Solving the Cold Start Problem Using Product Related Contents

Shi Hu
s3hu@cs.stanford.edu
Group Number: 10

1 Introduction

To make rating predictions, one of the most commonly used approach in recommender systems is the latent factor model. Despite its popularity, one of its drawbacks is that it only makes use of numeric ratings but ignores other resources, such as review texts. McAuley et al. [1] proposed a general framework to employ both numeric ratings and review texts, and showed their model can outperform the latent factor model in various types of products. Although they found reviews to be a highly informative text source, reviews are not available in many recommendation settings, nor are they available for new users and items. To address this cold start problem, we based our work on the framework they developed, and experimented with other product related contents to replace review texts.

2 Prior Work

There have been a few researchers whose recommendation frameworks are based primarily on texts rather than ratings. For example, Moghaddam et al. [2]'s Factorized LDA (FLDA) model is a probabilistic graphical model on both items and reviewers which (1) extracts major aspects from review texts, where aspects are attributes or components of items, and (2) predicts ratings of each aspect based on the reviews' polarity on that aspect. Their model is also chiefly designed to address the cold start problem. Specifically, it is trained on the category level instead of the item level, and uses a variational EM algorithm to find approximate estimates of FLDA parameters. In this algorithm, the non cold start items can serve as prior for the aspect and rating distributions for the cold start items.

Similar to [2], Levi et al. [3] also uses the NLP techniques to analyze the review texts. Additionally, the context of the reviews plays an important role in [3]'s model. When a user searches for a hotel, she needs to fill out three pieces of information: trip intent, nationality, and preferences for hotel aspects including *location, service, food, room, price-value quality and facilities*. This is useful because customers under analogous contexts usually rate items similarly. To compute the scores of hotels, the recommender first computes the scores of each review based on the context group. This is done by identifying the features (nouns or noun phrases) in each review, then assigning weights to the features based on both the polarity of the opinions in the reviews as well as the contexts. The feature scores are aggregated together and the result is the score of the review. The score of a

hotel is just the average score of all its review scores adjusted by the bias of the contexts.

To summarize, both papers share the common interest to our research, which is to tackle the cold start problem. However, our perspectives are quite different. Their approaches focus on exploiting the existing review texts, ours focus on experimenting other relevant resources to replace review texts in case they are not present.

3 Model

3.1 The Mathematics behind the HFT Model

[1] introduces the "Hidden Factors as Topics" (HFT) model, which employs both numeric ratings and review texts when making rating predictions. The underlying mathematics is explained below.

Traditionally, the product ratings can be predicted using the latent factor model. For a user u and an item i, the predicted score is

$$rec(u,i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i \tag{1}$$

Here, α is the average of all review scores, β_u and β_i are user and item biases deviated from the overal average. γ_u and γ_i and the K-dimensional latent user and item factors. The goal is to get the optimal values of parameters $\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i$, such that the differences between the predicted ratings and real ratings are minimum by some distance metric. We can also add a regularizer $\Omega(\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i)$ to the objective function to penalize complex models, that is,

$$f(\mathcal{T}|\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i,) = \sum_{r_{u,i} \in \mathcal{T}} (rec(i,u) - r_{u,i})^2 + \lambda\Omega(\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i), \tag{2}$$

where $\mathcal{T}$ is the training corpus of ratings, and λ balances the importance of the two terms.

In addition, LDA is one of the standard topic models to extract topics from text. Here, LDA is used to uncover hidden dimensions in review text. The topics are obtained by maximizing the corpus likelihood, where the corpus is the concatenation of all review texts written for an item, or similarly, all review texts written by a user. The likelihood of a corpus $\mathcal{T}$ is

$$p(\mathcal{T}|\theta, \phi, z) = \prod_{d \in \mathcal{T}} \prod_{j=1}^{N_d} \theta_{z_{d,j}} \phi_{z_{d,j}, w_{d,j}} \tag{3}$$

In this equation, d is one piece of review text, $z_{d,j}$ assigns the j-th word in the review d to a topic. θ is the topic distribution and $\theta_{z_{d,j}}$ is the likelihood of seeing topic $z_{d,j}$ among all topics. $w_{d,j}$ is the j-th word in review d. ϕ is the word distribution and $\phi_{z_{d,j}, w_{d,j}}$ the likelihood of seeing word $w_{d,j}$ among all words in topic $z_{d,j}$.

