CHIC: A Combination-based Recommendation System

Manasi Vartak
CSAIL, Massachusetts Institute of Technology
mvartak@mit.edu

Samuel Madden
CSAIL, Massachusetts Institute of Technology
madden@csail.mit.edu

ABSTRACT

Current recommender systems are focused largely on recommending items based on similarity. For instance, Netflix can recommend movies similar to previously viewed movies, and Amazon can recommend items based on ratings of similar users. Although similarity-based recommendation works well for books and movies, it provides an incomplete solution for items such as clothing or furniture which are inherently used in combination with other items of the same type, e.g., shirt with pants, and desk with a chair. As a result, the decision to buy a clothing or furniture item depends not only on the item itself, but also on how well it works with other items of that type. Recommending such items therefore requires a *combination-based* recommendation system that given an item, can suggest interesting and diverse combinations containing that item. This problem is challenging because features affecting combination quality are often difficult to identify; quality, being a function of all items in the combination, cannot be computed independently; and there are an exponential number of combinations to explore. In this demonstration, we present CHIC, a first-of-its-kind, combination-based recommendation system for clothing. The audience will interact with our system through the CHIC mobile app which allows the user to take a picture of a clothing item and search for interesting combinations containing the item instantly. The audience can also compete with CHIC to create alternate ensembles and compare quality. Finally, we highlight via visualizations the core modules of CHIC including model building and our novel search and classification algorithm, C-Search.

Categories and Subject Descriptors

H.4.0 [**Information Systems**]: Information Systems Applications-General

General Terms

Design

Keywords

Recommendation, Combination

SIGMOD'13, June 22–27, 2013, New York, New York, USA.

1. INTRODUCTION

Recommendation systems are widely used by e-commerce websites with the goals of reducing information overload, converting browsers to buyers and cross-selling by suggesting additional items to buy [10]. Most state-of-the-art recommender systems are focused on recommending items based on item or user similarity. For example, Netflix can recommend movies similar to previously-viewed movies, Amazon can recommend items based on ratings of similar users, and Pandora suggests music based on music you've liked before. Although these systems work well for items such as books, music and movies, they provide an incomplete solution for items such as clothing or furniture that are used in combination with other items of the same type. In such cases, the aesthetic appeal of an item depends on the items it is paired with, e.g., a shirt "looks good" with matching jacket and pants, while a sofa may "go with" a matching chair. As a result, the decision to buy a clothing or furniture item depends not only on the item itself, but also on how well it works with other items of that type. Therefore, recommending such items requires, in addition to similarity-based recommendation, a *combination-based* system that given an item, can suggest interesting and diverse combinations containing that item.

1.1 Motivating Examples

Example 1. *Alice is shopping on Amazon.com for a new sweater, and has identified a few sweaters that meet her criteria of color, size and price. While current recommendations shown to Alice (Figure 1) offer her other options based on top-selling sweaters, similar sweaters and items bought by similar customers, Alice is offered no insight into how the sweater could be paired with other items from the store (or items she owns) or if she could buy accessories for a more aesthetically appealing combination.*

While it might seem that "Items frequently bought together", i.e. frequent item-sets, would be able to provide combination-based recommendations, as highlighted in Figure 1, the recommendations are merely of similar clothing items instead of interesting ensembles of clothes. Without a clear demonstration of how the sweater might add to her existing set of clothes, Alice may decide not to purchase the item, leading to lost potential sales of not only the sweater but also accessories that would have been cross-sold through ensembles.

