A CLUSTERING APPROACH FOR PRODUCT RECOMMENDATION SYSTEM IN E-COMMERCE USING SENTIMENT AND SIMILARITY OPINION MINING

Benarji Tharini[1], Dr.Vishnu Vardhan Bulusu[2]
[1] Research Scholar Rayalaseema University, Kurnool, AP, India.
tarini_benarji@yahoo.com
[2] Professor in Department of CSE, Manthany JNTUH, TS, India.
mailvishnu@yahoo.com

Abstract.

In this paper we describe a novel approach to case-based product recommendation. It is novel because it does not leverage the usual static, feature-based, purely similarity-driven approaches of traditional case-based recommends. Instead we harness experiential cases, which are automatically mined from user generated Telugu Language reviews, and we use these as the basis for a form of recommendation that emphasizes similarity and sentiment. We test our approach in a realistic product recommendation setting by using live-product data and user Telugu Language reviews. We present a recommendation ranking strategy that combines similarity and sentiment to suggest products that are similar however superior to a query product according to the opinion of reviewers, and we demonstrate the practical benefits of this approach across a variety of E-Commerce product domains.

Keywords: User-Generated TELUGU reviews, Opinion mining, NLP, Sentiment-based product recommendation.

1 Introduction

On-line recommender systems are a new source of recommendations. Such systems are becoming more commonplace, especially on the Internet. They can bolster us as we approach our on-line business, whether it be browsing our favorite on-line book shop or researching next year's vacation. Recommender systems combine ideas from information retrieval and filtering, user modeling, machine learning, and human– computer interaction. Case-based reasoning has played a key role in the development of an important class of recommender system known as content-based or case-based recommenders.

Recommendation services have for quite some time been an important feature of e-commerce platforms, making automated product suggestions that match the learned preferences of users. Ideas from case-based reasoning (CBR) can be readily found in many of these services. Which rely on the similarity between product queries and a database of product cases (the case base)? However, the relationship between CBR which emphasizes the reuse of experiences and many of these 'case-based' recommenders can be tenuous. There are two main classes of recommender system: those that employ collaborative approaches and those that employ case-based approaches. Collaborative approaches exploit user histories, usually in the form of ratings-based profiles. Recommendations come from the profiles of the active user's recommendation partners. The partners are users whose ratings correlate closely with the active user's ratings. A collaborative recommender will recommend items that are not already in the active user's profile but rather which her partners have rated very.

Collaborative recommender systems require user ratings for the items that are to be recommended. They don't require item descriptions, and this is what sets them apart from their content or case-based cousins. Item descriptions (whether they be text-based or attribute-value based) are vital in case-based recommenders, which generate a set of recommendations for a target user by retrieving items whose descriptions best match the user's query. A case-based reasoning (CBR) system will have a case base of cases (i.e. previously solved problems and their answers). New problems are solved by transferring and adapting arrangements that were used for similar problems in the past. CBR is a multi-step reasoning strategy, the details of which are covered admirably elsewhere (Aamodt and Plaza, 1994). For our purposes, we feature one of the essential early steps: retrieval. In the retrieval step, the system receives a problem specification, searches through the case base, scores each case for similarity to the new problem specification, and selects the highest-scoring case(s), which are the subject of subsequent steps, for example, adaptation.

There are evident parallels between the CBR retrieval step and the way a recommender system should treat a user query. From a CBR viewpoint, the query serves as a problem specification, the item descriptions are cases, and similarity-based retrieval techniques select the best-matching items.

2 Review Literature:

Product recommender systems have, for quite a while, relied on two primary sources of recommendation knowledge, either user ratings (Sarwar et al. 2001; Shardanand and Maes 1995; Desrosiers and Karypis 2011; Resnick et al. 1994; Koren et al. 2009) or product descriptions (Pazzani and Billsus 2007a; Lops et al. 2011; Smyth 2007; Bridge et al. 2005). For example collaborative filtering approaches (Shardanand and Maes 1995; Resnick et al. 1994) rely on the former to identify a neighborhood of users who are similar to some tar-get user to act as a source of item recommendations; basically products are selected for recommendation based on their popularity and/or ratings amongst the similar users. Alter-natively, when product descriptions are available then content-based (Pazzani and Billsus 2007a; Lops et al. 2011) or case-based (Smyth 2007; Bridge et al. 2005) recommendation approaches can be used, selecting products for recommendation because they are similar to those that the target user has liked in the past. Each of these approaches have their own advantages and disadvantages and can often be used in concert (alleged crossover recommenders Burke 2002) for more effective recommendation.