In HFT, the latent factor γ_i and topic distribution θ_i are linked together by the transformation $\theta_{i,k} = \frac{exp(\kappa\gamma_{i,k})}{\sum_{k'} exp(\kappa\gamma_{i,k'})}$. Here, $k \in \{1 \dots K\}$ is the k-th topic of the corpus. The benefit of this transformation is it preserves monotonicity, that is, the n-th largest value in γ_i corresponds to the n-th largest value in θ_i, $\forall n \in \{1 \dots K\}$. Combining equation (1) with the log likelihood of (2), we get the final objective function

$$f(\mathcal{T}|\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i, \theta, \phi, \kappa, z) = \sum_{r_{u,i} \in \mathcal{T}} (rec(i,u) - r_{u,i})^2 - \mu l(\mathcal{T}|\theta, \phi, z). \tag{4}$$

The goal is to minimize the prediction errors while maximize the corpus likelihood. It is important to note the the topic distribution θ can also link to *user* latent factor γ_u. This property is useful when we do cross domain prediction described in Section 3.2.

Comparing equations (2) and (4), we see there are two types of regularizers, namely, $\Omega(\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i)$ in (2) and $l(\mathcal{T}|\theta, \phi, z)$ in (4). They are very different regulaziers and thus we did not use them together. That means, when trying out the best value for λ, we set μ to zero, and vice versa. It is worth noting that setting λ to zero means we are using the HFT model, and setting μ to zero means we are using the latent factor model.

3.2 The Modifications to the HFT Model

There are a couple of changes made to the existing codebase. First, the original model only reads in a ".votes" file that contains the following information for each item: user ID, item ID, a numeric rating and some review texts. The new model reads in a second file that contains relevant descriptions about those items. We remove the reviews in the ".votes" file where the items don't have descriptions. Here is a table presenting the datasets we will examine:

dataset	**# items**	**# reviewers**	**# reviews**	**# words in reviews**
All Amazon Catalogs	1,493,275	5,845,399	25,176,095	3,235,787,325
Amazon Movie	41,393	765,514	3,198,427	478,439,557
Amazon Music	4,055	209,121	489,672	71,531,530
RateBeer [4]	72,052	28,296	2,570,183	137,226,355
Epinions [5]	202	7,454	8,730	465,105

Table 1: Datasets

The reason why Amazon Movie and Music are listed separately from All Amazon Catalogs is because we use additional descriptions to evaluate these two catalogs, in addition to their product descriptions from Amazon. Specially, here is a list of product descriptions we use for each dataset in Table 1:

- All Amazon Catalogs (40 catalogs): product descriptions and editorial reviews from Amazon
- Amazon Movie: movie storylines and all plot keywords from IMDb [6]
- Amazon Music: music lyrics from Absolute Lyrics [7]
- RateBeer: beer descriptions from RateBeer
- Epinions: user self introductions from Google Plus [8]

Since it is possible that none of these descriptions is available in real world, we also experiment with cross domain prediction. Concretely, suppose we could not find any relevant product related data in a catalog, then we use user reviews in *other* catalogs as the descriptions to predict the target catalog. For example, if we want to recommend books to a group of users, but none of them has written any review to any book, but many have reviewed some *movies*, then we can plug in those users' movie reviews as the user descriptions to help estimate their book ratings, as the HFT model can work with both item and user descriptions. We do this research on the Amazon dataset. We only keep the numeric ratings and remove all review texts of all items in every catalog. The goal is

to mimic the situation where users have only *rated* some products but have not written any review text. For all users who have made some reviews in this catalog, we build their user descriptions by concatenating together all their reviews written for another catalog. This produces $2 \times \binom{40}{2} = 1560$ new datasets, as Amazon has 40 catalogs. We remove the reviews in the new datasets where the reviewers have no user descriptions, and subsequently remove the catalogs that have less than 1000 reviews. We are left with 951 new datasets in the end.

Second, we create a map data structure which maps each item ID to its product description as a list of words. Then we randomly assign a topic to each word in the description and obtain the initial topic distribution according to this random initialization.

Third, we use L-BFGS optimization method to optimize the parameters in the energy function (4) for the HFT model. After every 50 gradient iterations, we use Gibbs sampling to sample the most likely topic of each word and update the topic counts if the word has a new topic according to the sampling. This update will later change the topic distribution θ_i in the LDA component. Since θ_i and γ_i are linked together as described in Section 3.1, the latent factor γ_i's are consequently updated as well.

Fourth, we iterate the main loop 50 times, where each iteration does the third step above. In the end, we pick the parameters obtained from the iteration that produces the lowest validation error as the optimal parameters and record the corresponding test error.

