Sentiment Analysis of User Comments for One-Class Collaborative Filtering over TED Talks

Nikolaos Pappas
Idiap Research Institute
Rue Marconi 19
CH-1920 Martigny, Switzerland
nikolaos.pappas@idiap.ch

Andrei Popescu-Belis
Idiap Research Institute
Rue Marconi 19
CH-1920 Martigny, Switzerland
andrei.popescu-belis@idiap.ch

ABSTRACT

User-generated texts such as reviews, comments or discussions are valuable indicators of users' preferences. Unlike previous works which focus on labeled data from user-contributed reviews, we focus here on user comments which are not accompanied by explicit rating labels. We investigate their utility for a one-class collaborative filtering task such as bookmarking, where only the user actions are given as ground truth. We propose a sentiment-aware nearest neighbor model (SANN) for multimedia recommendations over TED talks, which makes use of user comments. The model outperforms significantly, by more than 25% on unseen data, several competitive baselines.

Keywords

Rating Inference, One-Class Collaborative Filtering

1. INTRODUCTION

Many problems in collaborative filtering (CF) such as social bookmarking, news and video recommendations make use of binary user ratings in terms of 'action' or lack thereof, i.e. 'inaction'. The difficulty of such one-class CF problems [6] comes from the lack of a negative class: it is inherently unsure whether user inaction means that an item was not seen or was not liked (hence not bookmarked). In this paper, we study the one-class CF problem of lecture recommendation over TED talks. We show how to infer additional user ratings by performing sentiment analysis (SA) of user comments and integrating its output in a nearest neighbor (NN) model. The resulting SANN model outperforms several competitive baselines, is robust to noisy SA results and improves its performance with the number of comments.

Work on inferring user ratings has mostly focused on reviews with explicit ratings [3, 11, 4]. Here, however, we show that unlabeled comments can also be leveraged to improve recommendations, in a challenging one-class setting. Other studies have used comments to enrich user profiles for news recommendation, but without attempting sentiment analysis and the inference of item ratings [9, 5].

2. SENTIMENT ANALYSIS

The first stage of our proposal is the sentiment classification of user comments, with two possible labels: positive (*pos*) and negative (*neg*). Given the lack of ground-truth labels, we focused on dictionary-based methods and specifically on an extension of the rule-based sentiment classifier presented in [10] and implemented in [7][1]. The classifier uses the MPQA polarity lexicon and can deal with negation, intensifiers, and polarity shifters. The rule-based classifier estimates the polarity of a sentence as a positive or negative numerical value. This value determines the sentiment label, *pos* or *neg* of the sentence; for zero, the neutral label (*neu*) is applied. The polarity of a comment is the sum over the polarities of the sentences that compose it, and its label (*pos* or *neg*) is given by the sign of the total.

To test the sentiment analysis component, we performed human labeling of a subset of the TED comments, with *pos*, *neg* or *neu* polarity labels (the latter included also undecided cases)[2]. Six human judges annotated 160 comments with 320 sentences, randomly selected from the TED data, with some overlap to assess agreement, using Fleiss' kappa (κ) metric. We obtained 260 labels for sentences and 135 for comments (excluding *neu*). Among these, 61 sentences and 29 comments were common across annotators, and agreement was found to be, respectively, $\kappa = 0.834$ and $\kappa = 0.650$. As agreement was substantial, we used the entire set as ground truth to evaluate automatic sentiment analysis (cases of disagreements were reconciled by a majority vote).

The results of our rule-based classifier (RB) and of a random baseline (Rand) are shown in Table 1. Our system reaches F-score of 74.90% and 72.60% on sentences and respectively comments for the classification task (plus moderate agreement with the ground truth of $\kappa = 0.53$ and $\kappa = 0.43$), a level that is sufficient to improve the one-class recommendation task, as we will show.

3. ONE-CLASS CF MODELS

The one-class collaborative filtering problem can be formalized as follows. Let U be the set of users of size $|U| = M$ and I the set of items of size $|I| = N$. The matrix of user-item ratings is R (of size $M \times N$), with $r_{ui} = 1$ indicating an 'action' rating (favorite item) and $r_{ui} = 0$ an 'inaction' one (not seen or not liked). Our goal is to predict the preference of the users in the future, therefore (as in previous studies) we hide for evaluation a certain proportion of '1' values per user and measure how well we predict them.

