Opinionated Product Recommendation

Ruihai Dong, Markus Schaal, Michael P. O'Mahony, Kevin McCarthy, and Barry Smyth

CLARITY: Centre for Sensor Web Technologies
School of Computer Science and Informatics
University College Dublin, Ireland

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 recommenders. Instead we harness experiential cases, which are automatically mined from user generated reviews, and we use these as the basis for a form of recommendation that emphasises similarity and sentiment. We test our approach in a realistic product recommendation setting by using live-product data and user reviews.

1 Introduction

Recommendation services have long 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 — so-called content-based (or case-based) recommenders — which rely on the similarity between *product queries* and a database of *product cases* (the case base). However, the relationship between CBR — which emphasises the reuse of *experiences* — and many of these 'case-based' recommenders can be tenuous. For example, many case-based recommenders do borrow similarity assessment techniques from CBR, as a basis for query-product similarity, but the idea that product cases (which are typically static feature-based records) are experiential is at best a stretch. Does this matter? After all such approaches have met with considerable success and have proven to be useful in practical settings. But how might we harness *genuine* experiential knowledge as part of a case-based product recommender? This is the question that we address in this paper. We do this by describing and evaluating a novel approach to product recommendation that relies on product cases that are genuinely experiential in nature as well as a unique approach to retrieval that is based on the combination of feature similarity and user sentiment.

Consider the *Fujifilm X100* camera. At the time of writing the *product features* listed by Amazon cover technical details such as *resolution* (12.3 MP), *sensor-type* (APS-C), *aperture* (f2), and *price*($1,079.00). These are the type of features that one might expect to find in a conventional product recommender, facilitating the recommendation of other products that share similar values for these same features. The features are clearly few in number: this limits the scope

S.J. Delany and S. Ontañón (Eds.): ICCBR 2013, LNAI 7969, pp. 44–58, 2013.

of assessing inter-product similarity at recommendation time. Moreover, features are often technical in nature, so it can be difficult to judge the importance of feature similarities in any practical sense. Is a 13.3 MP camera more or less similar to the *X*100 than a 11.3 MP alternative? However, the *X100* has 149 reviews which encode valuable insights into a great many features of the *X*100, from its beautiful design to its quirky interface, and from its great picture quality to the limitations of its idiosyncratic auto-focus or the lack of optical stabilisation. Clearly these features capture far more detail than the handful of technical 'catalog' features. The reviews also encode the *opinions* of users and as such provide a subjective basis for comparison; all other things being equal, for example, the *"beautiful retro design"* of the *X*100 certainly beats another camera suffering from *"terrible design"*. A key idea of this work is that we can mine these opinion-rich features directly from user-generated reviews and use them as detailed experiential product cases to provide a basis for recommendation.

The key contributions of this work are three-fold. Firstly, we describe how product features can be automatically mined from the plentiful user-generated reviews on sites like Amazon.com and TripAdvisor etc. These features are aggregated at the product-level to produce *product cases*. Secondly, we explain how these product features can be associated with sentiment information to reflect the opinions of reviewers, whether positive, negative, or neutral. The resulting product cases are thus genuinely *experiential* in nature, in the sense that they are based wholly on the opinions and experiences of the users of these products. Thirdly, we describe a novel approach to *"more-like-this"* style recommendations that are based on a combination of similarity and sentiment, to prioritise products that are similar to, but better than, a given target (query) product.

2 Related Work

Recent research highlights how online product reviews have a significant influence on the purchasing behavior of users; see [1–3].To cope with growing review volume retailers and researchers have explored different ways to help users find high quality reviews and avoid malicious or biased reviews. This has led to a body of research focused on classifying or predicting review helpfulness. For example [4–7] have all explored different approaches for extracting features from user-generated reviews in order to build 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 reviews, so-called *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 *motifs* that are indicative of spambots [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 build 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 recommenders* [11] and *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 is not 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 discuss 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 differ 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 but *preferred* by end users.

3 Recommending Experiential Product Cases

A summary of our overall approach is presented in Figure 1. Briefly, a case for a product P is made up of a set of product features and their sentiment scores mined from $Reviews(P)$, the set of reviews written for product P. The sentiment of each feature is evaluated at the review-level first and then aggregated at the case-level as an overall sentiment score for that feature. At recommendation time suitable cases are retrieved and ranked based on their similarity and sentiment with respect to a given query case Q.