Example 2. *James is choosing furniture for this new home and visits Ikea to choose a sofa and coffee table. He has some fixed criteria such as material, price and dimensions while buying these items. However, even when an item meets the above criteria, the decision to buy the furniture strongly depends on whether the new sofa*

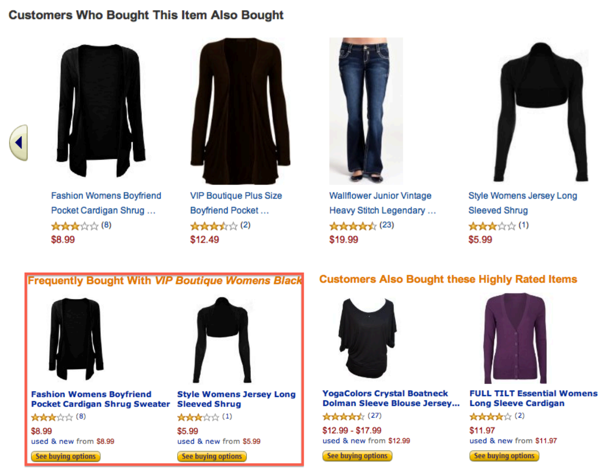

Figure 1: Recommendations for similar items, highly-rated items and items bought together

and coffee table will match existing furniture like chairs and carpets, and co-ordinate with wall colors. As a result, combination-based recommendation can successfully be used to help James identify the right furniture items.

In this demonstration, we focus on the problem of *combination-based* recommendation for clothing items.

1.2 Problem Description

E-commerce websites can significantly increase the conversion rate from browsers to buyers and increase cross-selling by implementing a *combination-based* recommendation system. We divide the problem of recommending combinations or ensembles into two sub-problems: (a) model building, and (b) searching and classifying combinations based on quality. The goal of the former problem is to build an accurate model predicting the "interesting"-ness a combination of items. For the purpose of this work, we frame problem (a) as a binary classification problem where we model the quality of a combination as a function of the hybrid features of that combination (such as degree of color matching and contrast, degree of texture diversity etc). We can then formally describe the search and classification problem (b) as follows: given an item, find k high quality combinations *containing that item*, where quality is determined by the classification model built in part (a).

Combination-based recommendation is a challenging problem for several reasons. First, it is extremely difficult to accurately identify features affecting combination quality. Second, since combination quality is a function of all constituent items of the combination (e.g., degree of color match, weather appropriateness), it cannot be computed independently for each item. And third, there are an exponential number of combinations to explore, making the naive solution computationally intractable.

1.3 State-of-the-Art Approaches

The problem of processing images of clothing items to find similar items has been explored in the computer vision literature through work such as [4]. Similarly, there has been some work on performing matching of clothing items on a limited scale. [3, 6] propose techniques based on tags to identify matches of clothes, while [13] proposes a technique to perform clothes matching (based on color and texture) for the blind. These systems rely on simple item-matching models, and it is unclear how they could be scaled to large datasets or multiple categories of items. Finally, [9] focuses on identifying the latest clothing trends by comparing pictures for similar features. [11] designed a system that suggests outfits based on occasion. Polyvore[1] is a popular fashion platform that allows users to manually mix and match diverse items to create interesting *sets*. However, Polyvore does not actually perform recommendations of matched sets of items.

Combination-based recommendation is related to traditional recommendation algorithms such as collaborative filtering, knowledge-based recommendation and content-based recommendations [7]. Similarly, there has been seminal work in the database community on mining frequent itemsets [1], but as noted above, these algorithms do not correctly address the problem of producing high-quality combinations. Finally, combination-based recommendation is also related to the top-k ranking problem. While several efficient top-k algorithms have been proposed [5, 8, 2], including one for groups [12], these algorithms do not address the unique challenges posed by combination-based recommendations.

In this demonstration, we present CHIC, a first-of-its-kind, combination based recommendation system for clothing.

2. CHIC OVERVIEW

CHIC produces high-quality recommendations through a two step process: first, it uses a crowd-sourced and web-scraped dataset to learn a predictive model for computing the quality of combinations; then, it combines the predictive model with our novel algorithm, called C-Search, to efficiently compute and classify combinations. Figure 2 shows the system architecture of our system.