[1] https://github.com/nik0spapp/unsupervised_sentiment
[2] https://github.com/nik0spapp/TED_sentiment_labels

	Sentences (*pos*=123, *neg*=137)				**Comments (*pos*=76, *neg*=59)**			
Methods	**Precision**	**Recall**	**F**	**Kappa**	**Precision**	**Recall**	**F**	**Kappa**
Rule Based (RB)	73.43	76.42	74.90	0.53	75.71	69.73	72.60	0.43
Baseline (Rand)	47.36	48.64	47.90	-0.01	56.24	53.42	54.63	-0.02
Annotators	-	-	-	0.83	-	-	-	0.65

Table 1: Performance of annotators, ruled-based and random classifier measured with Precision, Recall, F1 and Fleiss' kappa (κ). Inter-annotator agreement is substantial ($\kappa \geq 0.65$).

3.1 Neighborhood Models

Nearest neighbor (NN) models are often used in collaborative filtering and have been proven quite effective despite their simplicity [2]. Here, we use item-based neighborhood models which are described by Eq. 1. The prediction function $\hat{r}_{ui}$ estimates the rating of a user u for an unseen item i. It relies on the bias estimate b_{ui} of the user u for various items j (given in Eq. 2) and on a score computed using the k most similar items to i that the user u has already rated, i.e. the neighborhood $D^k(u;i)$. The denominator is a normalization factor.

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j \in D^k(u;i)} d_{ij}(r_{uj} - b_{uj})}{\sum_{j \in D^k(u;i)} d_{ij}} \tag{1}$$

The bias estimate b_{ui} is computed as the sum of the average ratings μ of items in a given dataset, the average rating b_u of a user u and the average rating b_i for a given item i. The coefficient d_{ij} is the similarity between item i and item j (Eq. 2) and is computed using the similarity s_{ij} between items i and j, weighted by the normalized importance of the number of common raters n_{ij} (close to 1 if $n_{ij} \gg \lambda$). We determine the optimal values of λ and of the neighborhood size k using cross-validation.

$$d_{ij} = s_{ij}\frac{n_{ij}}{n_{ij} + \lambda}; \; b_{ui} = \mu + b_u + b_i \tag{2}$$

The similarity s_{ij} can be computed using a function such as cosine similarity or Pearson's correlation between vectors representing i and j in the $N \times N$ co-rating matrix derived from R. However, given that ratings are binary and not real-valued, the biases should not be computed linearly, but using the formulas proposed below, which take into account the number of the items and the maximally rated items.

$$\mu = \frac{1}{|I|}\sum_{i \in I}\frac{r_i}{r_{max}}; \; b_u = \frac{r_u}{|I|}; \; b_i = \frac{r_i}{r_{max}} \tag{3}$$

3.2 Sentiment-Aware Neighborhood Model

We propose a sentiment-aware nearest neighbor model (SANN) which integrates into a NN model the preferences of the users that are extracted from user-generated text by sentiment analysis. In order to achieve that, the model in Eq. 1 must be modified as follows: firstly, the neighborhood $D^k(u;i)$ must account for the additional training data, and secondly, the rating function r_{uj} must map the output of sentiment analysis over comments to rating values that are understandable by the model. Thus, we modify the Eq. 1 of the traditional neighborhood models as follows:

$$\hat{r}_{ui} = b_{ui} + \sum_{j \in D^k_c(u;i)} d_{ij}(r'_{uj} - b_{uj}) \tag{4}$$

In this equation, $D^k_c(u;i)$ is the neighborhood of the k most similar items that the user has already rated or commented and r'_{uj} is the result of the mapping function that accounts for both explicit ratings and those inferred from comments, defined as follows:

$$r'_{uj} = \begin{cases} 1, & \text{if } r_{uj} = 1 \\ c_{uj}, & \text{if } r_{uj} \neq 1 \end{cases} \tag{5}$$

c_{uj} is a mapping function which maps the polarity level of a comment C_j of user u to a rating understandable by the SANN model. We will compare three such mapping functions, formally defined in Table 2: two of them, noted 'randSANN' and 'SANN', generate 3-way ratings (1, 0, -1) from the random (Rand) and respectively the rule-based (RB) sentiment classifiers, while the third one, noted 'polSANN', generates a polarity value between -1 and 1 by combining the polarity scores of the sentences that constitute a comment. For all the three functions, the *pos* class has a positive effect on $\hat{r}_{ui}$, while *neu* and *neg* classes have a negative effect. An optimal function (per user or global) could also be learned from the data, in future work.

