# Finding Similar Products in E-commerce Sites Based on Attributes

Urique Hoffmann, Altigran da Silva, and Moisés Carvalho

Instituto de Computação
Universidade Federal do Amazonas
Manaus, Brazil
{uhsa,alti,moises}@icomp.ufam.edu.br

**Abstract.** We present a preliminary study on the problem of finding products similar to a product given as input, based solely on their attributes. We assume that we are given a set of products from a same category of a same on-line store, were each product is described in a catalog by a number of attributes (e.g., general characteristics, technical specifications, etc.). This problem, which at a first glance may be seen as straightforward or even mundane, is in fact challenging and intriguing. In fact, any automatic solution for it requires techniques for comparing tens of different atributes, whose semantics are often very technical and specific (e.g., the shutter speed of a camera) and also requires dealing with hundreds of products in the category. To be generic, such a solution must also deal with several distinct product categories. In here, we describe and evaluate a similarity function we have proposed for comparing products based on their attributes. This function uses a number of attribute-specific similarity functions, which are selected according to a class assigned to the attribute. The assignment of classes to attributes is carried out by a simple classification strategy, which we also describe and evaluate. Experiments we carried out to evaluate our proposed similarity function using data from real catalogs in five distinct popular product categories have shown promising results.

**Keywords:** Similarity Functions, E-Commerce, Recommender Systems

## 1 Introduction

Recommendation Systems are used by most e-commerce sites to suggest products to their users and provide additional information to help customers to decide which products to acquire [8]. Products can be recommended based on several different types of information such as top overall sellers on a site, customer's demographics, customer's past buying behaviour, or product attributes, e.g., technical specifications, general characteristics, brand, etc. [8]. Recommendations based on this last type of information are called *content-based* or *knowledge-based* recommendations [3].

A simple way of enabling content-based recommendation is, given a product, presenting to the user other products that are similar to it with respect to their attributes. This is useful, for instance, when costumers explicitly want to find products with certain characteristics or when the seller wants to present to a customer products similar to a product she is interest in, e.g., for the sake of comparison or to provide alternatives to out-of-stock itens.

However, in typical e-commerce sites, looking for similar products may require the user to browse manually through a large number of pages and products. For instance, suppose a user is interested in a specific camera, say, "Nikon S3500". Currently, if this user wants to find alternative cameras that are similar to this model (i.e., having similar features), for the sake of comparing their prices, it is likely that she would have to browse over hundreds of other cameras in the catalog to find them. On the other hand, if this camera is not in stock, it would be interesting to provide the user with similar alternative cameras in stock, without having her to look over the whole catalog.

Another interesting aspect of this kind of recommendation is that it enables suggesting products to the customers without relying on historical data. It means that the system can recommend products and provide buying options even if a costumer is new to the system or if the item is new to the catalog.

To find whether two products are similar it is necessary to compare them. Products on e-commerce sites are often described by their *attributes*. It means that, to make a comparison between two products, it is necessary to compare their attributes. This can be unfeasible to be carried out manually by casual users on the Web.

For instance, in a certain e-commerce site, to verify whether the "Nikon S3500" camera is similar to another camera, say the "Sony W830", a user has the option of comparing the 26 atributes provided for the first camera with the corresponding attributes of the second cameras. The lists of attributes available for each camera in this site are presented in Figure 1. Notice that the second camera has only 18 attributes. Also, notice that many attributes are difficult to be compared, unless the user is an expert in the field.

In general, the same situation occurs in many other categories, that is, comparing products requires comparing tens of attributes, some of them with very specific semantics.

In this paper we present a preliminary study on the problem of finding products similar to a given product. We assume that we are given a set of products from a same category of a same on-line store, along with their attributes. For instance, one of the datasets used in our experiments comprises a set of 489 camera models under the *Cameras* category of a real on-line store.

Specifically, we describe and evaluate a generic similarity function we have proposed for comparing products based on their attributes. This function uses a number of attribute-specific similarity functions, which are selected according to a class assigned to the attribute. The assignment of classes to attributes is carried out by a simple but effective classification strategy, which we also describe and evaluate here.