3.1 Extracting Review Features

When it comes to extracting features from reviews for a particular product category (for example, *Laptops*, *Tablets*), we consider two basic types of features — *bi-gram* features and *single-noun* features. We use a combination of shallow NLP and statistical methods, by combining ideas from Hu and Liu [14] and Justeson and Katz [15]. To produce a set of bi-gram features we look for bi-grams in the review cases which conform to one of two basic part-of-speech co-location patterns: (1) an adjective followed by a noun (AN) such as *wide angle*; and (2) a noun followed by a noun (NN) such as *video mode*. These are candidate features but need to be filtered to avoid including AN's that are actually opinionated single-noun features; for example, *great flash* is a single-noun feature (*flash*) and not a bi-gram feature. To do this we exclude bi-grams whose adjective is found to be a sentiment word (for example, *excellent, great, terrible, horrible* etc.) using Hu and Liu's sentiment lexicon [16].

To identify single-noun features we extract a candidate set of nouns from the reviews. Often these candidates will not make for good case features however; for

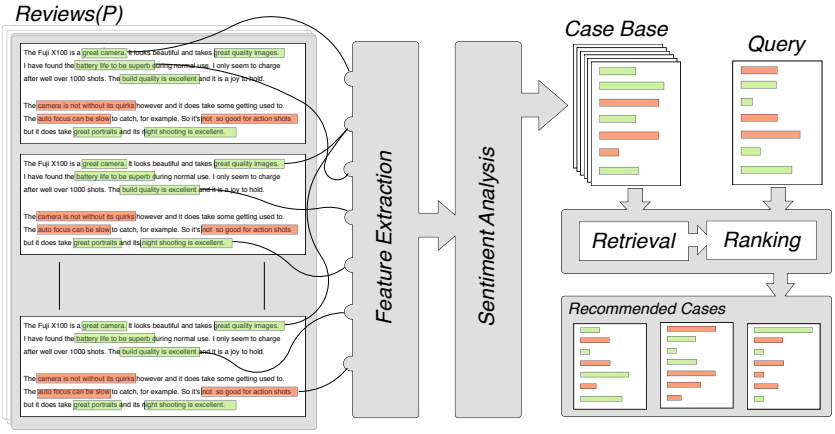


Fig. 1. An overview of how we mine user-generated reviews to create experiential product case bases for sentiment-based recommendation

example, they might include words like *family* or *day* or *vacation* which do not relate to product features. Hu and Liu [16] propose a solution to validate such features by eliminating those that are rarely associated with opinionated words. The intuition is that nouns that frequently co-occur with opinion laden words in reviews are likely to be relevant product features. We calculate how frequently each feature co-occurs with a sentiment word in the same sentence (again, as above, we use Hu and Liu's sentiment lexicon [16]), and retain a single-noun only if its frequency is greater than some threshold (in this case 70%).

This produces a set of bi-gram and single-noun features which we filter based on their frequency of occurrence, keeping only those features that occur in at least k of the s reviews; in this case, for bi-gram features we set $k_{bg} = s/20$ and for single noun features we set $k_{sn} = 10 \times k_{bg}$, where s is the total number of reviews for a category.

3.2 Evaluating Feature Sentiment

For each feature we evaluate its sentiment based on the sentence containing the feature. We use a modified version of the *opinion pattern mining* technique proposed by Moghaddam and Ester [17] for extracting opinions from unstructured product reviews. Once again we use Hu and Liu's sentiment lexicon as the basis for this analysis. For a given feature F_i and corresponding review sentence S_j from review R_k, we determine whether there are any sentiment words in S_j. If there are not then this feature is marked as *neutral*, from a sentiment perspective. If there are sentiment words then we identify the word w_{min} which has the minimum word-distance to F_i.

Next we determine the part-of-speech (POS) tags for w_{min}, F_i and any words that occur between w_{min} and F_i. The POS sequence corresponds to an *opinion pattern*. For example, in the case of the bi-gram feature *noise reduction* and the

review sentence, *"...this camera has great noise reduction..."* then w_{min} is the word *"great"* which corresponds to an opinion pattern of *JJ-FEATURE* as per Moghaddam and Ester [17]. After a complete pass of all features through all reviews we can compute the frequency of all opinion patterns that have been recorded. A pattern is deemed to be valid (from the perspective of our ability to assign sentiment) if it occurs more than the average number of times. For valid patterns we assign sentiment to F_i based on the sentiment of w_{min} and subject to whether S_j contains any negation terms within a 4-word-distance of w_{min}. If there are no such negation terms then the sentiment assigned to F_i in S_j is that of the sentiment word in the sentiment lexicon. Otherwise this sentiment is reversed. If an opinion pattern is deemed not to be valid (based on its frequency) then we assign a *neutral* sentiment to each of its occurrences within the review set.

