12-2015

Modeling Social Media Content with Word Vectors for Recommendation

Ying DING
Singapore Management University, ying.ding.2011@phdis.smu.edu.sg

Jing JIANG
Singapore Management University, jingjiang@smu.edu.sg

Modeling Social Media Content with Word Vectors for Recommendation

Ying Ding(✉) and Jing Jiang

School of Information Systems, Singapore Management University, Singapore, Singapore
{ying.ding.2011,jingjiang}@smu.edu.sg

Abstract. In social media, recommender systems are becoming more and more important. Different techniques have been designed for recommendations under various scenarios, but many of them do not use user-generated content, which potentially reflects users' opinions and interests. Although a few studies have tried to combine user-generated content with rating or adoption data, they mostly reply on lexical similarity to calculate textual similarity. However, in social media, a diverse range of words is used. This renders the traditional ways of calculating textual similarity ineffective. In this work, we apply vector representation of words to measure the semantic similarity between text. We design a model that seamlessly integrates word vectors into a joint model of user feedback and text content. Extensive experiments on datasets from various domains prove that our model is effective in both recommendation and topic discovery in social media.

1 Introduction

With the explosive usage of social media, there are many recommendation problems we face on different social media platforms. Recommendation in social media is an important way to improve services and attract more users. For example, on Twitter, followee recommendation can help people find users they are interested in, and thus good followee recommendation can provide users with more sources of interesting information and keep them using the platform. On many other platforms, recommendation is also important, like product recommendation in online review websites, event recommendation in online event websites, etc. To finish a recommendation task in social media well, it is important to understand users' online behaviour and accurately model their preferences and interests.

A traditional solution to these recommendation problems is collaborative filtering. There are generally two kinds of collaborative filtering methods: memory-based methods and model-based methods. Memory-based methods measure the similarity between users or items by directly using the adoption history. Then they perform recommendation based on the calculated similarities. Model-based methods use latent vectors to represent the interests of users and properties of items. They recommend items based on similarities between user latent vectors

Published in Social Informatics: 7th International Conference, SocInfo 2015, Beijing, China, December 9-12, 2015, Proceedings, LNCS 9471, pp. 274-288.
DOI: 10.1007/978-3-319-27433-1_19

and item latent vectors. While collaborative filtering can get decent performance, it only uses adoption or rating data but misses other available information.

An important characteristic of social media platforms is that they allow users to contribute content in free-text form. For example, users can publish posts about their daily lives on Twitter, discuss social-political issues in online forums, label web pages with tags in online bookmark websites and write reviews of products in online review websites. These textual data reflects people's opinions, interests and preferences. Presumably it can help improve recommendation. Indeed there has been several recent studies trying to combine textual data with rating or adoption data for recommendation. Among these studies, many use textual information separately from their recommendation model. They first extract useful information from text and then embed such information in their recommendation models [4,30]. Some other work uses a unified, principled model to combine text with rating or adoption data [2,15,19].

A limitation of these recent studies is that their textual similarity is based on lexical similarity only. When two items' descriptions are semantically related but use different words, these models may not consider the two items to be similar. In social media, however, the vocabulary used is very diverse and two pieces of text can be semantically similar even with low lexical overlap, so semantic similarity is especially important when we analyze social media content. For example, in the Meetup dataset we use, which is about online interest groups and organized events, there is a group tagged with "Buddhism" and another group tagged with "vegetarian." If we only consider lexical similarity, these two groups may not be considered related based on the tags. However, we should probably recommend the second group to users who have joined the first group as many Buddhists are also vegetarians. The challenge is how to incorporate the consideration of semantic similarity based on textual descriptions into a traditional collaborative filtering framework in a principled way.

With the recent advances in learning word embeddings from large corpora, we can use vector representation of words to measure the semantic similarity between two pieces of text. Word embeddings are techniques that can project words into vectors carrying their semantic meanings [20,25]. In this paper, we propose a new recommendation model that makes use of word embeddings such that combining content and collaborative filtering becomes more effective. Our model can jointly model ratings, latent factors, topics and word embedding vectors simultaneously. With the help of vector representations of words, the model is able to learn cleaner topics, more accurate latent factors and provide better recommendations. Extensive experiments show that our model outperforms other methods on item recommendation and topic discovery. For example, for the Meetup data, our method can successfully recommend the "vegetarian" group to users who have joined the "Buddhism" group, and based on the ground truth, for such users our method indeed gives better performance than other baseline methods we consider.