Moreover, the additional training data from commented items are considered for the creation of the co-rating matrix ($N \times N$) used for the similarity s_{ij} in Eq. 2.

Mapping function	**Notation**
$c_{uj} = sign_{rand}(C_j)$	randSANN
$c_{uj} = sign_{RB}(C_j)$	SANN
$c_{uj} = 1 + z_j \cdot \sum_{s \in C_j}\left(pol_{RB}(s)/\lvert s \rvert\right)$	polSANN

Table 2: Three mapping functions for the SANN model. Notations: $sign(C_j)$ is the sentiment of comment C_j (1 for *pos*, 0 for *neu* and -1 for *neg*); $pol(s)$ returns the polarity of a sentence s; and z_j is a normalization factor over all comments C^u of a user u, defined as $z_j = 1/(1 + |C^u| \cdot |\{C \, \text{s.t.} \, C \in C^u \wedge sign(C) = sign(C_j)\}|)$.

4. EXPERIMENTAL SETUP

4.1 The TED Dataset

To evaluate our models on the one-class problem, we focus on multimedia recommendations on the TED dataset [8][3]. TED (www.ted.com) is a popular online repository of public talks and user-contributed material (favorites, comments) under a Creative Commons license. We crawled the TED dataset in September 2012 and gathered data from 74,760 users and 1,203 talks, with 134,533 favorites and 209,566 comments. According to our RB classifier, 63% are positive, 27% are negative and 10% are neutral comments.

For this study, we selected users with at least 4 favorites (5,657 users) and included only comments that appear in the first level of the commenting area, i.e. excluding all the replies to other comments, because the target of their polarity judgment is uncertain (comments on comments rather than on the talk). The resulting data set has 129,633 ratings (favorites) and 20,792 comments (see Table 4).

[3] https://www.idiap.ch/dataset/ted

	k parameter (1 to 50)			λ parameter (1 to 100)		
Methods	**MAP@50**	**MAR@50**	**MAF@50**	**MAP@50**	**MAR@50**	**MAF@50**
TopPopular	3.46	12.69	5.44	3.46	12.69	5.44
normNN(PC)	4.06	13.70	6.27	4.13	14.00	6.37
NN(COS)	4.70	16.30	7.29	4.67	16.31	7.27
NN(PC)	4.75	16.47	7.37	4.76	16.58	7.40
SANN(COS)	5.07	17.90	7.91	6.12	20.65	9.45
SANN(PC)	**5.27**	**18.68**	**8.22**	**6.27**	**20.99**	**9.66**
Improvement	+10.9%	+13.4%	+11.5%	+31.8%	+26.5%	+30.6%

Table 3: Performance of various methods using 5-fold cross validation. The last row displays the improvement of the SANN model over the best NN model, which is always significant at the $p < 0.01$ level.

4.2 Evaluation Protocol

For evaluation, we split the dataset into a training set (80%) and a testing set (20%). More precisely, for each user we keep 80% of their positive ratings ('1' values) for training and hold out 20% for testing. Furthermore, we divide the test set in two subsets based on comment sparsity: a sparse and a dense one (see Table 4). For the sparse set, we keep all users with at least 12 ratings and for the dense one, all users with at least 12 ratings and one comment. We optimize the parameters on the training set using 5-fold cross-validation. When training, we use all users to obtain good approximations of item similarities (s_{ij} in Eq. 2).

Set	**Favorites**	**Comments**	**Users**
Training	108,256	20,792	5,657
Sparse held-out	17,227	8,728	2,409
Dense held-out	9,303	8,728	1,181

Table 4: Statistics of the TED dataset [8].

5. EXPERIMENTAL RESULTS

We evaluate the SANN model, comparing it with several baselines, and show the following: (i) the use of sentiment analysis of comments as additional training data improves performance over competitive baselines; (ii) the RB sentiment classifier is the reason for the improvement as compared to a random classifier; and (iii) performance increases with the number of comments.

We present first the results of parameter selection using cross-validation (5.1), and then we evaluate the best configurations on the two held-out sets (5.2).