An experimental evaluation we carried out and reported here has shown promising results. Our proposed similarity function showed to be accurate in finding similar products, achieving average F-1 values above 0.75 in 5 representative product categories we have tested. Also, our strategy for attribute classification has correctly classified most of the attributes from these categories.

| **Attribute** | *Nikon S3500* | *Sony W830* |
|---|---|---|
| Brand | Nikon | Sony |
| Type of Camera | Compact | Digital Camera |
| Monitor/Display | 2,7” LCD / TFT 230.000 | 2.7”-LCD TFT-Clear Photo LCD |
| Resolution | 20,1 | 20,1 |
| Internal Memory | 25MB | 27MB |
| Memory Cards | Yes | Yes |
| Compatible Memory Cards | SD, SDHC and SDXC | Memory Stick Duo, Memory Stick PRO Duo (High Speed) |
| Sensor | CCD 1/2, 3 inch. | Super HAD CCD |
| Optical Zoom | 7x | 8x |
| Digital Zoom | 4x | 32x |
| Lenses | Crystal NIKKOR 26-182mm fixed | - |
| Shutter Speed | 1/2000 - 1 s 4 s | - |
| Focus range | [W]: Aprox. 50 cm/[T]: Aprox. 1 m . . . | - |
| Opening | f/3.4-6.4 | - |
| Flash Modes | Automatic TTL Flash with pre-flash monitor | Auto/On/Off/ Slow Syncro / Flash Extended |
| Flash range | [T]:1,0 to 2,1m (3 feet 4 inch. to 7 feet 1 inch.) . . . | ISO Auto: Aprox. 0.3m to 2.8m |
| Battery Type | Rechargeable Li-ion Battery EN-EL19 | Battery Charger Adapter, Power Cable |
| Video Features | Full HD: 1920px1080p/30 / HD: 1280px720p/30 . . . | - |
| Scene modes | Backlight,. . .,Sports, Sunset | Sensitivity/Twilight/. . ./Pets |
| File Formats | .avi,.jpg,.wav | JPEG |
| Built-in microphone | Yes | - |
| Tripod mount | Yes | - |
| Menu Languages | Chinese,Danish,. . ., Arabic | - |
| Color | Purple | Violet |
| Dimensions (HxWxD) | 5,7x9,6x2cm | 9,31x5,25x2,25cm |
| Weight | 129g | 120g |

**Fig. 1.** Attributes available for camera models “Nikon S3500” and “Sony W830” with their values. Some values were truncated to save space.

The remainder of this paper is organized as follows. In Section 2 we review related work. In Section 3 we present our strategy for attribute classification and in Section 4 we present our proposed similarity function. Section 5 reports our experiments and its results. Finally, Section 6 presents our conclusions and directions for future work.

## 2 Related Work

Although important and challenging, effective methods for finding similar products are scarce both in industry and in the academy.

Kagie et. al. [7] proposed a content-based graphical shopping interface based on product attributes to recommend similar products. To use this interface, the user must first define an *ideal* product by providing desired values to its attributes. The interface then shows products considered as similar to this ideal product in a 2D Map. By interacting with this map, the user chooses, from the products plotted, the most similar to the ideal. The interface then recalculates the similarity between the ideal product to all other products in the dataset. This process continues until the interface shows a product the user considers as the most similar. In this work the authors consider only two of attribute classes: categorical and numeric.

Our approach differs from this in many aspects. First, in our approach the user does not need to specify an ideal product. In fact, this is avoided, since we consider that casual users in e-commerce sites are not willing to specify desired values for several attributes. Instead, we only require the user to select one product to be used for comparison. Second, besides categorical and numerical attributes, we consider two additional classes of atributes: *multi-categorial* and *dimensional*. We adopted these two additional attributes classes because they are very common in e-commerce products. Third, in our case there is no need to ask the user to provide the class of each attribute involved in the comparison. Fourth, while Kagie's work seems to focused on a single category, our approach was conceived to deal with many categories typically found in e-commerce sites. Fifth, we instead of using a 2D map with several products, our approach can produce, as output, a ranking of products in order of similarity.