4 Results

4.1 Rating Errors

The average rating errors for the five main datasets are shown in Table 2. The rating errors for all Amazon catalogs are shown in Table 9 in Appendix.

dataset	**latent factors**	**HFT (prod. desc.)**	**HFT (review texts)**
All Amazon Catalogs	1.5253	1.4244	1.4214*
Amazon Movie	1.0338	1.0169*	1.0547
Amazon Music	1.2526	1.2207	1.2148*
RateBeer	0.2967	0.2985	0.2965*
Epinions	1.1334	1.0766	1.0658*

Table 2: Average Rating Errors (the best performing model on each dataset is starred)

We can see that the HFT model using product descriptions outperforms the latent factor model in all datasets except for RateBeer, and it is nearly as good as HFT with review texts. I think the primary reason why it fails on RateBeer is because the beer descriptions are too elementary. Specifically, there are many one-line descriptions such as "2 cask regular", or "1 bottled". Further, many beers have the same descriptions. This will inevitably cause confusions in the system. These results indicate that if the descriptions or reviews are inadequate quantitatively and/or qualitatively, we are better off not having them.

For other datasets, the words in reviews and descriptions may be generated from similar distributions. As a consequence, for many datasets (including the ones in Table 9), the rating errors of the two HFT models are very similar. For the ones that have noticeable discrepancies (also see Table 9), I think the gaps might come from the fact that product descriptions in general only

express neutral to positive sentiments, while the sentiments in reviews could deviate a lot from these descriptions.

Next, we will show the results from cross domain prediction. Due to the page limit, we will only show a subset of the results for illustration purpose. The full result is available upon request.

	arts	baby	books	clothing	grocery	kitchen	jewelry	movies	music	shoes
arts	·	·	1.38	0.827	0.982	1.389	·	1.222	1.342	1.316
baby	·	·	1.505	·	·	1.57	·	1.593	2.075	·
books	0.742	1.198	·	0.836	0.887	0.854	0.9	0.805	0.744	0.84
clothing	0.782	0.956	0.467	·	0.579	0.447	0.619	0.445	0.525	0.541
grocery	1.311	·	1.327	1.295	·	1.41	1.408	1.295	1.321	1.309
kitchen	1.43	1.531	1.477	1.365	1.444	·	1.288	1.519	1.51	1.258
jewelry	0.822	·	1.143	1.127	1.271	1.297	·	1.017	1.239	1.409
movies	0.938	1.265	0.793	0.945	0.9	0.835	0.933	·	0.81	0.994
music	0.739	0.874	0.737	0.881	0.868	0.852	0.82	0.755	·	0.972
shoes	0.62	·	0.296	0.398	0.514	0.393	0.54	0.355	0.333	·

Table 3: Cross Domain Prediction using Latent Factor Model (catalogs that have less than 1000 reviews are omitted as "·")

	arts	baby	books	clothing	grocery	kitchen	jewelry	movies	music	shoes
arts	·	·	1.349	0.745	0.951	1.371	·	1.222	1.327	1.198
baby	·	·	1.455	·	·	1.498	·	1.462	1.734	·
books	0.646	1.035	·	0.767	0.846	0.835	0.809	0.866	0.735	0.776
clothing	0.424	0.737	0.369	·	0.421	0.376	0.387	0.361	0.351	0.363
grocery	1.101	·	1.251	1.16	·	1.343	1.228	1.222	1.258	1.136
kitchen	1.26	1.37	1.381	1.255	1.334	·	1.154	1.383	1.395	1.188
jewelry	0.82	·	1.057	0.987	1.149	1.18	·	0.961	1.126	1.309
movies	0.873	1.185	0.937	0.926	0.906	0.851	0.869	·	0.899	0.958
music	0.637	0.841	0.779	0.818	0.816	0.827	0.748	0.819	·	0.891
shoes	0.245	·	0.241	0.238	0.35	0.311	0.325	0.263	0.256	·

Table 4: Cross Domain Prediction using HFT with Descriptions (catalogs that have less than 1000 reviews are omitted as "·")

In Table 3 and 4, each entry represents the rating error on the row catalog using the column catalog as the description. As we can see, the majority of entries in Table 4 is less than the corresponding ones in Table 3. This may suggest the reviews written by the same customer tend to have similar word distributions no matter what product they review. Thus, we can further model a user by carefully examining their writings to infer their word distributions in reviews.