3 Related Work

Recent research features how online product reviews have a significant influence on the purchasing behavior of users; see [1–3].To cope with growing re-view volume retailers and researchers have explored deferent ways to help users find great reviews and avoid malicious or biased reviews. This has led to an assemblage of research focused on classifying or predicting review helpfulness. For example [4– 7] have all explored deferent approaches for extracting features from user-generated reviews in order to manufacture classifiers to identify helpful versus unhelpful reviews as the basis for a number of review ranking and filtering strategies.

It is becoming increasingly important to weed out malicious or biased re-views, supposed review spam. Such reviews can be well written and so appear to be superficially helpful. However reviews of this nature often adopt a biased perspective that is designed to help or hinder sales of the target product [8]. For example, Li et al. describe an approach to spam detection that is enhanced by information about the identity of the spammer as part of a two-tier, co-learning approach [9]. O'Callaghan et al. use network analysis techniques to identify recurring spam in user generated comments associated with YouTube videos by identifying discriminating comment themes that are indicative of spam bots [10].

In this work we are also interested in mining useful information from reviews and employ related feature extraction and opinion mining techniques to the above. However, our aim is to use this information to construct novel product case descriptions that can be used for recommendation rather than review filtering or classification. As such our work can be framed in the context of past approaches for case-based product recommendation including conversational recommender’s critiquing-based techniques [12], for example. For the most part, such past approaches are unified by their use of static case descriptions based around technical features. It isn't the type of case representation that is situated in any experiential setting. In contrast the cases that we produce from reviews are experiential: they are formed from the product features that users examine in their reviews and these features are linked to the opinions of these users. Past approaches also rely (usually exclusively) on query-case similarity as the primary recommendation ranking metric. In this work, while acknowledging that query similarity is an important way to anchor recommendations, we argue the importance of looking for cases that also dicer from the query case, at least in terms of the opinions of users at the feature level; see also [13]. We recommend cases that are similar to the query however preferred by end users.

4 **Mining product experiences**

The central aim of this work is to implement a practical technique for turning user-generated product TELUGU reviews into rich, feature-based, experiential product cases. The features of these cases relate to subjects that are discussed by reviewers and their aggregate opinions. Our intuition is that such features may provide access to a greatly expanded set of product features that would be unlikely to appear in classical catalog descriptions. Moreover, the avail-ability of user opinions for these features provides access to a rich source of experiential information that is obvious by its absence from other recommendation approaches.