## 3 Attribute Classification

Prior to the application of our similarity function, it is necessary to take each attribute found in the products of a given category we are interested in and assign each one to a single class of a simple attribute taxonomy comprising four classes, namely: *Numerical*, *Categorical*, *Multicategorical* and *Dimensional*.

This taxonomy was created based on previous work by Kagie et. al. [6,7] and in our own experience in dealing with e-commerce catalogs. The original taxonomy by Kagie et. al. in [7] included only *Numerical* and *Categorical* attributes. It was extended in [6] to include the *Multicategorical* class. We further expanded it with the *Dimensional* class to handle the common case of atributes that describe the dimensions of products, displays, etc.

Although a number of different approaches could have been used for this task, we opted for using a simple strategy in which the values expected for the attributes in a given class are described by a regular expression we call *domain descriptors*. Domain descriptors are similar to the *Data Frames* used by Embley et. al. in several methods (e.g., in [1]) and provide a description on how values of attributes of the four classes above are written.

The classification of a attribute is carried out as follows. Let $A_i$ be an attribute that occurs for products $p_1$,. . .,$p_m$ in a given category. For instance, attribute *Scene Modes* occurs in the description of many products in the *Compact Cameras* category. First, for all products $p_j(1{\leq}j{\leq}p)$, we take the value $v_{i,j}$ for $A_i$ occurring in $p_j$.

Next, we perform several cleaning and standardization operations over set of values $v_{i,j}$ of $A_i$ taken from products. These operations include duplicate values removal, white space and case normalization, among others. The result is a set of values $a_1$,. . .,$a_m$ which we call the *occurrences* of $A_i$. Notice that by doing so we assume that all values of $A_i$ have the same semantics in all $p_j$. For instance, we assume that the attribute *Scene Modes* has the same semantics in the description of all products in the *Compact Cameras* category.

Finally, we test each occurrence $a_1$,. . .,$a_m$ against each domain descriptor $\epsilon_k$ ($k$=1,. . .,4) and associate atribute $A_i$ with the atribute class $C_k$ whose domain descriptor $\epsilon_k$ recognizes the majority of its occurrences.

Although simple, this classification procedure is very effective as we demonstrate in experiments we carried out and report later in this paper.

## 4 Similarity Function

Based on the general coefficient similarity proposed by Gower [5], we propose a similarity function for comparing products as the sum of all non-missing similarity scores $s_{ijk}$ over the maximum number of attributes present in one of the products according to Equation 1.

$$\mathcal{S}_{ij} = \sum_{k=1}^{K} m_{ik} m_{jk} s_{ijk} / \max(\sum_{k=1}^{K} m_{ik}, \sum_{k=1}^{K} m_{jk}) \tag{1}$$

In this equation, similarity scores $s_{ijk}$ are computed for every atribute $A_k$ that has value for both products $p_i$ and $p_j$ being compared. Also, $m_{ik}$ ($m_{jk}$) is 0 when the value for attribute $A_k$ is missing for products $p_i$ ($p_j$) and 1 when it is not missing.

The specific functions used for computing the similarity score $s_{ijk}$ depend on the class of the attribute $A_k$. Recall from Section 3 that this class was already defined. For each one of the four atribute classes we defined an appropriate similarity function.

For the *Numerical* class, the similarity function is defined as the absolute difference between the values of the attribute in the two products, as shown in Equation 2.

$$s^N_{ijk} = 1 - \frac{|v_{ik} - v_{jk}|}{\max(v_{ik}, v_{jk}),} \tag{2}$$

where $v_{ik}$ and $v_{jk}$ are, respectively, the values of the attribute $k$ for products $p_i$ and $p_j$.

For the *Categorical* class, the similarity function is defined as

$$s^C_{ijk} = 1(v_{ik} = v_{jk}) \tag{3}$$

implying that objects having the same value get a similarity score of 1 and 0 otherwise.

For the *Multicategorical* class, the similarity function is computed using the Jaccard coefficient [4] between the sets of

$$s^M_{ijk} = \frac{|v_{ik} \cap v_{jk}|}{|v_{ik} \cup v_{jk}|} \tag{4}$$

In this case $v_{ik}$ and $v_{jk}$ denote the sets of individual categorical values composing the actual values. For instance, $v_{ik}$ would be $\{$*Auto*, *On*, *Off*, *Slow Syncro*, $\dots\}$ for the attribute *Flash Modes* in the camera *Sony W830* of Figure 1.