5.1 Parameter Selection

Both NN models and SANN models rely on two parameters: the first one is the k parameter (the neighborhood size) and the second one is the λ parameter (the shrinking factor). For the selection of k, we fixed $\lambda = 60$ for all models. For the selection of λ, we then fixed k to the optimal values that were obtained for each model. Moreover, two other options can be selected, namely the similarity function (Pearson's correlation (PC) or cosine similarity (COS)) and the use of normalization in Eq. 1 (noted as 'norm').

In Table 3, we present the results of 5-fold cross-validation on the training set for six different models, namely: TopPopular baseline which provides fixed recommendations based on the popularity of items, NN(COS), normNN(PC), NN(PC), SANN(COS) and SANN(PC). We also experimented with other normalized NN models, but as their results were inferior, we do not discuss them here.

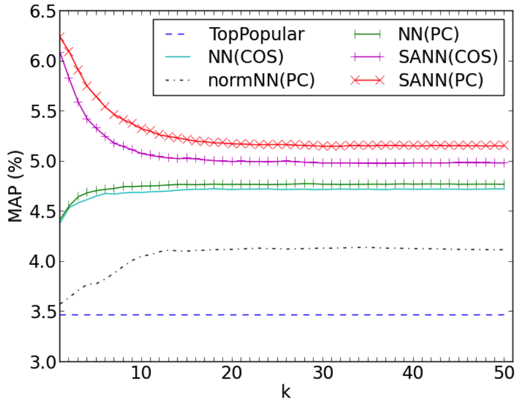

Figure 1: The effect of the neighborhood size k on MAP values at 50, using 5-fold cross-validation.

Figure 1 displays the effect of the neighborhood size k on the performance of the models. All models perform considerably better than the TopPopular, as expected from CF models. The best performing model over all k values is the SANN(PC) model (with significance at the $p < 0.01$ level). Both NN and SANN models stabilize their performance as k increases, which conforms to the theory of stability for k-nearest neighbor algorithms [1]. The decrease in performance of SANN as k increases is due to the inclusion of noisy ratings in the training data (along with ground-truth ones), while the performance of NN increases with k as no noise is included in this case. For the evaluation on the held-out sets (next section), the following values were selected based on cross-validation: $k = 1, \lambda = 7$ for the SANN(PC) model and $k = 28, \lambda = 19$ for the NN(PC) model.

5.2 Evaluation of the SANN Model

We compare the performance of the SANN model with the best performing baseline NN model (NN(PC)) and with the TopPopular baseline on the two held-out sets. We consider the two variants of the SANN model, namely randSANN(PC) and polSANN(PC) (also with $k = 1$ and $\lambda = 7$) to demonstrate the value of aggregated polarities.

Figure 2 displays performance on the dense held-out test set. The SANN model performs significantly better than the baseline models on unseen data. The ordering of the methods based on performance is the same as in the cross-validation experiments. The randSANN(PC) model performs slightly better than the NN(PC) model (the difference is not significant) but considerably worse than SANN(PC) and polSANN(PC), demonstrating the utility of sentiment analysis over comments.

Table 5 shows the results on both held-out test sets with MAP, MAR and MAF scores. The polSANN(PC) model outperforms significantly the other models, showing a relative improvement of 26.8% on mean average f-metric (MAF) on the dense held-out set and 11.4% on the sparse held-out set. When averaged over MAP values from 1 to 50, the improvement is significant at the $p < 0.01$ level (t-test).

The final experiment concerns the learning ability of the

	Dense held-out test set			Sparse held-out test set		
Methods	**MAP@50**	**MAR@50**	**MAF@50**	**MAP@50**	**MAR@50**	**MAF@50**
TopPopular	3.85	13.52	5.99	3.61	13.48	5.70
NN(PC)	5.67	18.07	8.63	5.23	18.06	8.11
randSANN(PC)	5.88	17.79	8.84	5.22	17.56	8.05
SANN(PC)	6.90	20.72	10.35	5.69	18.85	8.75
polSANN(PC)	**7.29**	**22.01**	**10.95**	**5.89**	**19.48**	**9.04**
Improvement	+28.5%	+21.8%	+26.8%	+12.6%	+7.8%	11.4%

Table 5: Performance of various methods on the two held-out test sets. The last row displays the improvement of the polSANN model over the best NN model.