In Section 2, we will briefly go through some recent related work. Our model will be introduced in Section 3 and our experiments are described in Section 4. We conclude this study and propose some potential research questions in Section 5.

2 Related Work

With the explosive growth of content in social media, recommender systems are becoming more and more important to users. New techniques adapted to different scenarios have been developed. On Twitter, various types of recommendation has been studied [14], like followee recommendation, tweet recommendation, hashtag recommendation, etc. User network in social media has also been used to improve product recommendation [8,18]. On some platforms, users can form a group for some purpose. Recommendation related to groups in social media has also been studied in some recent work. It includes work on recommendation for groups [1,7], which tries to recommend items to a group of users, group recommendation [33], which recommends groups to users to join, etc.

Traditional recommendation models mainly focus on users' adoption or rating histories [12]. In social media, the user-generated content provides us with useful information that can reflects users' opinions, interests and preferences. It is valuable to model this information to improve recommendation outputs [26]. To this end, people have tried to extract features from text as item representations [13,24]. While these studies use text separately from recommendation models, more work is trying to jointly model text and recommendation in one principled models [2,15,19]. However, these joint models rely on traditional method of modeling text, which uses only lexical similarity to calculate textual similarity.

Word embedding is a recently proposed technique inspired by advances in deep neural networks [20,25]. Based on the learning from large corpora, it can represent words with numerical vectors that carry their semantic meanings. The similarity between vectors of words that are semantically or syntactically similar will be high. It has been used in different applications such as information retrieval [5] and text summarization [11]. Deep learning itself has also been applied to model text in recommendation problems [31]. While this work applies a deep learning model—stacked autoencoder—to learn text representations. It does not make use of word embeddings pre-trained on a large external corpora, which can presumably better represent the semantic meanings of words than representations trained only from texts related to the recommendation problem itself.

3 Method

In this section, we formally formulate our problem and present our proposed model. Based on our model's properties, we denote it by **C**ollaborative filtering with word **E**mbedding-based **T**opic models (CET).

3.1 Problem Formulation and Notation

Suppose we have a collection of N_I items $\mathcal{I} = \{i_1, i_2, \cdots, i_{N_I}\}$ and a collection of N_U users $\mathcal{U} = \{u_1, u_2, \cdots, u_{N_U}\}$. We also observe a collection of ratings[1]

[1] For convenience, we assume we have numerical rating data, but the model can be easily generalized for binary adoption data.

$\mathcal{R} = \{r_{ui}\}$ where r_{ui} is the rating of item i by user u. For each item, there is an associated document d_i, which is a sequence of words. This document can be from different sources of user-generated content. For example, in online review websites, we can use the reviews of a product as the document associated with the product. For items that have user-assigned tags, we can use the set of tags as the associated document for an item. The set of all words appearing in our data comprises the vocabulary $\mathcal{V}$ and for each word w of this vocabulary, we assume that we have a pre-trained vector v_w of dimension K, which can be learned by word embedding models [20,25]. Our task of this work is to recommend items to users according to both their rating histories and the textual data generated by users in social media.

3.2 Collaborative Filtering with Word Embedding-Based Topic Models

Our model is based on matrix factorization, topic modeling and word embedding vectors. On the rating part, we apply matrix factorization as the generative process. On the text part, we design a generative process based on Latent Dirichlet Allocation (LDA) [3]. We also assume that the topic distribution of an item is linked to the item's latent vector used in matrix factorization, which is an idea previously explored in [15,19,29]. By doing this, we build a single unified and principled model that combines text and ratings. Similar to [29], we assume that item factors are derived from the corresponding topic distributions instead of setting them to be identical. This renders our model more flexible in modeling latent factors. Different from standard LDA, which treats each word as a single discrete symbol, we use the vector representations of them instead. We still assume that there is a multinomial topic distribution for each document. But for each topic, we assume there is a multivariate Gaussian distribution, which is used to generate word vectors. There are two parameters for each topic t, which are the mean vector μ_t and co-variance matrix Σ_t. To generate a word in a document, we first need to sample a topic according to the document-topic distribution, and then sample a vector from the Gaussian distribution of the sampled topic. The generative process of our model is shown below and we list the used notation in Table 1.