For the *Dimensional* class, the similarity function is the normalized euclidean distance over the dimension values, as described in Equation 5.

$$s^D_{ijk} = 1 - [\sum_{d=1}^{D}((v^d_{ik})' - (v^d_{jk})')^2]^{\frac{1}{2}} \tag{5}$$

where $D$ is the number of dimensions found in the values of the attribute and $v^d_{ik}$ is the value for dimension $d$ in $v_{ik}$ (the same applies to $v^d_{jk}$). For computing this function, each dimension is mean-centered and normalized using

$$(v^d_{xk})' = ((v^d_{xk}) - \mu^d)/\sigma^d \tag{6}$$

$\mu^d$ and $\sigma^d$ are, respectively, the mean and the standard deviation of the set of values of dimension $p$ in all values of atribute $k$, for the products in the category.

As a final comment, it is worth noting that the general coefficient similarity proposed by Gower [5], and latter used by Kagie et. al in [6, 7], is unsuitable to deal with objects with few common attributes. For instance, if directly applied to the problem of comparing products, when two products have just one common attribute and this attribute have the same value in both products, the Gower similarity measure will assign the highest similarity score between these two products. Our function tries to overcome this problem by penalizing the score when the products have few common attributes, as defined in Equation 1.

## 5 Experimental Results

In this section we report the results of experiments we performed to evaluate the attribute classification strategy presented in Section 3, and the similarity function described in Section 4.

### 5.1 Experimental Setup

For the experiments, we have used five datasets provided by Neemu[1], a company that develops search and recommendation technology for major e-commerce sites in Brazil. These datasets comprise five different popular product categories, namely: *Cameras*, *Camcorders*, *Laptops*, *Smartphones* and *TVs*. The product

[1] `http://www.neemu.com`

descriptions available in these datasets often provide many attributes that are not related to the product characteristics themselves. For instance, attributes related to the packing of the products such as, packing dimension, package contents, etc., are very common. Thus, we disregarded these attributes in our experiments. In addition, we removed all atributes that are not found in at least 20% of the products in a given category. By doing so, we tried to increased the percentage of attributes that can be effectively compared to calculate the similarity between products.

Table 1 compares the number of attributes originally available in each dataset and the final number of attributes we considered in each category. Notice that, even though many attributes were removed, still the number of attributes considered is large to be handled manually by humans. This table also presents the number of distinct products available in each dataset.

| Dataset | Products | Initial Attributes | Remaining Attributes |
|---|---|---|---|
| Cameras | 489 | 178 | 28 |
| Camcorders | 41 | 69 | 26 |
| Laptops | 423 | 76 | 28 |
| Smartphones | 147 | 105 | 48 |
| TVs | 244 | 96 | 37 |

**Table 1.** Datasets used in the experiments with the number of products available and the number of attributes considered.

In Table 2, we present the number of attributes in each of the classes of our taxonomy. This classification was carried out manually to be used as a golden standard. Notice that the large majority of the attributes are categorical. This trend was observed in all categories. Also, a single dimensional attribute was available in each category,.

| Dataset | Numeric | Categorical | Multicategorical | Dimensional |
|---|---|---|---|---|
| Cameras | 5 | 16 | 6 | 1 |
| Camcorders | 4 | 14 | 7 | 1 |
| Laptops | 5 | 21 | 1 | 1 |
| Smartphones | 6 | 35 | 6 | 1 |
| TVs | 9 | 24 | 3 | 1 |

**Table 2.** Attributes from the datasets by class.

### 5.2 Attributes Classification

In Table 3, we summarize the results obtained with our attribute classification strategy. For this, we used the well known Precision, Recall and F-1 metrics. In this table, each line corresponds to the results obtained with attributes of a distinct classe, namely, "NUM" (*Numerical*), "CAT" (*Categorical*), "MCA" (*Multicategorical*) and "DIM" (*Dimensionall*).