3.3 Generating Experiential Cases

For each review R_i the above methods generate a set of valid features $F_1, ..., F_{m_i}$ and their associated sentiment scores *positive*, *negative*, or *neutral*. We can now construct experiential product cases in a straightforward fashion, as a set of product features paired with corresponding sentiment scores as per Equation 1.

$$Case(P) = \{(F_j, Sentiment(F_j, P)) : F_j \in Features(P)\} \tag{1}$$

The case features ($Features(P)$) for a product P are the union of the valid features extracted from its reviews. Each of these features may be present in a number of P's reviews and with different sentiment scores. To assign a sentiment score to a feature at the case-level we aggregate the individual review-based sentiment scores according to Equation 2, where $Pos(F_j, P)$ is the number of positive sentiment instances of F_j among the reviews of product P, and likewise for $Neg(F_j, P)$ and $Neutral(F_j, P)$. Thus, $Sentiment(F_j, P)$ will return a value between -1 (*negative* sentiment) and +1 (*positive* sentiment). For example, one of the features extracted for the $X100$ camera mentioned earlier is its *lens quality* which is invariably mentioned in a positive fashion across many reviews. As such its overall sentiment score is 0.72 (25 positive mentions, 5 neutral mentions, and only 2 negative mentions).

$$Sentiment(F_j, P) = \frac{Pos(F_j, P) - Neg(F_j, P)}{Pos(F_j, P) + Neg(F_j, P) + Neutral(F_j, P)} \tag{2}$$

3.4 From Case Retrieval to Sentiment-Enhanced Recommendation

Now that we have a case base of experiential cases we can describe our approach to recommendation. First it is worth stressing again that, unlike many more conventional approaches to product recommendation, these experiential cases do not have a fixed set of shared static features. Instead each case is represented by its own (possibly unique) set of features, mined from its own product reviews.

We must ensure some minimal set of shared features between cases to serve as the basis for comparison. First we define *k-comparability* as a boolean property of two cases P_u and P_v which is true if and only if P_u and P_v share at least k features. During retrieval we only consider cases that are at least *k-comparable* (have at least k features in common) with the target query case Q; see Equation 3, where CB denotes the case base of all product cases.

$$Retrieve_k(Q) = \{P \in CB : k - comparable(Q, P)\} \tag{3}$$

In a conventional product recommender system, we would likely rank these cases in decreasing similarity to the query case, for some suitable similarity metric (for example, Jaccard or Cosine similarity). However, we adopt a very different approach in this work. Remember that the values of our case features are sentiment scores; that is, overall judgements by real users about how good or bad a given feature is. It stands to reason that we would like to rank cases according to how much *better* their respective feature scores are compared to the query case. If the query case has a sentiment score of 0.5 for *lens quality* then we would surely prefer to rank another case with a score of 0.8 for *lens quality* ahead of a case with a *lens quality* of score 0.6, all other things being equal, and even though the latter case has a more similar *lens quality* sentiment score than the former case compared to the query. Thus, cases that have a *better* sentiment score across their shared features ($Features(Q) \cap Features(C)$) should be preferred.

$$better(F, Q, P) = \frac{Sentiment(F, P) - Sentiment(F, Q)}{2} \tag{4}$$

$$Better(Q, P) = \frac{\sum_{\forall F \in Features(Q) \cap Features(P)} better(F, Q, P)}{|Features(Q) \cap Features(P)|} \tag{5}$$

We compute a *better* score between the sentiment for a feature F in a query Q and a retrieved case P; see Equation 4. This returns a value from -1 (the sentiment for F in Q is better than in P) to +1 (the sentiment for F in P is better than in Q). Then we calculate a *Better* score at the case-level as the average better scores for the features shared between Q and P; see Equation 5.

Thus, for a given query case we first retrieve a set of *k-comparable* cases for a suitable value of k ($k = 15$ in this work) and then these cases are ranked in terms of the degree to which their sentiment scores are better over the shared features in the query case. We then return the top n ranked products as recommendations.

4 Evaluation

In this section we test how well this experience-based product recommendation works in practice. We do this by using a large corpus of more than 12,000 product reviews for about 1,000 consumer electronic products, ultimately comparing the performance of our sentiment-based recommendation to a more conventional recommendation approach using a reliable and objective ground-truth.

