 855–864.
[36] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1225–1234.
[37] Y. Sun, J. Han, Mining Heterogeneous Information Networks: Principles and Methodologies, Morgan and Claypool Publishers, 2012.
[38] C. Shi, Y. Li, J. Zhang, Y. Sun, P.S. Yu, A survey of heterogeneous information network analysis, IEEE Trans. Knowl. Data Eng. 29 (1) (2017) 17–37.
[39] Y. Dong, N.V. Chawla, A. Swami, metapath2vec: scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 135–144.
[40] T. Fu, W.-C. Lee, Z. Lei, Hin2vec: explore meta-paths in heterogeneous information networks for representation learning, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, 2017, pp. 1797–1806.
[41] C. Shi, B. Hu, X. Zhao, P. Yu, Heterogeneous information network embedding for recommendation, IEEE Transactions on Knowledge and Data Engineering (2018). Early Access
[42] D. Zhang, J. Yin, X. Zhu, C. Zhang, Metagraph2vec: Complex Semantic Path Augmented Heterogeneous Network Embedding, Springer, 2018, pp. 196–208.
[43] M. Liu, J. Liu, Y. Chen, M. Wang, H. Chen, Q. Zheng, Ahng: representation learning on attributed heterogeneous network, Inf. Fusion 50 (2019) 221–230.
[44] Z. Yan, J. Liu, L.T. Yang, W. Pedrycz, Data fusion in heterogeneous networks, Inf. Fusion 53 (2020) 1–3.
[45] J. Tang, M. Qu, Q. Mei, Pte: predictive text embedding through large-scale heterogeneous text networks, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 1165–1174.
[46] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, J. Tang, Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM, 2018, pp. 459–467.
[47] P. Cui, X. Wang, J. Pei, W. Zhu, A survey on network embedding, IEEE Trans. Knowl. Data Eng. (2018) preprint.
[48] C. Chen, H. Tong, Fast eigen-functions tracking on dynamic graphs, in: Proceedings of the 2015 SIAM International Conference on Data Mining, SIAM, 2015, pp. 559–567.
[49] P. Goyal, N. Kamra, X. He, Y. Liu, Dyngem: deep embedding method for dynamic graphs, arXiv:1805.11273 (2018).
[50] D. Zhu, P. Cui, Z. Zhang, J. Pei, W. Zhu, High-order proximity preserved embedding for dynamic networks, IEEE Trans. Knowl. Data Eng. 11 (30) (2018) 2134–2144.
[51] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: International Conference on Neural Information Processing Systems, 2012, pp. 1097–1105.
[52] C. Feichtenhofer, A. Pinz, A. Zisserman, Convolutional two-stream network fusion for video action recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1933–1941.
[53] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2012) 221–231.
[54] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.
[55] Y.H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, G. Toderici, Beyond short snippets: deep networks for video classification 16(4) (2015) 4694–4702.
[56] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, F.F. Li, Large-scale video classification with convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1725–1732.
[57] M. Chen, S. Mao, Y. Liu, Big data: a survey, Mob. Netw. Appl. 19 (2) (2014) 171–209.
[58] A. Gandomi, M. Haider, Beyond the hype: big data concepts, methods, and analytics, Int. J. Inf. Manage. 35 (2) (2015) 137–144.
[59] Y. Zheng, Methodologies for cross-domain data fusion: an overview, IEEE Trans. Big Data 1 (1) (2015) 16–34.

[60] G. Nachouki, M. Quafafou, Multi-data source fusion, Information Fusion 9 (4) (2008) 523–537. Special Issue on Web Information Fusion.
[61] A. Smirnov, T. Levashova, Knowledge fusion patterns: a survey, Inf. Fusion 52 (2019) 31–40.
[62] X.-W. Chen, X. Lin, Big data deep learning: challenges and perspectives, IEEE Access 2 (2014) 514–525.