- For each user, sample a bias $b_u \sim \mathcal{N}(0, \sigma_U)$ and a latent vector $p_u \sim \mathcal{N}(0, \Gamma_U)$.
- For each item i, sample a topic distribution $\theta_i \sim \mathrm{Dir}(\alpha)$ for text. Sample a latent vector $q_i \sim \mathcal{N}(\theta_i, \Gamma_I)$ and a bias $b_i \sim \mathcal{N}(0, \sigma_I)$. For each word w in the associated text:
  - Sample a topic $z \sim \mathrm{Multi}(\theta_i)$.
  - Sample a word embedding vector $v_w \sim \mathcal{N}(\mu_z, \Sigma_z)$.
- For rating of item i by user u, sample a numerical value $r_{ui} \sim \mathcal{N}(b_u + b_i + p_u^{\intercal} q_i, \sigma_R)$.

With this model, we can find the underlying topics of words based on their semantic meanings. This can help us recommend items to users even if the used text is very diverse, which is common in social media. For example, although "fitness" and "exercise" are two different words, in pre-trained word embeddings, their distance is smaller than a random pair of words, so they are more likely to be generated by the same multivariate Gaussian than from different Gaussian distributions. In our model, items whose descriptions contain "fitness" and items whose descriptions contain "exercise" will have similar topic distributions and so are their latent factors. Then for a user who has adopted items with the word "fitness," our model is more likely to recommend items with the word "exercise" to him. Unfortunately, traditional models may not achieve this as they do not consider the semantic meaning of words.

It is worth pointing out that in our CET model, the modified LDA component, which generates word embedding vectors from a mixture of multivariate Gaussian distributions, is almost the same as in a recent work by Das et. al. [6]. However, we developed our model independently and our focus is to apply the model for the purpose of recommendation. Note also that although here we assume the text is associated with each item, our model is not restricted to this setting. If there is text associated with users, our model can also be directly applied by switching the generative process of items with that of users.

Table 1. Notation of our model.

Variable	Description
r_{ui}	Rating of item i by user u
v_w, v_{wi}	Vector of word w learned by word embeddings and corresponding value at the ith dimension
α	The hyper-parameters for the Dirichlet distribution
$\sigma_U, \sigma_I, \sigma_R$	The standard deviation for univariate Gaussian distributions
$\varGamma_U, \varGamma_I$	The covariance matrices for multivariate Gaussian distribution
b_u, b_i	The rating bias of user u and rating bias of item i
p_u, q_i	The latent factor of user u and latent factor of item i
θ_i	The topic distribution of item i
$\mu_t, \varSigma_t$	The mean and covariance matrix for the multivariate Gaussian distribution of topic t
$\mathrm{Dir}(\alpha)$	A Dirichlet distribution with hyper-parameter α
$\mathcal{N}(\mu, \varSigma)$	A Gaussian distribution with mean μ and covariance matrix $\varSigma$
$\mathrm{Multi}(\theta)$	A discrete distribution with θ as parameter

3.3 Parameter Estimation

When applying our model to a dataset, text, ratings and word vectors are all given, and we need to find the hidden parameters that can maximize the posterior likelihood. So, our goal of training is to learn the parameters that can maximize the following probability:

$$p(\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{B}_U, \boldsymbol{B}_I, \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\varSigma} | \boldsymbol{W}, \boldsymbol{R}). \tag{1}$$

Here $\boldsymbol{P}$ and $\boldsymbol{Q}$ refer to all latent vectors for items and users, $\boldsymbol{B}_U$ and $\boldsymbol{B}_I$ refer to bias terms of users and items. $\boldsymbol{W}$ refers to all the words we observe and $\boldsymbol{R}$ refers to all the ratings. $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ represent all means and covariance matrices of the Gaussian distributions of all topics. The hyperparameters are omitted in the formula. As there is no closed form solution for our problem, we use Gibbs-EM algorithm [28] for parameter estimation. For each iteration, we alternate between Gibbs sampling and gradient descent. More specifically, in each iteration, we first perform Gibbs sampling based on parameters learned in the last iteration, which will be fixed in the sampling stage. Then based on the sampled hidden variables, we optimize our objective function using gradient descent.

E-step: We fix the parameters $\boldsymbol{\theta}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and collect samples of the hidden variables $\boldsymbol{Z}$ to approximate the distribution $P(\boldsymbol{Z}|\boldsymbol{W}, \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$. The distribution of the hidden labels for Gibbs Sampling is:

$$P(z_{ij} = t) \propto \theta_{it} \cdot \mathcal{N}(v_{w_{ij}}|\mu_t, \Sigma_t). \tag{2}$$

Here, z_{ij} is the topic assignment of the word at the jth position of text of item i and w_{ij} denotes the corresponding word.