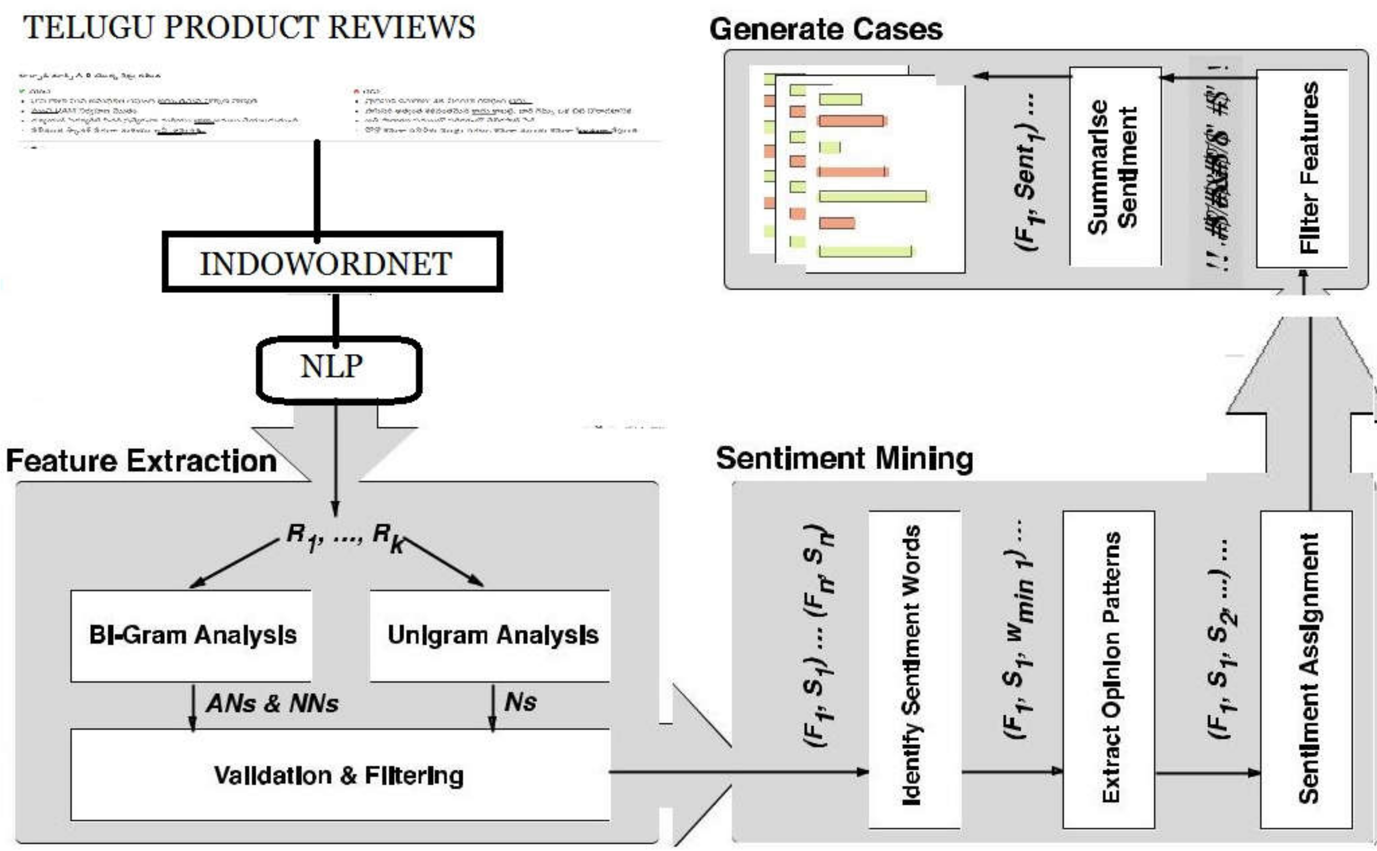

Fig. 1 Extracting experiential product cases from user-generated reviews

IndoWordNet

IndoWordNet is a linked lexical knowledge base of word nets of 18 scheduled languages of India, namely Assamese, Bangla, Bodo, Gujarati, TELUGU, Kannada, Kashmiri, Konkani, Malayalam, Meitei (Manipuri), Marathi, Nepali, Odia, Punjabi, Sanskrit, Tamil, Telugu and Urdu. Such project indeed took off in 2000 with TELUGU WordNet being created by the Natural Language Processing group at the Center for Indian Language Technology (CFILT) in the Computer Science and Engineering Department at IIT Bombay. [5] It was made publicly available in 2006 under GNU license. The TELUGU WordNet was created with support from the TDIL project of Ministry of Communication and Information Technology, India and also partially from Ministry of Human Resources Development, India. The word nets follow the principles of minimality, coverage and replace ability for the synsets. That means, there should be at least a 'core' set of lexemes in the synsets that uniquely give the concept represented by the synsets (minimality), e.g., {house, family}standing for the concept of 'family' ("she is from a noble house"). Then the synsets shouldcover ALL the words representing the concept in the language (coverage), e.g., the word 'ménage' will have to appear in the 'family' synsets, albeit, towards the end of the synsets,since its usage is rare. Finally, the words towards the beginning of the synsets should beable to replace one another in reasonable amount of corpora (replace ability), e.g., 'house' and 'family' can replace each other in the sentence "she is from a noble house". IndoWordNet is highly similar to EuroWordNet. However, the pivot language is TELUGU which, of course, is linked to the English WordNet. Also typical Indian language phenomena like complex predicates and causative verbs are captured in IndoWordNet. IndoWordNet is publicly brows able. The Indian language word net building efforts forming the subcomponents of IndoWordNet project are: North East WordNet project,Dravidian WordNet Project and Indradhanush project all of which are funded by the TDIL project .

Word nets of other languages of India then followed suit. The large nationwide project of building Indian language word nets was called the IndoWordNet project. IndoWordNet[1] is a linked lexical knowledge base of word nets of 18 scheduled languages of India, viz., Assamese, Bangla, Bodo, Gujarati, TELUGU, Kannada, Kashmiri, Konkani, Malayalam, Meitei, Marathi, Nepali, Oriya, Punjabi, Sanskrit, Tamil, Telugu and Urdu. The word nets are getting created by using expansion approach from the TELUGU WordNet. The TELUGU WordNet was created from first principles (mentioned below) and was the first wordnet for an Indian language. The method adopted was same as the Princeton WordNet for English. Polish WordNet is being mapped to Princeton WordNet based on the strategy followed by IndoWordNet.[6]

NLP

Natural language processing (NLP) is an area of computer science and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural language generation.