[63] M.M. Najafabadi, F. Villanustre, T.M. Khoshgoftaar, N. Seliya, R. Wald, E. Muharemagic, Deep learning applications and challenges in big data analytics, J. Big Data 2 (1) (2015) 1.
[64] Q. Zhang, L.T. Yang, Z. Chen, P. Li, A survey on deep learning for big data, Inf. Fusion 42 (2018) 146–157.
[65] D.M. Blei., A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (2012) 993–1022.
[66] X. Cheng, X. Yan, Y. Lan, J. Guo, Btm: topic modeling over short texts, IEEE Trans. Knowl. Data Eng. 26 (12) (2014) 2928–2941.
[67] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[68] Y. Fu, Y. Ge, Y. Zheng, Z. Yao, Y. Liu, H. Xiong, J. Yuan, Sparse real estate ranking with online user reviews and offline moving behaviors, in: IEEE International Conference on Data Mining, 2015, pp. 120–129.
[69] C. Yang, Z. Liu, D. Zhao, M. Sun, E. Chang, Network representation learning with rich text information, in: Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.
[70] H. Lee, A. Battle, R. Raina, A.Y. Ng, Efficient sparse coding algorithms, in: Proceeding of Advances in Neural Information Processing Systems, 2007, pp. 801–808.
[71] J. Zhou, J. Chen, J. Ye, Malsar: Multi-Task Learning via Structural Regularization, Arizona State University 21(2011).
[72] D.P. Bertsekas, Nonlinear programming, Journal of the Operational Research Society 48 (3) (1997). 334–334
[73] Y. Lécun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[74] A. Graves, Generating sequences with recurrent neural networks, arXiv:1308.0850 (2013).
[75] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[76] S. Chopra, R. Hadsell, Y. Lecun, Learning a similarity metric discriminatively, with application to face verification, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 2005, pp. 539–546 vol. 1.
[77] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[78] B. Duke, G.W. Taylor, Generalized hadamard-product fusion operators for visual question answering, in: 2018 15th Conference on Computer and Robot Vision (CRV), IEEE, 2018, pp. 39–46.
[79] D. Teney, P. Anderson, X. He, A. van den Hengel, Tips and tricks for visual question answering: learnings from the 2017 challenge, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4223–4232.
[80] K. Soomro, A.R. Zamir, M. Shah, Ucf101: a dataset of 101 human actions classes from videos in the wild, arXiv:1212.0402 (2012).
[81] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: International Conference on Machine Learning, 2005, pp. 89–96.
[82] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. (9) (2008) 1263–1284.
[83] R.A. Nelson, R. Glotfelty, Movie stars and box office revenues: an empirical analysis, J. Cult. Econ. 36 (2) (2012) 141–166.
[84] L.V.D. Maaten, G. Hinton, Visualizing data using T-SNE, J. Mach. Learn. Res. 9 (2605) (2008) 2579–2605.
[85] R. Tibshirani, Regression shrinkage and selection via the lasso: a retrospective, J. R. Stat. Soc. 73 (3) (2011) 273–282.
[86] H. Drucker, Improving regressors using boosting techniques, in: Fourteenth International Conference on Machine Learning, 1997, pp. 107–115.
[87] A. J. Smola, A tutorial on support vector regression, Stat. Comput. 3 (14) (1998) 199–222.
[88] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[89] J.H. Friedman, Greedy function approximation: a gradient boosting machine., Ann. Stat. 29 (5) (2001) 1189–1232.
[90] R.F. Baumeister, E. Bratslavsky, C. Finkenauer, K.D. Vohs, Bad is stronger than good, Rev. Gener. Psychol. 5 (4) (2001) 323–370.
[91] Y. Freund, R. Iyer, R.E. Schapire, Y. Singer, An efficient boosting algorithm for combining preferences, J. Mach. Learn. Research 4 (6) (2003) 170–178.
[92] D. Metzler, W.B. Croft, Linear feature-based models for information retrieval, Inf. Retr. 10 (3) (2007) 257–274.