M-step: With the collected samples of $\boldsymbol{Z}$, we need to find the values of $\boldsymbol{P}$, $\boldsymbol{Q}$, $\boldsymbol{B}_U$, $\boldsymbol{B}_I$, $\boldsymbol{\theta}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ that maximize the following objective function:

$$\mathcal{L} = \sum_{\boldsymbol{Z} \in \mathcal{S}} \log P(\boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{R}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{B}_U, \boldsymbol{B}_I, \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\Sigma}|\alpha, \sigma_U, \sigma_I, \sigma_R, \Gamma_U, \Gamma_I), \tag{3}$$

where $\mathcal{S}$ is the set of samples collected in the E-step.

It is noted that θ for any document is constrained to be a multinomial distribution. To transform this constrained optimization problem to an unconstraint one, we use a set of auxiliary variables λ_{it} to replace θ_{it} with $\frac{\exp(\lambda_{it})}{\sum_{t'} \exp(\lambda_{it'})}$. We use gradient descent to find the optimal value of $\boldsymbol{P}$, $\boldsymbol{Q}$, $\boldsymbol{B}_U$, $\boldsymbol{B}_I$, $\boldsymbol{\theta}$. $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be updated using the following equations:

$$\mu_{ti} = \frac{1}{N_{t\cdot}} \sum_{w=1}^{V} N_{tw} v_{wi} \qquad \Sigma_t^{ii} = \frac{1}{N_{t\cdot}} \sum_{w=1}^{V} N_{tw} (v_{wi} - \mu_{ti})^2. \tag{4}$$

where N_{tv} is the number of times word type w is assigned to topic t, $N_{t\cdot}$ is the number of times all word types are assigned to topic t and Σ_t^{ii} is the element at row i, column i of matrix Σ_t.

After all parameters in the model are learned, we use $\hat{r}_{ui} = p_u^{\intercal} q_i + b_u + b_i$ to predict the rating of item i by user u. In our implementation, we perform 600 runs of Gibbs EM. Because Gibbs sampling is time consuming, in each run we only perform one iteration of Gibbs sampling and collect that one sample. We then have 60 iterations of gradient descent. The gradient descent algorithm we use is L-BFGS, which is efficient for large scale data set [22]. We downloaded word vectors from the homepage of word2vector[2] and use them as our word embedding vectors.

[2] https://code.google.com/p/word2vec/

4 Experiment

Our model can be applied to many recommendation tasks on social media where user-generated content plays an important role. To test it, we pick two representative social media platforms for experiments. The first is Meetup[3], an event-based online social network. Meetup allows users to create interest groups and organize events. A commonly studied recommendation task on Meetup is how to recommend an interest group to a user. The second is Amazon's product review platform. We use user-generated product reviews as additional textual information to help product recommendation. The content in these two platforms are also representative. In the online social network website we use, content contains tags given by users. Because there is not a controlled vocabulary of tags and the number of tags assigned to each item can be small, the data is very sparse. In online review website, users can write their reviews in free form. So the content is relatively rich but the diversity is still high.

For each dataset, we use 10% of the data as the development set and another 10% of the data as the testing set. The remaining 80% of the data is used for training. We tune all models according to the development set and test them on the testing set. As our model does not update word embedding vectors. Those words with no pre-trained vectors are of no use to CET. So we just delete them all. The average percentage of words with embedding vectors is 54.7% over all datasets. To show the effectiveness of our model, we choose several appropriate state-of-the-art recommendation techniques for comparison. Besides showing their performance, we also do statistical significance test of results using Wilcoxon signed-rank test.

4.1 Group Recommendation in Meetup

The first experiment is conducted on a Meetup dataset [16]. Meetup is an online event-based social network. In this website, users can build or join groups and each group can organize and publish offline events for people to participate in. Users and groups can use tags to label themselves to show their interests. The text we use is tags associated with groups. The dataset we use is a random sample from the data used in [16]. There are 2225 users, 6950 groups, 8015 user-group membership pairs and each group has 7.06 tags on average. This data is very sparse as only 0.04% of its user-group matrix entries contain values. For this dataset, we only have the information about which groups a user has joined. For the groups the user has not joined, there can be different reasons. The user may not like the group or the user may be unaware of the group at all. This type of negative examples is called implicit feedback. Because of this, we choose two models that work on implicit feedback as our baselines as follows.