Our 3-step opinion mining approach is summarized in Fig. 1 and its different component parts are based on crafted by others in the opinion-mining literature; see for example (Hu and Liu 2004a; Justeson and Katz 1995; Hu and Liu 2004b; Moghaddam and Ester 2010). (1) For a given product domain (e.g. digital cameras, printers, etc.) we use shallow NLP techniques to extract a set of candidate features from the reviews of all products in that domain; then, each particular product P in the domain is represented by a subset of these features which appear in Reviews(P) = {R1, R2, . . . , Rk }, the reviews of P . (2) For each feature, Fi , we tally how frequently it is associated with positive, negative, or neutral sentiment based on the opinions expressed in the reviews of P . (3) These features and sentiment scores are aggregated at the product level to generate a case of features and overall sentiment scores.

Extracting features

We consider two basic types of features — bi-gram features and single-thing features — and use a combination of shallow NLP and statistical methods to mine them (Hu and Liu 2004a; Justeson and Katz 1995). For the former we search for bi-grams in reviews which conform to one of two basic part-of-speech co-location patterns: (1) an adjective followed by a thing (A) (e.g. wide angle); or (2) a thing followed by a thing (N) (e.g. video mode). These candidate features are filtered to avoid including A 's that are actually opinionated single-thing features; e.g. great flash is really a single-thing feature, flash. To do this we exclude bi-grams whose adjective is a sentiment word (e.g. excellent, terrible etc.) in the sentiment lexicon which we use in this work (Hu and Liu 2004a)1.

For single-thing features we also extract a candidate set, this time of things, from the reviews however we validate them by eliminating things that are rarely associated with sentiment words as per (Hu and Liu 2004b). The reason is that such things are unlikely to refer to product features (examples of such things include month, friends and day etc.). We calculate how frequently each feature co-happens with a sentiment word in the same sentence, and retain a single-thing just if its frequency is greater than some fixed threshold (in this case 30 %).

Generating experiential product cases

For each product *P* we now have a set of features $F(P) = \{F_1, \ldots, F_m\}$ extracted from *Reviews(P)*, and how frequently each feature F_i is associated with *positive, negative*, or *neutral* sentiment in the particular reviews in *Reviews(P)* that discuss F_i . For the purpose of this work we only include features in a product case if they are mentioned in more than 10 % of the reviews for that product. For these features we calculate an overall sentiment score as shown in (1) and their popularity as per (2). Then each product case, *Case(P)*, can be represented as shown in (3). Note, $Pos(F_i, P)$, $Neg(F_i, P)$, and $Neut(F_i, P)$ denote the number of times that feature F_i has positive, negative and neutral sentiment in the reviews for product *P* , respectively.

$$Sent(F_i, P) = \frac{Pos(F_i, P) - Neg(F_i, P)}{Pos(F_i, P) + Neg(F_i, P) + Neut(F_i, P)} \qquad (1)$$

$$Pop(F_i, P) = \frac{|\{R_k \in Reviews(P) : F_i \in R_k\}|}{|Reviews(P)|} \qquad (2)$$

$$Case(P) = \{[F_i, Sent(F_i, P), Pop(F_i, P)] : F_i \in F(P)\} \qquad (3)$$

5 Recommending products

Given the feature-based product representations above it is most natural to consider a content-based/case-based approach to recommendation (Pazzani and Billsus 2007a; Smyth 2007): to retrieve and rank recommendations based on their feature similarity to a query product. We will describe one such technique in what takes after. However, the availability of feature sentiment hints at an alternative approach to recommendation in which new products can be recommended because they offer improvements over certain features of the query product. We will also describe simply such an alternative and a half and half technique that allows for the flexible combination of similarity and sentiment.

5.1 Similarity-based recommendation

In our content-based recommendation strategy, each product case is represented as a vec-tor of features and corresponding popularity scores as per (2). As such, the *value* of a feature represents its frequency in reviews as a proxy for its importance. Then we use the cosine metric to compute the similarity between the query product, *Q*, and candidate recommendation, *C* as per (4).

$$Sim(Q,C) = \frac{\sum_{F_i \epsilon F(Q) \cup F(C)} Pop(F_i,Q) \times Pop(F_i,C)}{\sqrt{\sum_{F_i \epsilon F(Q)} Pop(F_i,Q)^2} \times \sqrt{\sum_{F_i \epsilon F(C)} Pop(F_i,C)^2}} \quad (4)$$

Clearly this is a very simple content-based recommendation technique, but it is in-line with many conventional approaches (Sarwar et al. 2001; Pazzani and Billsus 2007b), and serves as a useful baseline to evaluate the more sophisticated methods described below. As an aside we could have also considered a variation on the above where feature values were sentiment rather than popularity scores and, indeed, we have previously considered this in related work (Dong et al. 2013).

5.2 Sentiment-enhanced recommendation

The availability of feature sentiment suggests a very different approach to recommendation. Rather than looking for products that are *similar* to a query product, either in terms of feature popularity, as above, or feature sentiment why not look for products that offer *better* sentiment scores than the query product?