4.1 Data Sets and Setup

The review data for this experiment was extracted from Amazon.com during October 2012. We focused on 3 different product categories: *GPS Devices, Laptops, Tablets*. For the purpose of our experiments, we filtered for products with 10 or more reviews. Table 1 shows the number of products and reviews in the raw data, and the number of products with at least 10 reviews, per product category. Each of these 3 product categories is turned into a product case base using the approach described previously, and Table 1 also shows the mean and standard deviation of the number of features extracted per product type.

Table 1. Evaluation of product categories and case bases

Category	#Reviews	#Prod.	#Prod.(Filtered)	# Features Mean (Std. Dev.)
GPS Devices	12,115	192	119	24.32 (10.82)
Laptops	12,431	785	314	28.60 (15.21)
Tablets	17,936	291	166	26.15 (10.48)

4.2 Feature Sparsity and Case *k-Comparability*

The statistics above suggest that cases are being generated with a rich set of 20-30 features. This bodes well but it is of limited use unless these features are shared among the products in a case base. Small numbers of shared features greatly restrict our ability to compare cases during retrieval and lead to the type of *sparsity problem* that is common in collaborative filtering systems [18].

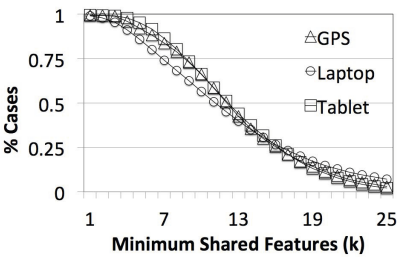


Fig. 2. The average percentage of cases with a minimum of k shared features with the query case

To explore this we can examine the average size of the *k-comparability* sets, as a percentage of case base size, for each product type across different levels of k; see Figure 2. We can see that for $k = 15$, in all three case bases (Laptop, GPS Devices, and Tablets) the average number of *k-comparable* cases is about 35%

of all cases (approximately 109 Laptops, 41 GPS Devices, or 58 Tablets). This is important for a few reasons. It helps to validate the type of features that we are extracting from reviews. The fact that there are so many cases sharing at least k features, even for relatively large values of k, means that we are extracting features that are frequently recurring in product reviews. It also means that there is no significant feature sparsity problem in any of the case bases examined. From a recommendation standpoint this means that we are able to compare product cases based on a rich set of shared features since it is now practical to set k to be between 10 and 20, for example, in these case bases and to still ensure large enough retrieval sets to form the basis for final recommendation.

4.3 Recommendation Quality as Sentiment Improvement

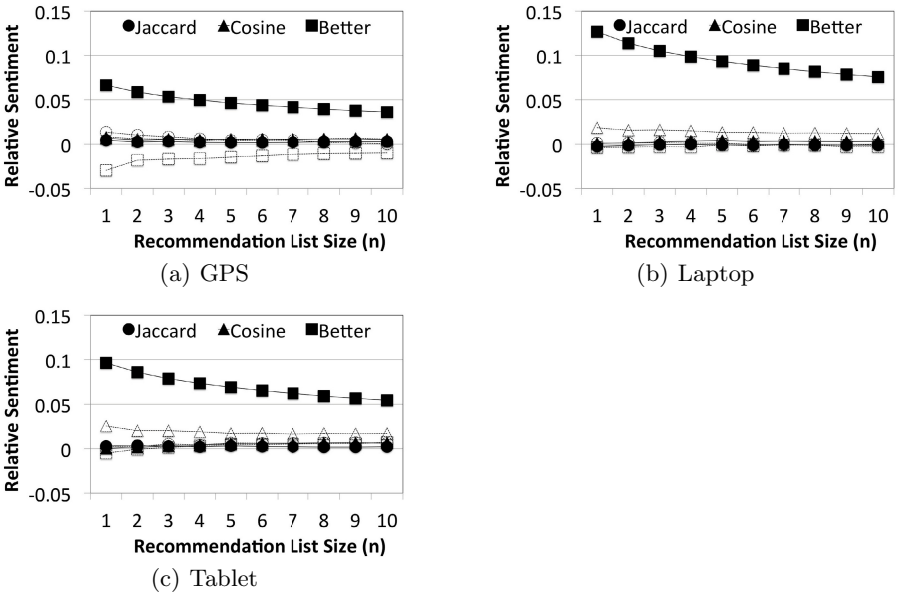
It is good that we are extracting many features from reviews and it is promising that these features tend to recur across many case base products. But are these features, and their associated sentiment scores, effective from a recommendation perspective? Do they facilitate the recommendation of useful products? To test this we consider a number of recommendation strategies as follows:

- *Jaccard* — cases are ranked based on a simple Jaccard metric ($|Features(Q) \cap Features(C)|/|Features(Q) \cup Features(C)|$) over case features; that is, sentiment information is not used and cases are preferred if they share a higher percentage of features with the query case.
- *Cosine* — cases are ranked based on a standard cosine similarity metric calculated from the sentiment scores of shared features.
- *Better* — cases are ranked based on the *Better* metric described previously, which prioritises cases that enjoy improved sentiment scores relative to the query. Again, *Better* scores are calculated from shared features only.

Using a standard *leave-one-out* methodology, each one of these strategies produces a different ranking of retrieved cases. To evaluate the quality of these rankings we consider two different types of ground-truth data: (1) relative sentiment and (2) independent product ratings.

Relative Sentiment. First, we can measure the relative improvement in the sentiment of the recommended cases compared to the query case from the average *Better* scores for the recommended products. Obviously this is biased towards the *Better* technique because it uses this very metric to rank its own recommendations. Nevertheless it provides a useful guide to understanding the extent to which there is room to improve upon alternative strategies. The results are presented in Figure 3 as a graph of the average *Better* scores for recommended cases for increasing large recommendation lists and for each of the recommendation strategies above; we use a *k-comparability* threshold of $k = 15$. We can see that there is a significant uplift in the relative sentiment of the *Better* recommendation lists, which is sustained, albeit decreasing, over all values of n. For instance,

for *Laptops* we can see that for recommendation lists of size 3 the *Better* strategy recommends cases that are more positively reviewed on average than the query case ($Better(Q, P) = 0.11$). By comparison the cases recommended by the *Jaccard* or *Cosine* strategies offer little or no sentiment improvement.



(a) GPS

(b) Laptop

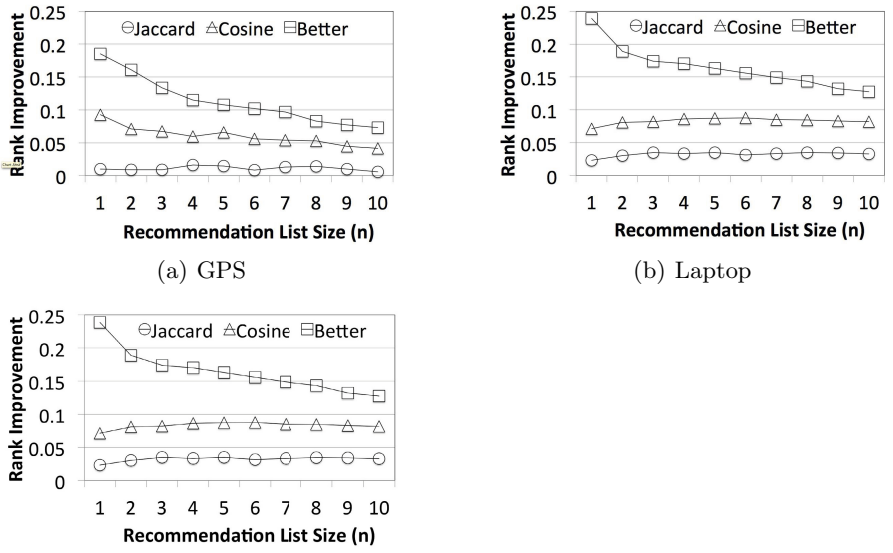
(c) Tablet

Fig. 3. Mean relative sentiment versus recommendation list size ($k = 15$)

Residual Sentiment. As an aside it is also worth considering the relative sentiment of those features in the query case and the recommended cases that are *not* among the k or more shared features. We can estimate a *residual sentiment score* (RSS) for these features by subtracting the average sentiment score for the residual query case features from the average sentiment score of each of the recommended cases' residual features, such that an $RSS > 0$ means that the residual features of the recommended cases tend to be better than those of the query case. These results are also shown in Figure 3 for each of the recommendation strategies but using dashed lines. The RSS values tend to fall close to zero in most scenarios especially for larger values of n, thereby indicating that there is little lost sentiment due to these residual features. Nonetheless there does appear to be some limited sacrifice associated with the *Better* strategy since its RSS scores tend to be slightly less than 0 for the GPS product case base. In general, however, this type of tradeoff seems justified given the much improved sentiment scores of the shared features for the *Better* strategy.