[3] http://www.meetup.com

CTR: Collaborative Topic Regression [29] is a model designed for scientific article recommendation with implicit feedback. It assumes that each article's latent factor is a deviation from its topic distribution.

OCF: One-class Collaborative Filtering [23] extends traditional matrix factorization to model implicit feedback. In our experiments, we use the re-weighting technique proposed in this paper.

Quantitative Study. We use MPR (Mean Percentage Ranking) [9] as the evaluation metric. For each user-group pair in our testing data, we randomly select 1,000 "negative" groups and mix them with the "positive" group. We rank all these 1,001 groups based the predicted rating from the target user. Then, we calculate MPR as follows:

$$\mathrm{MPR} = \frac{1}{N}\sum_{i=1}^{N}\frac{R_i}{M}, \tag{5}$$

where R_i is the position of the adopted group in testing pair i and M is the number of ranked groups, which is 1001 in our experiment, and N is the number of pairs in testing data. For a testing instance i, Percentage Ranking (PR) is defined as $\frac{R_i}{M}$, which will be used in the next subsection.

The MPR for CET, CTR and OCF are shown in Figure 1. OCF performs the worst for this dataset. This is because the dataset is too sparse and it is very hard to learn useful item latent vectors purely based on user membership information. By utilizing tag information, CTR can obtain a much better MPR value. CET can even outperform CTR by using word embedding vectors as it utilize the semantic meaning of words. Statistical test shows that CET's performance is significantly better than CTR and OCF at 5% level. It proves that compared with the baselines, our model can learn latent factors much more effectively.

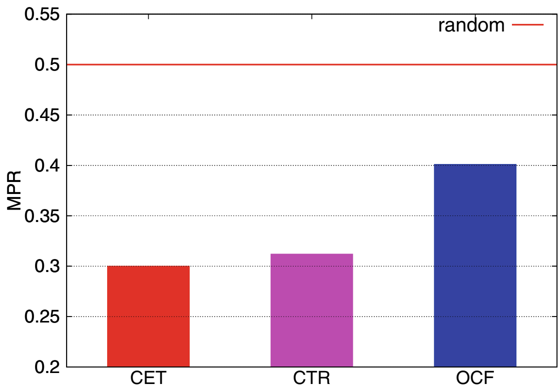

Fig. 1. Mean Average Ranking (MAR) for CET, CTR and OCF on Meetup data.

Qualitative Study. To qualitatively understand how our model outperforms CTR, we display some sample users in Table 2. The representative tags of groups they have joined as indicated in the training data, the tags of "positive" groups in the testing data (i.e. groups that should be recommended) and the corresponding Percentage Ranking (PR) by CET and CTR are also shown together. For user 1408, the tags of groups he has joined tell us that he is interested in exercises and outdoor activities. A recommendation method should rank groups related to this topic higher than others. The group with tags "aerobics" and "running" shows up in our test instances. Our CET model ranks it higher than 98% of the negative examples while CTR only ranks it higher than 42% of the negative examples. The reason is that tags used in social media is very diverse, and groups with similar properties may share no words at all. Traditional way of using lexical similarity to compute textual similarity cannot work very well in this case. So it becomes hard for them to recommend groups based on tags. However, by leveraging words' vector representation, CET can tackle this problem better. The second and third cases also prove this. It is interesting that CET is also able to recommend groups that is conceptually related but have different properties. In the fourth row, we can see that user 399 is interested in Buddhism, and he has also joined a group about vegetarian, which appears in our test dataset. Buddhism is about religion while vegetarian is about food preference. It is impossible to connect them based only on lexical similarity. However, we know that many Buddhists are also vegetarians. So these two words are semantically related and it is reasonable to recommend a vegetarian group to a person interested in Buddhism. While CTR fails to do this, our CET model successfully recommends this group based on using semantic similarity between words.

Table 2. Sampled users, the representative tags of groups they join, the tags of group we should recommend and the percentage ranking of CET and CTR.

User ID	**Tags of groups they have joined**	**Tags of groups we should recommend**	**PR by CET**	**PR by CTR**
1408	fitnees friends music meditation hiking yoga	aerobics running	0.020	0.577
1247	cooking nutrition movies fitness	volleyball	0.135	0.663
835	photo weightloss fitness	theater art museum	0.001	0.528
399	photoshop alternative mediation buddhism	vegetarian nutrition	0.042	0.563

We also show the top words of the topics learned by CET and CTR in Table 3. As we can see, topics learned by CET look much neater. We can find some noisy words in topics learned by CTR. For example, dance and Japanese are in the same topic and hiking and dogs are also in the same topic. Previous work has shown that LDA, which is used to model text in CTR, is not able to learn topics well when documents are very short [32]. The average number of tags for each meetup group is only 7, so it is really hard for LDA to learn good topics. However, by using the embedding vectors, which carry semantic meanings of words, CET can cluster word much better and learn neater and more meaningful topics.