4.2 Top Words

One of the benefits of the original HFT model is that it can discover the underlying topics and associate them with textual labels. Because the review topics and latent factors are linked together,

we can visualize the latent factors through the labels. In some sense, those factors are the aspects of products that customers pay attention to. To illustrate this facility, here are two examples produced by HFT using descriptions:

learning	antivirus	tool	photo	office
skills	spyware	photo	cs3	documents
kids	antivirus	naturallyspeaking	gps	pdf
learning	mcafee	greeting	indesign	2007
math	viruses	appointments	illustrator	microsoft
adventure	threats	lt	adobe	office
grammar	protection	calendars	topo	document
vocabulary	guitar	paperport	golive	excel
lessons	virus	filemaker	poser	server
game	hackers	corel	photoshop	quickbooks
rosetta	ipod	clip	cs2	visio

Table 5: software (Amazon)

binder	knife	phone	document	electronics
wilson	xacto	handsets	wowpad	statistics
jones	avery	handset	dictionary	bush
binders	sharpeners	caller	wowpads	boomerang
binder	rolodex	call	buddy	daytimer
safco	sharpener	headset	stickers	solar
nogap	averycomtemplates	waiting	chalkboard	talkswitch
hp	merge	belt	anti	scientific
polypropylene	sharpening	id	registering	calculations
officemate	helical	dialing	selectip	conversions
dring	electric	answering	definition	ipod

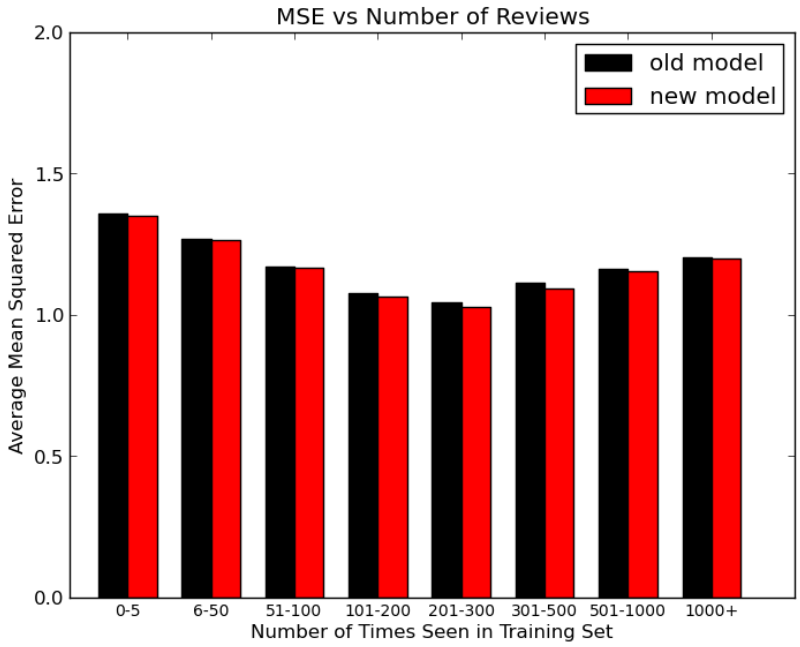
Table 6: office products (Amazon)

Each column in Table 5 and 6 contains 10 most representative words of a topic for software and office products. The top row in each table are the topics that we manually summarized for the corresponding columns below them. These results show HFT with descriptions is also able to group words in descriptions by the words' nature.

4.3 A Comparision of the Models

Next, we show a plot that compares the performance of HFT using reviews (old model) versus HFT using descriptions (new model). We experiment on the Amazon dataset. It is important to note that the quality of the descriptions from Amazon is quite good in general, and a lot of them contain editorial reviews, which can be an excellent replacement for customer reviews. To do the experiment, we first calculate the mean squared error for each item in the test set and count their occurences in the training set. Then we group items together based on the number of times they

appear in the training set. For all items within the same group, we calculate the average mean squared errors.



# times seen in train set	# items
0 - 5	541,171
6 - 50	250,254
51 - 100	23,910
101 - 200	11,852
201 - 300	3,367
301 - 500	1,739
501 - 1000	832
1000+	188
total	833,313

Table 7: # items in each group

The bar chart shows HFT with descriptions slightly outperforms HFT with reviews in all groups. It is worthwhile to mention that this outcome doesn't contradict to the results in Section 4.1, because the latter simply computes the mean of rating errors of all catalogs and ignores the number of items in each catalog. HFT with descriptions performs a bit better in music and movie catalogs, which contain more items than most others.