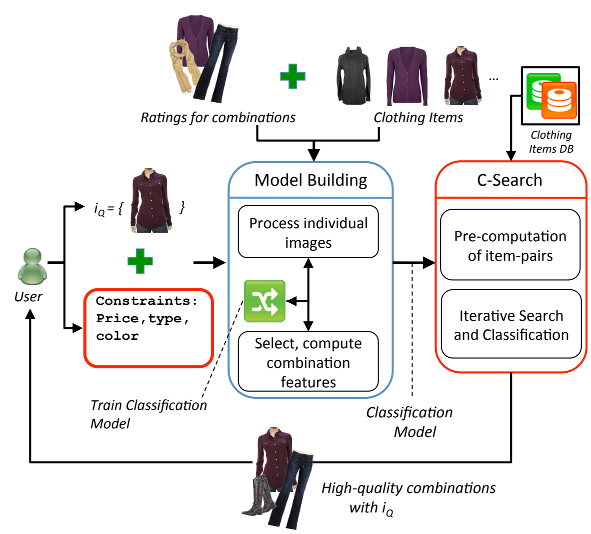

Figure 2: CHIC: System Architecture

2.1 Model Building

The model underlying CHIC is built using a large ratings dataset that has been crowd-sourced and scraped from the web. For building the model, we perform feature extraction at two levels. First, the individual clothing item images are processed using techniques such as color histogram computation, background detection, Gabor filters etc. (as described in [4]) to extract basic features including set of colors, texture, print and shape. We also compute additional

[1] www.polyvore.com

features such as material of the item, price etc. from the item description. Next, given a combination of items, we compute the hybrid features for that combination from the features of the constituent images. We then construct a classification model predicting the combination rating from the hybrid features. We began with a potential feature set of over 50 features and used forward-selection to identify relevant features useful in predicting combination quality.

2.2 C-Search: Search and Classify

When producing recommendations based on combinations, there are two main options for implementing the system. The first is a pre-computation-based approach where we pay an extremely large, but one-time, pre-computation cost to exhaustively make and evaluate all possible combinations of items. Searches then merely turn into lookups. Although this approach has excellent query time, downsides include the extremely large cost of computing all combinations, extracting relevant features, and storing the combinations. Most online stores have tens of thousands of clothing items in each category such as outerwear, sweaters etc. (e.g. Amazon has 60K tops) and therefore the total number of combinations, ($6 * 10^{47}$ in the case of Amazon) is intractable. Moreover, when new items are added to the database, we must pay a large price to incorporate them into the system. A few optimizations that make this approach tractable include storing the top combinations for each item and adopting a nearest neighbor approach for new items.

The reverse approach is to not perform any pre-computation, and instead, only compute combinations containing a particular query item on the fly. This has the advantages of a negligible upfront cost and potentially fewer combination evaluations using bounding properties of the classification model. However, even though this approach will not exhaustively test combinations, the large number of items that can be matched with any particular item leads to unacceptable response times.

In CHIC, we choose the best of both worlds by partially pre-computing combinations, so that they are computationally tractable, and then building larger combination from seed combinations.

2.2.1 *Pre-computation*

We break down the problem of computing high-quality combinations into the smaller problem of computing high quality pairs of items and then iteratively growing these pairs. Continuing the previous calculations, pre-computing pairs translates to the evaluation of 1.2M pairs of items, as opposed to 10^{47} combinations, which is very much tractable. We construct a regression model from individual item features to predict how much a pair of items "match". The regression scores are then used to rank pairs while constructing larger combinations.

2.2.2 *Search and Classification*

Given a new query item, CHIC undertakes an iterative search that builds combinations of progressively larger size. Suppose CHIC is supplied a single item, i_Q, as a query and that the item is not present in the CHIC database. CHIC uses nearest neighbor search to find the image i_{DB} in the CHIC database that is closest to the query item. CHIC then locates high quality pairs containing i_{DB} (or constructs them if i_{DB} hasn't be queried before). It then combines these high quality pairs with a ranked list (based on regression scores) of item pairs to select high quality combinations. This process is repeated until a complete combination is obtained.