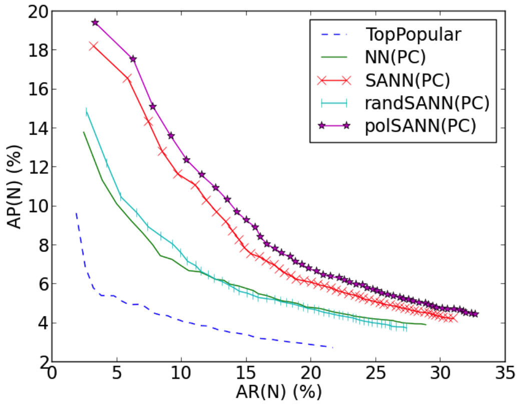

Figure 2: Method comparison in terms of average precision (AP) and recall (AR) (1 to 50).

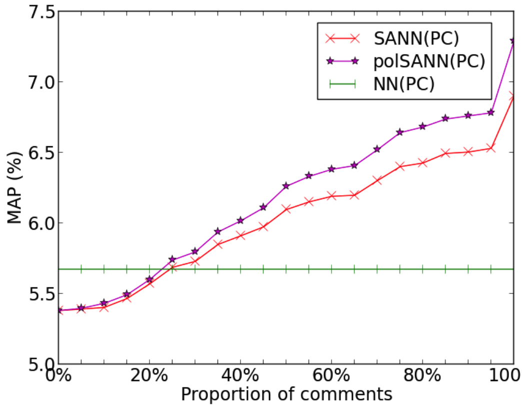

Figure 3: SANN models' performance (MAP at 50) when varying the number of comments for training.

algorithm in relation to the proportion of comments used for training. Figure 3 displays the performance of two SANN models on the dense held-out set, when varying the proportion of training comments for each user. The increase in the proportion of comments leads to a clear increase in performance. When fewer than 20% of the comments are taken into account, the models perform similarly to the NN(PC) model for neighborhood size $k = 1$ (see also Fig. 1).

6. CONCLUSION AND FUTURE WORK

The proposed SANN models were shown experimentally to be able to overcome some difficulties of the one-class collaborative filtering task and to outperform significantly several competitive baselines. Moreover, they demonstrated robustness to noise and exhibited appropriate learning abilities with respect to the amount of available user-generated text. The results of this study suggest several directions for future work: (i) extending the results to other datasets; (ii) learning the optimal mapping function from sentiment analysis scores to ratings; and (iii) generalizing the proposed approach to more advanced recommendation models.

7. ACKNOWLEDGMENTS

The work described in this article was supported by the European Union through the inEvent project FP7-ICT n. 287872 (see `http://www.inevent-project.eu`).

8. REFERENCES

[1] O. Bousquet and A. Elisseeff. Stability and generalization. *J. of Machine Learning Research*, 2002.

[2] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-N recommendation tasks. In *4th Int. Conf. on Recommender Systems*, Barcelona, Spain, 2010.

[3] G. Ganu, N. Elhadad, and A. Marian. Beyond the stars: Improving rating predictions using review text content. In *12th Int. Workshop on the Web and Databases*, Rhode Island, USA, 2009.

[4] C. Leung, S. Chan, F.-L. Chung, and G. Ngai. A probabilistic rating inference framework for mining user preferences from reviews. *World Wide Web*, 2011.

[5] A. Messenger and J. Whittle. Recommendations based on user-generated comments in social media. In *3rd Int. Conf. on Social Computing*, Boston, USA, 2011.

[6] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *8th Int. Conf. on Data Mining*, Pisa, Italy, 2008.

[7] N. Pappas, G. Katsimpras, and E. Stamatatos. Distinguishing the popularity between topics: A system for up-to-date opinion retrieval and mining in the Web. In *14th Int. Conf. on Intelligent Text Proc. and Computational Linguistics*, Samos, Greece, 2013.

[8] N. Pappas and A. Popescu-Belis. Combining content with user preferences for TED lecture recommendation. In *11th Int. Workshop on Content Based Multimedia Indexing*, Veszprém, Hungary, 2013.

[9] J. Wang, Q. Li, Y. P. Chen, and Z. Lin. Recommendation in Internet forums and blogs. In *48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 2010.

[10] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Conf. on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, Canada, 2005.

[11] W. Zhang, G. Ding, L. Chen, and C. Li. Augmenting Chinese online video recommendations by using virtual ratings predicted by review sentiment classification. In *Int. Conf. on Data Mining Workshops*, Washington, USA, 2010.