The bar chart reveals several pieces of information. First, if we've seen an item too little or too many times, the HFT model doesn't perform too well. It works best when we've seen an item

200 to 300 times. This means too little or too many ratings can both impair the association of topics and latent factors. It is fair to say that we may need more data to conclude that too many ratings can harm the system, as there are much less observations in the last group than the first. Second, descriptions can be more helpful than review texts because they are more accurate and comprehensive.

5 Conclusions

We've shown an empirical study of the HFT model using various data sources to replace review texts. We examine the new model's performance and compare it with the original one. The results demonstrate that when the data are descriptive, the HFT model can perform well no matter where the data comes from. Further, the model is most helpful after seeing a moderate number of ratings.

6 Future Work - A Network Approach

Some recommender systems contain social networks. An empirical study is made on LibraryThing [9]. Our dataset contains 110,592 users, 385,252 books, 219,790 pairs of friends and around 1,707,071 reviews. We randomly select 20,000 pairs of people and measure their Jaccard similarity of items that they reviewed. Also, for each item, we randomly select a pair of people who have reviewed this item and compare their rating difference.

	Jaccard sim.	rating diff.
friends	0.0094	0.7218
non friends	0.0004	0.9558

Table 8: network effect

As we can see, friends tend to review more common items than strangers, and have more similar opinions to the items they review. Therefore, one possible approach to improve the HFT model in the future is to incorporate the friends information if they are available.

7 Acknowledgements

I would like to thank Dr. Julian McAuley for the mentorship and for providing some of the datasets, as well as Prof. Jure Leskovec for the advice.

8 Appendix

catalog	latent factors	HFT (prod. desc.)	HFT (review texts)
all electronics	2.34930	2.08932	2.08678*
appliances	1.80156	1.65004	1.64838*
arts	1.53992	1.39309*	1.40862
automotive	1.57750	1.42441	1.42360*
baby	1.70517	1.44148	1.43226*
baby products	1.58683	1.43274*	1.44650
beauty	1.41913	1.36925	1.35331*
books	1.16539*	1.20276	1.20263
camera and photo	1.45315	1.45579	1.39264*
cell phones and accessories	2.33340	2.15835*	2.16595
clothing and accessories	0.37011	0.33930*	0.34038
computers	2.11756	2.00394	1.95713*
electronics	1.78175	1.68073*	1.68764
gift cards store	0.58837	0.56770*	0.57297
gps and navigation	1.74453	1.75035	1.71348*
grocery and gourmet food	1.57616	1.48951	1.48926*
health and personal care	1.57338	1.49538	1.48581*
home improvement	1.82385	1.67527	1.67177*
home and kitchen	1.63565	1.53646	1.53476*
industrial and scientific	0.45750	0.42118	0.41710*
jewelry	1.33594	1.25841*	1.26228
kindle store	1.51603	1.40763	1.40127*
kitchen and dining	1.89495	1.69540	1.68369*
magazine subscriptions	1.08054	1.08146	1.07724*
movies and tv	1.05911	1.04873*	1.08094
mp3 players and accessories	2.40070	2.20656	2.15911*
music	0.98139	0.96419*	0.99579
musical instruments	1.48939	1.36047	1.35968*
office products	1.74662	1.60504*	1.61510
office and school supplies	1.82758	1.77352*	1.84429
patio	1.83726	1.68538*	1.69105
pet supplies	1.69999	1.57354	1.56783*
purchase circles	1.35133	1.35230	1.29237*
shoes	0.26040	0.21819*	0.22217
software	2.39923	2.18528*	2.19253
sports and outdoors	1.22505	1.14025*	1.14172
tools and home improvement	1.59718	1.50947	1.50200*
toys and games	1.47658	1.36176*	1.36277
video games	1.61673	1.49187*	1.49941
watches	1.54864	1.47879	1.47500*

Table 9: Rating Errors of All Amazon Catalogs (the best performing model on each dataset is starred)

References

[1] Julian McAuley, Jure Leskovec, *Hidden factors and hidden topics: understanding rating dimensions with review text*, in RecSys, 2013.

[2] Samaneh Moghaddam, Martin Ester, *The FLDA Model for Aspect-based Opinion Mining: Addressing the Cold Start Problem*, in WWW, 2013

[3] Asher Levi, Osnat Mokryn, Christophe Diot and Nina Taft, *Finding a Needle in a Haystack of Reviews: Cold Start Context-Based Hotel Recommender System*, in RecSys, 2012

[4] http://www.ratebeer.com

[5] http://www.epinions.com

[6] http://www.imdb.com

[7] http://www.absolutelyrics.com

[8] https://plus.google.com

[9] http://www.librarything.com