Table 3. Top words of sampled topics learned from Meetup data by CET and CTR.

CET	dance dancing salsa tango salsa-dancing latin-dance flamenco dance-lessons ballet latin-dancing
	hiking excursionismo-hiking kayaking camping outdoors snowshoeing skiing backpacking walkers paddling
	dogs puppy cats pets chihuahua pug yorkie sheltie dachshund dog-lovers
	language culture spanish-culture english french-culture language-and-culture languages japanese-language german-culture european-culture
	movies films movie film movie-nights arthouse movies-dinner movies-and-dinner cinema-and-films dinner-and-a-movie
CTR	dance wellness group-fitness-training japanese dance-lessons cloud-computing english-conversation python democrat korean
	hiking outdoor-recreation startup-ventures javascript creative-writing new-york-city dogs singles-who-love-to-travel activities css
	business-networking weightloss stress foodie crosscultural socializing-dogs dog-lovers london liberty anime
	social language theater bike beer backpackers business-and-social-networking museum rockclimbing men
	fitness movies movie-nights exercise-nutrition business film snowboard cinema-and-films movies-dinner mountain-biking

4.2 Product Recommendation in Online Review Website

For the second experiment, we use data from Amazon, which is composed of 9 datasets used in [19]. We have users' explicit ratings at scale 1-5 of items and their reviews. Similar to [19], we use the aggregated reviews of an item as the associated text of it. Users and items with fewer than 3 reviews are filtered out. Statistics of this type of dataset are shown in Table 4.

We choose two state-of-the-art techniques that model both explicit ratings and text information as our baselines.

HFT: Hidden Factors as Hidden Topics [19] is a model that directly ties each dimension of hidden factors in matrix factorization of ratings to one hidden topic in review text by using an exponential transformation function.

Table 4. Dataset statistics, which show number of users, number of items, number of reviews, total number of word types, average number of tokens per review in each column.

dataset	#users	#items	#reviews	#word types	#tokens/review
office	691	313	4034	12652	46.33
patio	748	344	6814	8691	32.7
software	314	235	2468	14317	83.03
beauty	4281	1817	33290	22208	33.91
sports	8039	5545	91294	37645	30.23
tools	4935	3346	38998	68390	55.14
toys	3479	2776	25951	51224	50.07
games	9919	6124	88684	301829	115.83
health	4529	2460	35123	39674	40.36

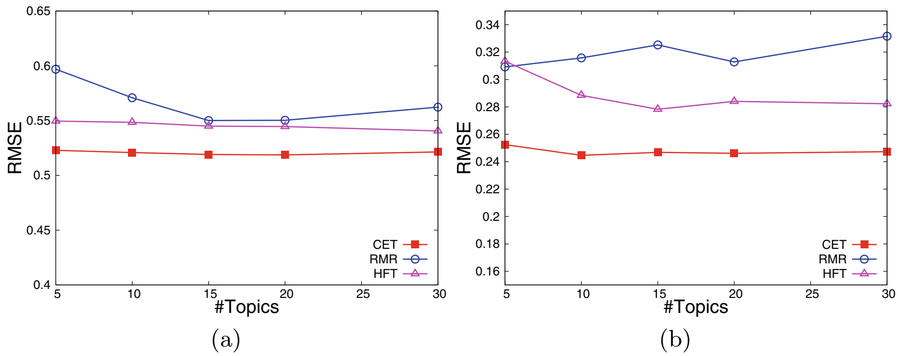

Fig. 2. RMSE over topic numbers on two datasets. The left one is *office* dataset, the right one is *patio* dataset.

RMR: Ratings Meet Reviews [15] is a model similar to HFT except the way they link ratings with reviews. It assumes that each user has one Gaussian rating distribution on each topic, which characterizes how the user is interested in this topic.