SENTIMENT PREDICTION

Sentiment prediction has been a great area of research in the recent times and is a challenging task especially in morphologically rich languages. The task requires us to classify a given sentence either as "Positive" or "Negative". In order to do this, we went ahead and tried out multiple deep learning based methods, however, we got the best results with a word-level multi-layer Convolution Neural Network(CNN) which we used as our final model.

Sentiment-enhanced recommendation

The availability of feature sentiment suggests a very different approach to recommendation. Rather than looking for products that are similar to a query product, either in terms of feature popularity, as above, or feature sentiment for what reason not search for products that offer better sentiment scores than the query product?

For example, consider a user who is considering a particular mobile. One of the features extracted from the mobile 's reviews is "processor speed " and let us assume that it has a popularity score of 0.6; indicating that about 60 % of the reviews refer to this feature. Let us also assume an intermediate sentiment score of 0.75; indicating a solid positive sentiment. When selecting a new mobile for recommendation would it is a good idea for us to, all other things being equal, search for other mobile that have processor speed mentioned in a similar extent of reviews or that have a similar overall sentiment associated with processor speed? Or then again would it be advisable for us to seek to find cameras that offer an improved sentiment score for this feature? Surely the latter makes more sense in the context of likely consumer preferences?

The starting point for this is the better work appeared as (5), which calculates a straight-forward better score for feature Fi between query product Q and recommendation candidate

C. A better score less than 0 means that the query product Q has a better sentiment score for Fi than C whereas a positive score means that C has the better sentiment score for Fi compared to Q.

$$better(F_i,Q,C) = \frac{Sent(F_i,C) - Sent(F_i,Q)}{2} \quad (5)$$

We can then calculate an overall *better score* at the product level by aggregating the individual better scores for the product features. There are two obvious ways to do this. First, in (6) we compute the average better scores across the features that are shared between

\emdash and C. However, this approach ignores those (potentially many) features that may be unique to Q or C, so called *residual features*. For instance, in Fig. 2 we see an example of the approach for two candidate recommendations, C_1 and C_2, with respect to a query case Q. In terms of their shared features, C_1 offers a better sentiment improvement than C_2 and so would be selected ahead of C_2 on sentiment grounds during recommendation.

$$B1(Q, C) = \frac{\sum_{F_i \in F(Q) \cap F(C)} better(F_i, Q, C)}{|F(Q) \cap F(C)|} \qquad (6)$$

A second alternative, to deal with these residual features, is to assign non-shared features a neutral sentiment score of 0 and then compute an average better score across the union of features in Q and C as in (7).

$$B2(Q, C) = \frac{\sum_{F_i \in F(Q) \cup F(C)} better(F_i, Q, C)}{|F(Q) \cup F(C)|} \qquad (7)$$

In Fig. 3 we return to our example of C_1 and C_2 above, but this time their fortunes are reversed based on a comparison of all (shared plus residual) features. This time C_2 wins out over C_1 with respect to Q.

5.3 Clustering similarity and sentiment

The above provides two alternatives for a sentiment-based approach to recommendation, which ranks product cases in decreasing order of their better score (either B1 or B2). They prioritize recommendations that enjoy more positive reviews across a range of features relative to the query product. However, these recommendations may not necessarily be very similar to the query product. What is required is a way to combine similarity and sentiment during recommendation with the goal that we can prioritize products that are similar to the query product while also being more positively reviewed.

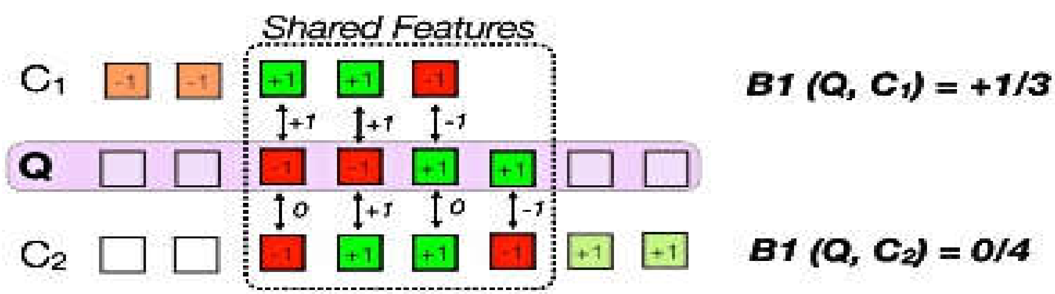

Fig. 2 Case C_1 offers better sentiment improvement than C_2 when compared to the query product Q based on shared features