Once a combination has been obtained, we compute the hybrid features for that combination and run our classification model on it. Since we have built the combination from high-quality pairs, the probability of the combination being of good quality is high. However, we run a set of "blacklist rules" before applying our classification model to further reduce unnecessary computation.

3. DEMONSTRATION

In our demonstration, we present a real user scenario that demonstrates the importance of *combination-based* recommendations and the novelty of our solution. For this purpose, we have created an iPhone app which allows the user to take pictures of clothing items and search for interesting combinations instantly. The user can also use the app to search for complete item combinations by specifying criteria such as color, price, type etc. The audience can compete with CHIC by creating alternate ensembles and comparing quality. We visualize the core algorithms of CHIC to demonstrate the key ideas and innovative strategies used to generate high-quality combinations.

3.1 Demonstration Setup

We have implemented the CHIC system via a Django backend that is accessed through our app. The website can also be used directly to perform search. We have implemented an additional module on the website that allows users to manually mix and match items to create combinations. The audience will use this interface to compete with CHIC. The search module of CHIC takes as input a user query in the form of either an incomplete combination of clothing items or some criteria for building a full combination (e.g. color, price etc.) and produces a set of high-quality combinations. Manual matching and CHIC run on the same dataset.

The audience will interact with CHIC through three interfaces: (i) the CHIC mobile search interface, (ii) the web-based manual mix and match interface, and (iii) the CHIC system interface.

3.1.1 *CHIC Search Interface*

The CHIC web interface allows users to pick items from our pre-existing database and search for interesting combinations containing those items. The CHIC mobile app is more interactive and allows users to take pictures of any clothing item and ask the system to suggest high-quality combinations with that item.

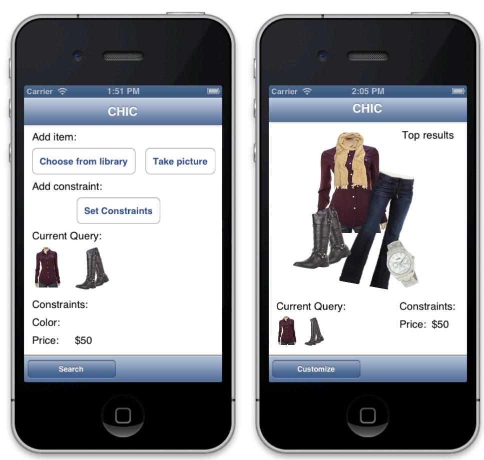

Figure 3: CHIC: Search Interface

3.1.2 *Manual Matching View*

The manual matching view is used by the audience to compete with CHIC to create high-quality combinations. Various filters are available to aid the audience in creating combinations. CHIC and manual matching are run in parallel and we provide the audience real-time feedback about the quality of the combination as well as time taken for matching.

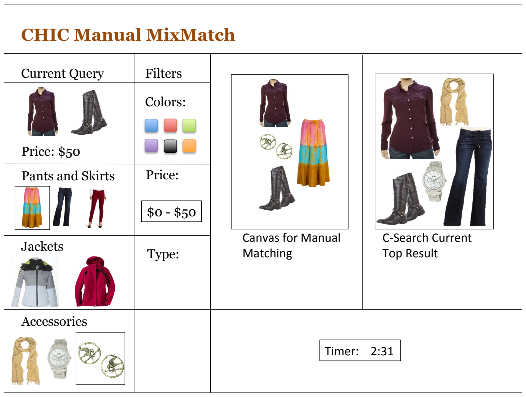

Figure 4: CHIC: Manual Matching Interface

3.1.3 *CHIC System View*

We use the CHIC System View to present run-time progress of C-Search and data about past computations. Specifically, we visualize the model being used and the key steps in C-Search. This includes the pre-computed item pairs, steps in iteratively generating combinations and the classification model.