Table 5. RMSE of CET, HFT and RMR. For each dataset, the best result is in bold font. † indicates that CET significantly outperforms RMR at 1% level. ‡ indicates that CET significantly outperforms both RMR and HFT at 1% level.

	office	patio	software	beauty	sports	tools	toys	video	health
CET	**0.521**†	**0.252**‡	**0.725**‡	**0.371**‡	**0.215**‡	**0.746**‡	**0.967**‡	1.183	**0.483**‡
RMR	0.597	0.309	0.767	0.484	0.351	0.802	1.013	**1.138**	0.595
HFT	0.552	0.283	0.776	0.444	0.262	0.813	1.146	1.172	0.548

Table 6. Top words of sampled topics learned by CET, HFT and RMR.

Topics learned by CET					
work	tape	paper	product	pages	daughter
office	file	binder	products	templates	old
job	files	printed	price	interface	home
desk	tapes	printing	buy	page	son
working	folder	printer	purchase	multiview	mother
phone	folders	print	buying	text	father
cabinet	video	binders	brand	functionality	niece
offices	taped	pencil	purchasing	webpage	grandmother
works	filing	sheets	brands	app	granddaughter
telephone	clips	ink	pricing	template	dad
Topics learned by HFT					
desk	folders	binder	pen	cards	black
keyboard	tabs	binders	markers	paper	color
mouse	folder	rings	fine	card	folders
pad	file	pages	pens	print	look
hp12c	reinforced	open	colors	business	good
feet	tab	one	ink	avery	colors
rest	manila	pockets	write	printer	great
wrist	use	ring	sharpie	printed	nice
holder	smead	front	use	quality	side
platform	box	plastic	highlighters	make	well
Topic learned by RMR					
desk	folders	binder	markers	cards	folders
keyboard	files	binders	colors	paper	black
mouse	hanging	rings	pens	card	color
pad	using	open	ink	avery	look
rest	still	pockets	pen	print	file
wrist	drawer	pages	sharpie	business	good
holder	pendaflex	ring	highlighters	printer	great
feet	bottom	front	great	printed	nice
platform	product	avery	write	make	one
tray	capacity	cover	marker	professional	colors

Quantitative Study. For Amazon review dataset, we use RMSE (Root Mean Squared Error) [19] as the evaluation metric, which is defined as:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{r}_i - r_i)^2}, \tag{6}$$

where r_i is the true rating for the ith testing instance and $\hat{r}_i$ is the prediction. The results over all 9 datasets are shown in Table 5. We can see that CET significantly outperforms RMR and HFT on most datasets. It means our model can effectively learn users' interests by modeling rating and text information.

To have a closer look at how the performance of all three models change over different number of topics, we pick two datasets *office* and *patio* and show the results in Figure 2. We can see that CET outperforms both baselines when using different numbers of topics. Its performance is also more stable over topic numbers compared with the other two.

Qualitative Study. In this subsection, we show the top words of some sampled topics learned by CET, RMR and HFT in Table 6. All topics are from the *office* domain and the number of topics is set to 30 for all models. As we can see, CET can learn meaningful topics like office, file, paper, purchase and so on as well as HFT and RMR. By taking a closer look at the top words of these topics, we can find that the top words of CET are cleaner. Most of the top words are about the same topic and there is less noise in these words. However, there exist some noisy words in the top word list of HFT and RMR, many of them are general words like "one", "use", "well", etc. By using word vector to represent words, words can be clustered better compared with models like HFT and RMR. It is interesting that CET also discover a topic, family members, which cannot be learned by RMR and HFT. This may be a topic worth mining for recommendation as it probably reflects who the product is bought for. However, CET is not perfect and it fails to discover the topic about pens.

5 Conclusions and Future Work

In this work, we have proposed a recommendation model for social media based on users' ratings, text and word embedding vectors. Compared with existing work, our model is able to find the similarity between two pieces of text based on their semantic similarity rather than simply lexical similarity. This makes it more effective for recommendation problems in social media. Extensive experiments on two recommendation problems in social media show that this model can outperform state-of-the-art methods. A closer look at topics also tells us that by using the semantic meanings reflected in embedding vectors, our model can learn cleaner topics. When documents have as few as 7 words on average, our model can still learn meaningful topics and get good recommendation results.

We have shown that using vectors learned from neural network based model can improve both recommendation and topic discovery in social media. It would be interesting to try vectors learned by other word embedding models such as topical word embedding [17], multi-prototype word embedding [21] and so on. Besides, it is a promising direction to model text in other ways beyond bag of words. Models which take order into consideration, like Recursive Neural Network [27] and Convolutional Neural Networks [10] are worth trying.