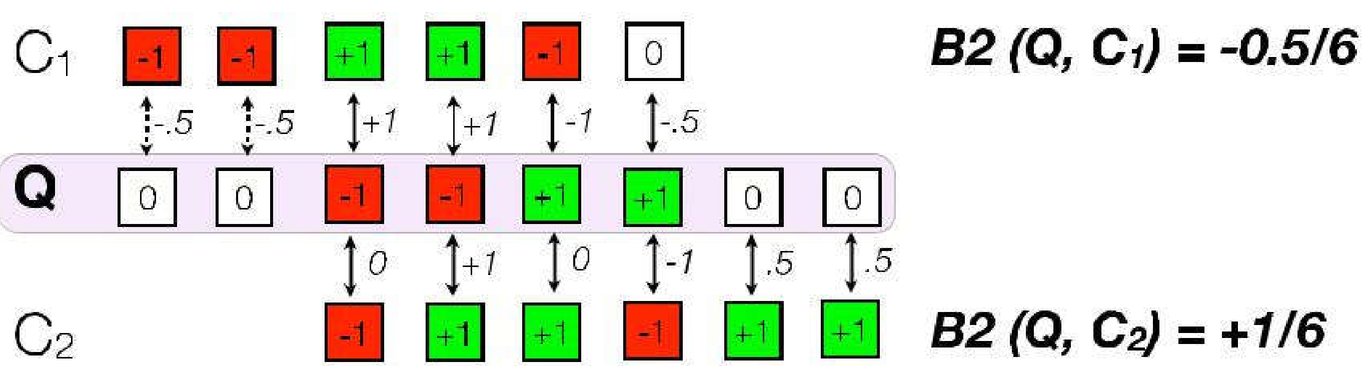

Fig. 3 Case C_2 offers better sentiment improvement than C_1 when compared to the query product Q based on shared and residual features

Perhaps the simplest way to combine similarity and sentiment approaches is to use a hybrid scoring metric such as that shown in (8); in this instance *Sent (Q, C)* can be implemented as either *B*1 or *B*2 above[2]. Thus we compute an overall score for a candidate recommendation *C* based on a combination of *C*'s similarity and sentiment scores with respect to *Q*. In what follows we will use this as our basic recommendation ranking approach, implementing versions that use *B*1 and *B*2 and varying the parameter *w* to control the relative influence of feature similarity and sentiment during recommendation.

$$Score(Q, C) = (1 - w) \times Sim(Q, C) + w \times \left(\frac{Sent(Q, C) + 1}{2}\right) \quad (8)$$

Combining related features

The approach to opinion mining that we have described so far is susceptible to a proliferation of mined features because it is insensitive to the many and varied ways that people will inevitably refer to the same product features.

Feature clustering

One way to address this is to attempt to cluster similar features together on the basis of similarities in the way that they are referred to in user generated reviews. For example, we can associate each extracted feature with a description vector that is made up of the set of terms extracted from the sentences that refer to this feature; see (10) where $Sens(F_i)$ denotes the set of sentences in which feature F_i occurs and $Terms(S_k)$ denotes the set of terms contained in a sentence S_k [4]. Thus each feature F_i is associated with a set of terms and each feature can be associated with a normalized term frequency weight, w_j (Manning et al. 2008). In this way, each feature can be compared based on the similarity of their description vectors.

$$Desc(F_i) = \bigcup_{\forall S_k \in Sens(F_i)} \{t_j : t_j \in Terms(S_k), w_j\} \quad (10)$$

Next, we can apply standard clustering techniques to these description vectors. In this experiment we use CLUTO[5] and select a standard partitional clustering algorithm. In fact we consider two experimental conditions. In the *standard* condition, the objective is to take feature synonyms into account and to cluster related features together; for example, *picture* and *shots* are synonyms of feature *image*. By experiment, setting the target number of clusters to be 35 % of the total number of features extracted for each domain (i.e. such that each cluster contains approximately three related features) provided good performance in this regard. Thus, this approach allows us to consider the performance of clustering with minimal fine-tuning and a particular objective (capturing feature synonyms) in mind.

The second clustering condition, *optimized*, considers a number of clustering algorithms and cluster criterion functions available from the CLUTO toolkit, and the best performing combination for each product domain over a range of partitions with different numbers of clusters is selected. This affords us with an opportunity to evaluate performance when a greater degree of fine-tuning has been carried out in order to understand the potential of this particular variation.