| Class | Cameras | | | Camcorders | | | Laptops | | | Smartphones | | | TVs | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F | P | R | F | P | R | F | P | R | F | P | R | F |
| NUM | 1.00 | 1.00 | 1.00 | 1.00 | 0.75 | 0.85 | 1.00 | 1.00 | 1.00 | 0.85 | 1.00 | 0.92 | 1.00 | 0.77 | 0.87 |
| CAT | 1.00 | 0.93 | 0.96 | 0.93 | 1.00 | 0.96 | 0.95 | 0.90 | 0.92 | 1.00 | 0.91 | 0.95 | 0.88 | 1.00 | 0.94 |
| MCA | 0.85 | 1.00 | 0.92 | 1.00 | 1.00 | 1.00 | 0.00 | 0.00 | 0.00 | 0.75 | 1.00 | 0.85 | 1.00 | 0.66 | 0.80 |
| DIM | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |

**Table 3.** Experimental Results for Attribute Classification

As it can be notice, our strategy has correctly classified most of the attributes from all categories we tested. We obtained perfect classification in many cases and F-1 values equal or above 0.8 were obtained in all cases but one. This case is the *Multicategorical* class in the Laptops category, which has a single attribute (see Table 2), and our classification strategy missed it. In many cases, the value of some attributes eventually presented noise our cleaning operation was unable to identify and fix Nevertheless, we believe the small number of failures does not compromise the effectiveness of our strategy and, as will see next, did not harm the overall results of our method.

### 5.3 Similarity Measure Evaluation

Evaluating the effectiveness of the similarity measure we described in Section 4 proved to be a challenge by itself. Indeed, carrying out a thorough evaluation to obtain values of Precision, Recall and F-1 would require to compare hundreds of products, examining the values of tens of attributes, some of them very technical. Thus, we opted to evaluate our proposed similarity measure in a task close to its intended application. This task consists in taking a product given as input, using the similarity measure to compare this product with all others in the same category, and verifying if the $k$ products deemed as the most similar are indeed similar to the input product, according to a human-based evaluation. The results are reported in terms of the precision considering these top-$k$ answers, a metric often known as P@$k$. In our case we used $k = 5$, which is reasonable in terms of recommender systems.

For each of the five product categories, we randomly selected 10 products, which we refer to as *query products*, and, for each of them, we examine the 5 most similar products in the same category according to our similarity measure. Thus, a total of 250 pairs of products were manually evaluated. The results are presented in Figure 2.