4.4 Recommendation Quality as Relative Ratings Differences

Obviously the above results are somewhat limited by the fact that our measure of recommendation quality (sentiment improvement) is closely coupled to the ranking metric used by the *Better* strategy. As an alternative, in this section we consider our second ground-truth — the average user-provided product ratings — as a truly independent measure of recommendation quality. In other words, we can evaluate recommendation quality in terms of whether or not recommended cases tend to attract higher overall product ratings than the query case. Rather than report changes in product ratings directly we look at the rank improvement of a recommended product relative to the query case with respect to the rating-ordered list of products in the case base. In other words we rank all cases in a case base by overall ratings score and then calculate the quality of a recommended case in terms of its percentage rank difference to the query case. Thus if the query case is ranked 50th in a case base of 100 products by rating and a recommended case is ranked 25th then the relative rank improvement is +25%. We do this because it provides a more consistent basis for comparison across different case bases with different numbers of products and ratings distributions.



(a) GPS

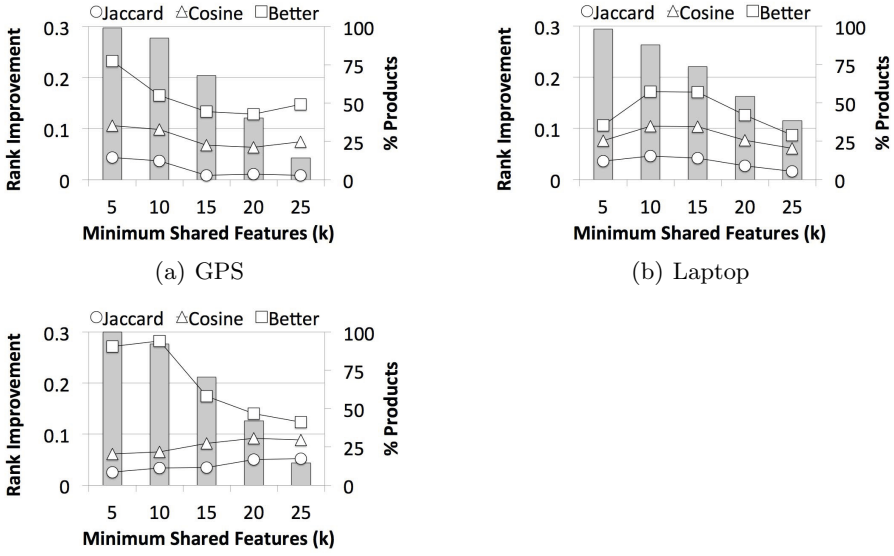
(b) Laptop

(c) Tablet

Fig. 4. Mean relative rank improvement versus recommendation list size($k = 15$)

Relative Ratings Ranks. The results are presented in Figure 4 as the relative rank difference (or improvement) versus n (for $k = 15$). These results demonstrate a clear benefit to the *Better* strategy for all values of n and across all product types. For example, for *Laptops* we can see that the *Better* strategy

tends to recommend cases with a relative rank improvement of between 24% ($n = 1$) and 13% ($n = 10$); given the 314 laptop cases this means that this strategy is recommending cases that are, on average, up to 75 rank positions better than the query case in terms of overall product rating ($n = 1$). This is a significant improvement compared to the baseline strategies, which achieve a rank improvement of only about 4% or 12-13 rank positions (*Jaccard*) and 9% (or 28 rank positions) for *Cosine*. In other words for each recommendation cycle the *Better* strategy is capable of suggesting new cases that are objectively better than the current query case. It is also worth highlighting that this particular quality measure is obviously considering the 'whole product' quality of recommended cases; we do not need to separately consider the quality of the shared and residual features because user ratings are applied at the product case level and not at the feature level. Thus, even though our recommendations are made on the basis of a set of $k = 15$ or more shared features we can say with some certainty that the products recommended by *Better* are better overall than the query product, and not just with respect to the features that they share with the query case.