Generating cases from clustered features

Using this clustering approach we can modify the case generation step of our approach. Each case is now made up of a set of clusters and each cluster is comprised of a set of features. In effect, each cluster corresponds to a type of high-level feature, such that the features it contains are related in some way. For example, we might expect to find a cluster that corresponds to the "picture quality" of a digital camera and for it to contain features such as "image resolution", "picture clarity", "night images" etc. Now, for a given cluster C_j we compute its sentiment and popularity scores in a manner similar to the way in which we compute the individual feature scores in (1) and (2), except that now each cluster contains a set of features. Thus the sentiment and popularity scores are each aggregated across all of these in-cluster features, $F_1, \ldots, F_m$, as per (11) and (12):

$$Sent(C_j, P) = \frac{\sum_{F_i \in C_j} Pos(F_i, P) - \sum_{F_i \in C_j} Neg(F_i, P)}{\sum_{F_i \in C_j} Pos(F_i, P) + \sum_{F_i \in C_j} Neg(F_i, P) + \sum_{F_i \in C_j} Neut(F_i, P)} \quad (11)$$

$$Pop(C_j, P) = \frac{|\{R_k \in Reviews(P) : F_1 \in R_k \vee F_2 \in R_k \vee \cdots \vee F_m \in R_k\}|}{|Reviews(P)|} \quad (12)$$

where $Pos(F_i, P)$, $Neg(F_i, P)$ and $Neut(F_i, P)$ denote the number of times feature $F_i \in C_j$ has positive, negative and neutral sentiment in the reviews for product P $Reviews(P)$, respectively.

Conclusions

Intuitively user-generated product reviews appear to provide a rich source of recommendation raw material however to the best of our knowledge these data sources have not been used as the basis for recommendation, at least in a direct way. In this article we have described an approach to mining product descriptions from raw review texts and we have indicated how this information can be used to drive a novel recommendation technique that combines aspects of product similarity and feature sentiment. In divert we have presented results from a comprehensive evaluation product domains containing more than 1,000 products and 90,000 reviews. These results point to clear benefits in terms of recommendation quality, by combining similarity and sentiment information, compared to a suitable ground-truth .Importantly, these recommendations have been produced without the need for large-scale transaction/ratings data (cf. collaborative filtering approaches) or structured product knowledge or meta-data (cf. conventional content-based approaches). On this basis we can be somewhat confident that the approaches we have described in this work provide a useful new approach to case-based product recommendation.

References

1. Ruihai Dong , Barry Smyth, From More-Like-This to Better-Than-This: Hotel Recommendations from User Generated Reviews, Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, July 13-17, 2016, Halifax, Nova Scotia, Canada

2. Ruihai Dong , Barry Smyth, User-based opinion-based recommendation, Proceedings of the 26th International Joint Conference on Artificial Intelligence, August 19-25, 2017, Melbourne, Australia
Hyewon Lim , Hyoung-Joo Kim, Item recommendation using tag emotion in social cataloging services, Expert Systems with Applications: An International Journal, v.89 n.C, p.179-187, December 2017

3. M. Arun Manicka Raja , S. Swamynathan, Tweet Sentiment Analyzer: Sentiment Score Estimation Method for Assessing the Value of Opinions in Tweets, Proceedings of the International Conference on Advances in Information Communication Technology & Computing, p.1-6, August 12-13, 2016, Bikaner, India
4. Benarji Tharini1, Dr.Vishnu Vardhan Bulusu2" Development of a Micro Telugu Opinion WordNet and Aligning with TELOWN Ontology for Automatic Recognition of Opinion Words from Telugu Documents" Volume 7, Issue VI, JUNE/2018. INTERNATIONAL JOURNAL OF RESEARCHVolume 7, Issue VI, JUNE/2018. ISSN NO : 2236-6124
1. Archak, N., Ghose, A., & Ipeirotis, P.G. (2011). Deriving the pricing power of product features by mining consumer reviews. Management Science, 57(8), 1485–1509.
2. Baccianella, S., Esuli, A., & Sebastiani, F. (2009). Multi-facet rating of product reviews. In Advances in Information Retrieval, 31th European Conference on Information Retrieval Research (ECIR 2009) (pp. 461–472). Toulouse, France: Springer.
3. Bar-Haim, R., Dinur, E., Feldman, R., Fresko, M., & Goldstein, G. (2011). Identifying and following expert investors in stock microblogs. In Proceedings of the Conference on Empirical Methods in Natural Lan-guage Processing, EMNLP '11, pp 1310–1319. PA, USA: Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=2145432.2145569.
4. Boiy, E., & Moens, M.F. (2009). A machine learning approach to sentiment analysis in multilingual web texts. Information Retrieval, 12(5), 526–558.
5. Bridge, D., Goker,¨ M.H., McGinty, L., & Smyth, B. (2005). Case-based recommender systems. Knowledge Engineering Review, 20(03), 315–320.
6. Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User Model User-Adapted
7. International, 12(4), 331–370. doi:10.1023/A:1021240730564.
8. Burke, R., Hammond, K., & Yound, B. (1997). The findme approach to assisted browsing. IEEE Expert, 12(4), 32–40. doi:10.1109/64.608186.
9. Dasgupta, S., & Ng, V. (2009). Mine the easy, classify the hard: A semi-supervised approach to automatic sentiment classification. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, ACL '09, pp 701–709. PA, USA: Association for Computational Linguistics. http://dl.acm.org/citation. cfm?id=1690219.1690244.
10. De Francisci Morales, G., Gionis, A., & Lucchese, C. (2012). From chatter to headlines: Harnessing the real-time web for personalized news recommendation. In Proceedings of the fifth ACM Interna-tional Conference on Web Search and Data Mining, WSDM '12, pp. 153–162. NY, USA: ACM. doi:10.1145/2124295.2124315.
11. Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In Recommender Systems Handbook (pp. 107–144): Springer.
12. Ding, X., Liu, B., & Yu, P.S. (2008). A holistic lexicon-based approach to opinion mining. In Proceedings of the 1st ACM International Conference on Web Search and Data Mining (pp. 231–240): ACM.
13. Dong, R., O'Mahony, M.P., Schaal, M., McCarthy, K., & Smyth, B. (2013). Sentimental product recommen-dation. In Proceedings of the 7th ACM Conference on Recommender Systems, RecSys '13 (pp. 411–414). NY, USA: ACM. doi:10.1145/2507157.2507199.
14. Dong, R., O'Mahony, M.P., & Smyth, B. (2014). Further experiments in opinionated product recomen-dation. In Proceedings of the 22nd International Conference on Case-Based Reasoning, ICCBR '14 (pp. 110–124): Springer.