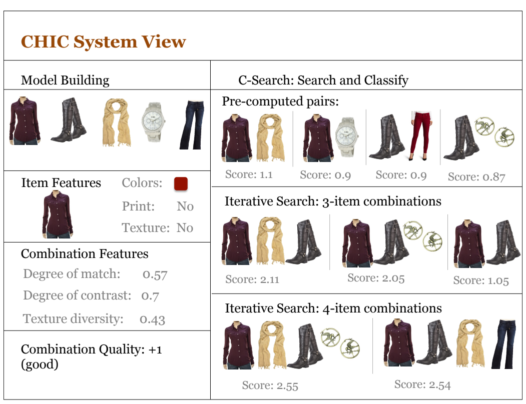

Figure 5: CHIC: Manual Matching Interface

3.2 Walkthrough Example

The CHIC demo consists of the following steps:
Step 1: The user takes a picture of a clothing item and optionally supplies constraints that the combination must satisfy (e.g. color, price etc.) as in Figure 3.
Step 2: As soon as the user submits the query to CHIC, the manual matching view (Figure 4) becomes available for the user to manually create interesting combinations. As C-Search proceeds, at each time step, we display the best combination found so far along with its quality. In the manual matching view, the user also gets information about the quality of his/her combination so far. Various filters are available to aid in this process.
Step 3: As CHIC search is running, the user can use the System View (Figure 5) to explore the model being used and to understand how the iterative search is proceeding. In particularly, we illustrate feature extraction, iterative combination generation, and classification.
Step 4: The user can also browse interesting combinations produced by CHIC when assigned ad-hoc queries supplying constraints such as type, color and price.

4. CONCLUSION

In this demonstration, we present CHIC, a first-of-its-kind combination based recommendation system. CHIC uses a specialized model building module along with an innovative search algorithm, called C-Search, to produce high quality combinations. Our demo presents the audience with a real-world scenario that illustrates the importance of combination-based recommendation and provides hands-on interaction with our system. Our visualizations of core algorithms illustrate the inner workings of our system.

5. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, 1993.

[2] S. Chaudhari and L. Gravano. Evaluating top-k selection queries. In *VLDB*, pages 397–410, 1999.

[3] C.-I. Cheng and D. S.-M. Lio. An intelligent clothes search system based on fashion style. In *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, Kunming*, 2008.

[4] E. Hsu, C. Paz, and S. Shen. Clothing image retrieval for smarter shopping. In *Stanford EE368*, 2011.

[5] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid. Supporting top-k join queries in relational databases. In *VLDB*, pages 754–765, 2003.

[6] T. Iwata, S. Watanabe, and H. Sawada. Fashion coordinates recommender system using photographs from fashion magazines. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[7] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender Systems: An Introduction*. Cambridge University Press, New York, NY, USA, 1st edition, 2010.

[8] C. Li, K. C.-C. Chang, I. F. Ilyas, and S. Song. Ranksql: Query algebra and optimization for relational top-k queries. In *SIGMOD*, pages 131–142, 2005.

[9] X. Li, H. Yao, X. Sun, R. Ji, X. Liu, and P. Xu. Sparse representation based visual element analysis. In *International Conference on Image Processing*, 2011.

[10] J. B. Schafer, J. Konstan, and J. Riedi. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic commerce*, 1999.

[11] E. Shen, H. Lieberman, and F. Lam. What am i gonna wear?: Scenario-oriented recommendation. In *Proceedings of the 12th international conference on Intelligent user interfaces*, 2007.

[12] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang. Probabilistic top-k and ranking-aggregate queries. *ACM Transactions on Database Systems*, 33(3), 2008.

[13] Y. Tian and S. Yuan. Clothes matching for blind and color blind people. In *International Conference on Computers Helping People with Special Needs*, 2010.