(a) GPS

(b) Laptop

(c) Tablet

Fig. 5. Mean relative rank improvement versus minimum shared features and average percentage of product cases retrieved ($n = 3$)

Relative Rank Versus *k*. We have yet to consider the impact of k (the minimum number of overlapping features between query and retrieved cases) on recommendation quality. By increasing k we can narrow the scope of comparable products considered for recommendation. This will inevitably limit our ability

to recommend products that offer large improvements in quality relative to the query product; it is the nature of product spaces that competition increases as one narrows the product focus, thereby offering less scope for improvement from one product to the next. To test this hypothesis we fix the number of recommended cases ($n = 3$) and compare the relative rank improvement for different values of k, as shown in Figure 5; for reference, we also include a bar chart of the average number of cases retrieved at these k values. Once again we can see a consistent benefit accruing to our *Better* strategy compared to *Cosine* and *Jaccard*. And as predicted, by and large, the relative rank improvement tends to decrease for increasing k. For example, for *Tablets* we see that the relative rank difference of 27% (around 45 rank positions) for $k = 10$ falls to about 12% (or 20 rank positions) for $k = 25$. This compares to rank improvements that are less than 10% for *Cosine* and hardly more than 5% for *Jaccard*, a comparison that is broadly repeated for the *GPS* and *Laptop* case bases too.

4.5 Query Case Similarity

These recommendation quality improvements demonstrate the ability of the *Better* strategy to recommend higher quality products than both alternatives. This is very encouraging but one point that is not clear is the balance between query similarity and this sentiment improvement. If, for example, the cases recommended by *Better* were very different from the query then perhaps these sentiment improvements would be less appealing. To explore this Figure 6 presents the average similarity between the query case and the top n ($n = 10$) recommended cases for the 3 case bases; the standard Jaccard similarity metric is used over features in the query case and the recommended cases.

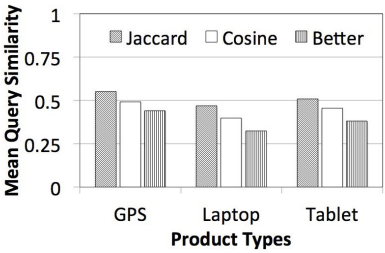


Fig. 6. The average feature similarity between the query case and recommended cases ($n = 10$) for *Jaccard*, *Cosine* and *Better* strategies ($k = 15$)

The results show that there is a reduction in query case similarity for the *Cosine* and *Better* techniques, as expected. However the scale of the reduction is small since it corresponds to only about 2 or 3 features of difference between *Jaccard* and *Better*, for example. And in practice the type of products recommended by *Better* are similar to those recommended by the other techniques.

Moreover, we know from the results above that this relaxation in similarity translates into significant improvements in the quality of recommended cases.

5 Discussion

Our core contributions are: (1) mining product features from user-generated reviews; (2) assigning sentiment to these features to produce experiential product cases; and (3) a novel approach to recommendation that combines product similarity and sentiment to improve recommendation quality.

We believe that the combination of these contributions is important for a number of reasons. Conceptually it keeps the spirit of experience reuse that is the core of CBR, but which is often not deeply ingrained in more traditional case-based recommenders; our product cases are fundamentally *experiential* in the sense that they are based wholly on the experiences of the users of these products. Furthermore, the combination of similarity and sentiment during retrieval facilitates the prioritisation of cases that are not only similar to a query case but also objectively better, at least with respect to the views and usage experiences of product owners. Moreover, the proposed approach is eminently practical: user generated reviews are plentiful even if the type of technical feature specifications used in more traditional product case representations are not. And this means that our approach can be readily deployed in most real-world settings without the need for additional knowledge. Finally, the approach is adaptive and self-regulating. As product reviews accumulate over time the views of users on a particular product or feature may change, and these changes will be reflected in the product cases as they are regularly re-generated from the evolving review-set.

Our results show this approach to be practical and it delivers strong product recommendations that are objectively better than the query, instead of just similar to it. This is important when helping users to *explore* a product space during the early stages of their pre-purchase research. These users are unlikely to have a clear picture of the product they want. The role of the recommender is to help them to explore the trade-offs within the product-space but without prematurely narrowing their search. Critiquing-based recommenders [12, 19] and other forms of conversational recommenders [11, 20] do support this type of discovery, but are based on fixed feature similarities. Our approach combines similarity and sentiment and allows to guide recommendation by quality rather than just similarity. Thus, a user who starts with a point-and-shoot camera might be guided towards more flexible and powerful DSLR models based on superior picture quality, flexibility, and general price-performance features. These products might not be considered in more traditional case-based recommenders due to the lack of similarity between point-and-shoot and DSLR market segments.

In a practical setting for the product types considered in this paper a k comparability score of 15 to 25 provides the right balance of similarity (between the query and retrieved cases) and opportunity for recommendation improvement. These levels of minimum feature overlap provide a suitable basis for case comparison. They constrain the type of cases that are retrieved to be more or less

related to the query case and at the same time include cases that are likely to offer improved quality. Smaller values of k provide even greater opportunities for higher quality recommendations but run the risk that the retrieved cases will no longer be sufficiently similar to the query case.