References

1. Baltrunas, L., Makcinskas, T., Ricci, F.: Group recommendations with rank aggregation and collaborative filtering. In: Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 119–126 (2010)

2. Bao, Y., Fang, H., Zhang, J.: TopicMF: Simultaneously exploiting ratings and reviews for recommendation. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 2–8 (2014)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine learning research **3**, 993–1022 (2003)
4. Chen, G., Chen, L.: Recommendation based on contextual opinions. In: Dimitrova, V., Kuflik, T., Chin, D., Ricci, F., Dolog, P., Houben, G.-J. (eds.) UMAP 2014. LNCS, vol. 8538, pp. 61–73. Springer, Heidelberg (2014)
5. Clinchant, S., Perronnin, F.: Aggregating continuous word embeddings for information retrieval. In: Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality, pp. 100–109 (2013)
6. Das, R., Zaheer, M., Dyer, C.: Gaussian LDA for topic models with word embeddings. In: Proceedings of the 53nd Annual Meeting of the Association for Computational Linguistics (2015)
7. Gorla, J., Lathia, N., Robertson, S., Wang, J.: Probabilistic group recommendation via information matching. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 495–504 (2013)
8. Guo, G., Zhang, J., Yorke-Smith, N.: Trustsvd: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 123–129 (2015)
9. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, pp. 263–272 (2008)
10. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pp. 1749–1751 (2014)
11. KgebŁck, M., Mogren, O., Tahmasebi, N., Dubhashi, D.: Extractive summarization using continuous vector space models. In: Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality, pp. 31–39 (2014)
12. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**, 30–37 (2009)
13. Krestel, R., Fankhauser, P., Nejdl, W.: Latent dirichlet allocation for tag recommendation. In: Proceedings of the Third ACM Conference on Recommender Systems, pp. 61–68 (2009)
14. Kywe, S.M., Lim, E.-P., Zhu, F.: A survey of recommender systems in twitter. In: Aberer, K., Flache, A., Jager, W., Liu, L., Tang, J., Guéret, C. (eds.) SocInfo 2012. LNCS, vol. 7710, pp. 420–433. Springer, Heidelberg (2012)
15. Ling, G., Lyu, M.R., King, I.: Ratings meet reviews, a combined approach to recommend. In: Proceedings of the 8th ACM Conference on Recommender Systems, pp. 105–112 (2014)
16. Liu, X., He, Q., Tian, Y., Lee, W.C., McPherson, J., Han, J.: Event-based social networks: linking the online and offline social worlds. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1032–1040 (2012)
17. Liu, Y., Liu, Z., Chua, T., Sun, M.: Topical word embeddings. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2418–2424 (2015)
18. Ma, H., Yang, H., Lyu, M.R., King, I.: Sorec: social recommendation using probabilistic matrix factorization. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 931–940 (2008)

19. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: Understanding rating dimensions with review text. In: Proceedings of the 7th ACM Conference on Recommender Systems, pp. 165–172 (2013)
20. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems **26**, 3111–3119 (2013)
21. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1059–1069 (2014)
22. Nocedal, J.: Updating QUASI-Newton matrices with limited storage. Mathematics of Computation **35**(151), 773–782 (1980)
23. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R., Scholz, M., Yang, Q.: One-class collaborative filtering. In: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, pp. 502–511 (2008)
24. Pennacchiotti, M., Gurumurthy, S.: Investigating topic models for social media user recommendation. In: Proceedings of the 20th International Conference Companion on World Wide Web, pp. 101–102 (2011)
25. Pennington, J., Socher, R., Manning, C.: Glove: Global Vectors for Word Representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
26. Shi, Y., Larson, M., Hanjalic, A.: Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. ACM Computation Survey **47**(1), 3:1–3:45 (2014)
27. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631–1642 (2013)
28. Wallach, H.M.: Topic modeling: Beyond bag-of-words. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 977–984 (2006)
29. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 448–456 (2011)
30. Wang, F., Pan, W., Chen, L.: Recommendation for new users with partial preferences by integrating product reviews with static specifications. In: Carberry, S., Weibelzahl, S., Micarelli, A., Semeraro, G. (eds.) UMAP 2013. LNCS, vol. 7899, pp. 281–288. Springer, Heidelberg (2013)
31. Wang, H., Wang, N., Yeung, D.: Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1235–1244 (2015)
32. Yan, X., Guo, J., Lan, Y., Cheng, X.: A biterm topic model for short texts. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1445–1456 (2013)
33. Zhang, W., Wang, J., Feng, W.: Combining latent factor model with location features for event-based group recommendation. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 910–918 (2013)