15. Dong, R., Schaal, M., O'Mahony, M.P., McCarthy, K., & Smyth, B. (2013). Opinionated product recom-mendation. In Proceedings of the 21st International Conference on Case-Based Reasoning, ICCBR '13 (pp. 44–58). Heidelberg: Springer.
16. Dong, R., Schaal, M., O'Mahony, M.P., & Smyth, B. (2013). Topic extraction from online reviews for clas-sification and recommendation. In Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI '13. Menlo Park, California: AAAI Press.
17. Dooms, S., De Pessemier, T., & Martens, L. (2013). Movietweetings: a movie rating dataset collected from twitter. In Workshop on Crowdsourcing and Human Computation for Recommender Systems, CrowdRec at RecSys, Vol. 13.
18. Feldman, R., Rosenfeld, B., Bar-Haim, R., & Fresko, M. (2011). The stock sonar sentiment analysis of stocks based on a hybrid approach. In Proceedings of the 23rd IAAI Conference.
19. Garcia Esparza, S., O'Mahony, M.P., & Smyth, B. (2010). On the real-time web as a source of recommen-dation knowledge. In Proceedings of the fourth ACM Conference on Recommender Systems, RecSys '10 (pp. 305–308). NY, USA: ACM. doi:10.1145/1864708.1864773.
20. Herlocker, J.L., Konstan, J.A., & Riedl, J. (2000). Explaining collaborative filtering recommendations. In Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, CSCW '00 (pp. 241–250). NY, USA: ACM. doi:10.1145/358916.358995.
21. Hsu, C.F., Khabiri, E., & Caverlee, J. (2009). Ranking comments on the social web. In Proceedings of the 2009 IEEE International Conference on Social Computing (SocialCom-09) (pp. 90–97). Vancouver, Canada.
22. Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04 (pp. 168–177). NY, USA: ACM. doi:10.1145/1014052.1014073.
23. Hu, M., & Liu, B. (2004). Mining opinion features in customer reviews. In Proceedings of the 19th National Conference on Artifical Intelligence, AAAI'04 (pp. 755–760): AAAI Press. http://dl.acm.org/citation. cfm?id=1597148.1597269.
24. Huang, J., Etzioni, O., Zettlemoyer, L., Clark, K., & Lee, C. (2012). Revminer: An extractive interface for navigating reviews on a smartphone. In Proceedings of the 25th Annual ACM Symposium on User Inter-face Software and Technology, UIST '12 (pp. 3–12). NY, USA: ACM. doi:10.1145/2380116.2380120.
25. Jiang, L., Yu, M., Zhou, M., Liu, X., & Zhao, T. (2011). Target-dependent twitter sentiment classification (pp. 151–160): ACL.
26. Justeson, J.S., & Katz, S.M. (1995). Technical terminology: Some linguistic properties and an algorithm for identification in text. National Language Engineering, 1(1), 9–27.