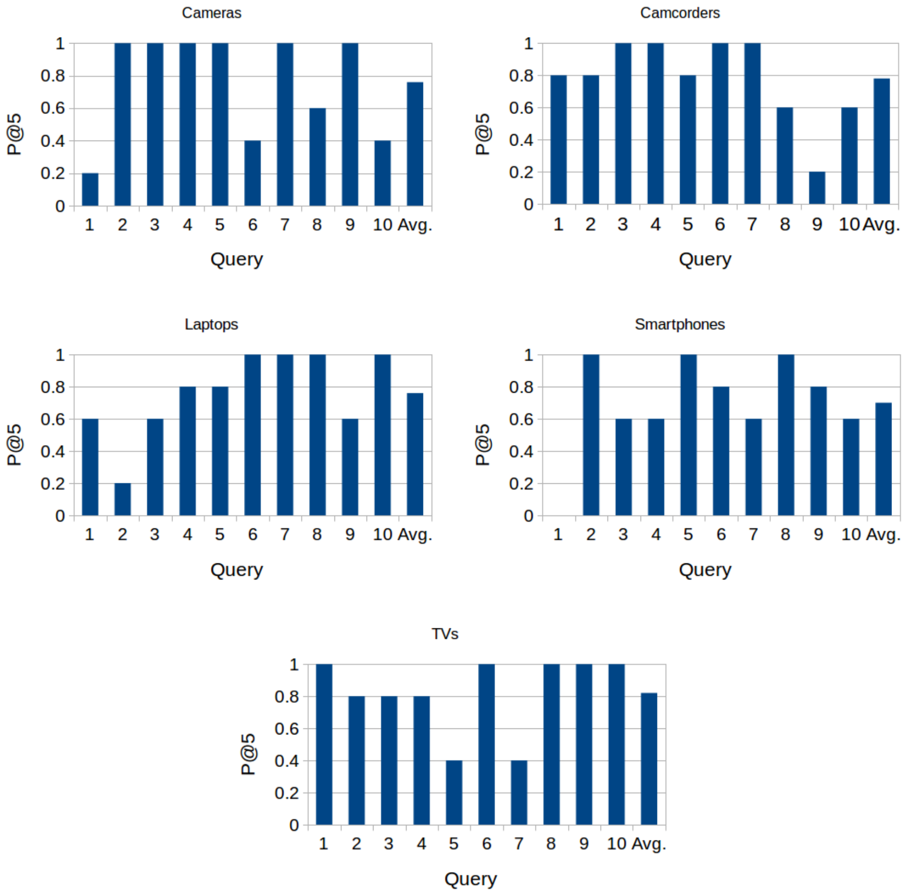

**Fig. 2.** Experimental Results for the Similarity Measure

In Figure 2, each graph corresponds to a product category and shows the P@5 values resulting from each of the 10 query products, along with the average of the ten values. Our similarity measure led to P@5=1 in 22 out of the 50 query products. Only in 8 cases, the P@5 values were below 0.5. In all categories, the average of P@5 values was around 0.75. An average above 0.8 was observed for the TVs category. Notice that the very low P@5 values obtained for some queries (e.g., 0 for query 1 in Smartphones or 1 for query 9 in Cancorders) does not necessarily implies that our similarity measure failed. For instance, it might happen that the query product has very few or none similar product in the catalog. In this case, our function just gave a low similarity score, but no similar products would appear among the top-5 answers. To solve this, a threshold on similarity score could be applied. However, there is no obvious way of imposing this threshold. Thus, we leave this study for future work.

## 6 Conclusions and Future Work

In this paper we presented a preliminary study on the problem of finding products similar to a product given as input. This problem, although important for e-commerce sites, has been ill addressed so far both in the industry and in the academy. We described and evaluated a similarity function we have proposed for comparing products based on their attributes. Our function is generic in the sense that it deals different types of attributes occurring in products from distinct categories. Prior to its application, the function requires that each attribute has been classified into to a class that determines an specific similarity function that handles this attribute. We demonstrate that this classification can be carry out by a simple but highly effective strategy we proposed, which relies of regular expressions. Experiments we have performed with our similarity function with datasests with real products, revealed that it is accurate in finding similar products, achieving average F-1 values above 0.75 in 5 representative product categories.

Our plans for future work address two main aspects. First, we are working on improving the effectiveness of our function by considering that different attributes may have different degrees of importante for users when comparing two products of a given category. Thus, we are investiganting ways for capturing this knowledge from the user and using it to improve our function. For this, we have been working on machine learning techniques, which require training from user data. Thus, the second aspect we are currently addressing is on how to obtain training data without requiring users to label instances specifically for this problem. Another interesting future work we plan to address is considering additional similarity functions for attributes. For instance, in the case of categorical data it is worth investigating the metrics studied in [2].

## References

1. M. Al-Muhammed and D. Embley. Ontology-based constraint recognition for free-form service requests. In *IEEE 23rd International Conference on Data Engineering*, pages 366–375, April 2007.
2. S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the SIAM International Conference on Data Mining*, pages 243–254, 2008.
3. R. Burke. Knowledge based recommender systems. In J. Daily, A.Kent, and H.Lancour, editors, *Encyclopedia of Library and Information Science*, volume 69. 2000.
4. S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 4(1):300–307, 2007.
5. J. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27(4):857–874, 1971.
6. M. Kagie, M. van Wezel, and P. J. Groenen. Choosing attribute weights for item dissimilarity using clikstream data with an application to a product catalog map. In

*Proceedings of the 2008 ACM Conference on Recommender Systems*, pages 195–202, 2008.

7. M. Kagie, M. van Wezel, and P. J. Groenen. A graphical shopping interface based on product attributes. *Decision Support Systems*, 46(1):265 – 276, 2008.
8. J. Schafer, J. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1-2):115–153, 2001.