6 Conclusions

In this paper we have described a novel approach to case-based product recommendation. Experiential cases are automatically mined from plentiful user-generated product reviews as the basis for a novel sentiment-based product recommendation strategy. We have demonstrated the benefits of this approach across a number of product domains, in a realistic recommendation setting, and using objective real-user judgements as an objective ground-truth. The results are very promising:

- The generated cases are feature-rich, in the sense that typical cases include 25-30 distinct features and corresponding sentiment scores;
- There is a reasonably dense pattern of overlapping features between cases, thus providing a strong basis for comparison and recommendation;
- It is possible to make recommendations that represent significant improvements in quality with respect to the query case.

In closing our aim has been to describe and demonstrate the viability of a novel approach to case-based product recommendation. But in doing so we have only taken the first step in what has the potential to be a powerful general approach to recommendation on the *experience web*. There is much potential to improve and extend this work by exploring different techniques for topic mining and feature extraction, for example, or alternative ways to evaluate and aggregate sentiment. And of course there are many opportunities to further improve case retrieval, for instance by exploring the use of different feature weighting models. These and other matters reflect our current priorities for future research.

Acknowledgments. This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

References

1. Zhu, F., Zhang, X.M.: Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics. Journal of Marketing 74(2), 133–148 (2010)
2. Dhar, V., Chang, E.A.: Does chatter matter? the impact of user-generated content on music sales. Journal of Interactive Marketing 23(4), 300–307 (2009)
3. Dellarocas, C., Zhang, M., Awad, N.F.: Exploring the value of online product reviews in forecasting sales: The case of motion pictures. Journal of Interactive Marketing 21, 23–45 (2007)

4. Kim, S.-M., Pantel, P., Chklovski, T., Pennacchiotti, M.: Automatically assessing review helpfulness. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Sydney, Australia, July 22-23, pp. 423–430 (2006)
5. Baccianella, S., Esuli, A., Sebastiani, F.: Multi-facet rating of product reviews. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 461–472. Springer, Heidelberg (2009)
6. Hsu, C.-F., Khabiri, E., Caverlee, J.: Ranking comments on the social web. In: International Conference on Computational Science and Engineering, CSE 2009, vol. 4, pp. 90–97. IEEE (2009)
7. O'Mahony, M.P., Smyth, B.: Learning to recommend helpful hotel reviews. In: Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys 2009), New York, NY, USA, October 22-25 (2009)
8. Lim, E.-P., Nguyen, V.-A., Jindal, N., Liu, B., Lauw, H.W.: Detecting product review spammers using rating behaviors. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 939–948. ACM, New York (2010)
9. Li, F., Huang, M., Yang, Y., Zhu, X.: Learning to identify review spam. In: Proceedings of the Twenty-Second international Joint Conference on Artificial Intelligence, IJCAI 2011, vol. 3, pp. 2488–2493. AAAI Press (2011)
10. O'Callaghan, D., Harrigan, M., Carthy, J., Cunningham, P.: Network analysis of recurring Youtube spam campaigns. In: ICWSM (2012)
11. Bridge, D., Göker, M.H., McGinty, L., Smyth, B.: Case-based recommender systems. The Knowledge Engineering Review 20(03), 315–320 (2005)
12. Reilly, J., McCarthy, K., McGinty, L., Smyth, B.: Dynamic critiquing. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 763–777. Springer, Heidelberg (2004)
13. Smyth, B., McClave, P.: Similarity vs. diversity. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 347–361. Springer, Heidelberg (2001)
14. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004, pp. 168–177. ACM, New York (2004)
15. Justeson, J., Katz, S.: Technical terminology: Some linguistic properties and an algorithm for identification in text. In: Natural Language Engineering, pp. 9–27 (1995)
16. Hu, M., Liu, B.: Mining opinion features in customer reviews. Science 4, 755–760 (2004)
17. Moghaddam, S., Ester, M.: Opinion digger: An unsupervised opinion miner from unstructured product reviews. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 1825–1828. ACM, New York (2010)
18. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295. ACM (2001)
19. Burke, R.D., Hammond, K.J., Young, B.: The findme approach to assisted browsing. IEEE Expert. 12(4), 32–40 (1997)
20. Thompson, C.A., Goeker, M.H., Langley, P.: A personalized system for conversational recommendations. J. Artif. Intell. Res. 21, 393–428 (2004)