# An Entity Resolution Framework for Matching Product Offers with Reviews

Chander Jayaraman
IIT Bombay,India
chander@cse.iitb.ac.in

## ABSTRACT

Product review sites are objective sources of information for millions of people who rely on unbiased product reviews to purchase products from online comparison shopping sites. Typically, a large number of product offers from multiple vendors are received continually at these shopping sites. Providing potential customers with reviews from such review sites for the appropriate product is the principal problem we address in this paper. We describe a framework that matches product offers with reviews as an instance of the Entity Resolution problem and show some of the potential challenges that occur in the Product review space. We highlight the design & principal components of the system and present a novel feature annotation approach in the Product review space, where key features embedded in the offer fields are extracted. We also provide detailed experimental results carried on real-world datasets and illustrate the performance of the system using measures for our evaluation.

## Categories and Subject Descriptors

H.4 [**Information Systems**]: Database Management; D.2.8 [**Database applications**]: Data mining

## General Terms

Algorithms, Experimentation, Performance

## Keywords

Data cleaning and Data Integration, Entity Resolution, Product Offers matching, Feature Annotation

## 1. INTRODUCTION

As the electronic information available in the form of online comparison shopping sites, product review sites, e-commerce portals etc. continue to grow at an exponential rate, customers are beginning to rely increasingly on product reviews for purchasing products at such comparison shopping sites. Unfortunately, product offers and product reviews are spread across multiple locations with different data representations, or with different structural forms. This entails the necessity to determine which review corresponds to the offer referenced by the same real-world entity. We refer to this task commonly as Entity Resolution[2], Record Linkage[11, 16, 21], Data deduplication[17], database hardening[7], merge/purge problem[12], and object identification[18].

Entity Resolution in the Product Matching space is much more complicated due to the sheer volume of product offers, and also due to presence of noisy data with each record having different attributes, descriptions along with multiple vendors with different representations (e.g. abbreviations, vendor codes etc.). A universal similarity function for matching product offers is not feasible due to such variations. A lack of global unique identifiers like Universal Product Codes(UPCs) or Manufacturer Part Numbers(MPNs) limits the availability of labeled data to accurately match Product offers.

Also, in many scenarios, product features like manufacturer, category or model number is embedded as a part of record attributes like Product URL, title or category hierarchy, which is again not consistent across categories. Exploiting subtle variations in these attributes, through a feature extraction and enrichment mechanism can lead to effective matching of offers to the corresponding product. An example depicting such finer variations is shown for two such (distinct) Product offers from an online shopping site in Table 1 and the corresponding co-referent reviews from a review site are shown in Table 2. We note the following observations:

- The tokens "Body with Lens Kit - 18 mm-135 mm" in the product title for the first offer from the product site differentiates it from the second offer. Correspondingly, the first review title differs from the second due to the tokens "with 18-135mm lens" being present. Similar variations in the URL tokens are observed as highlighted. This indicates that presence of alphanumeric tokens in product titles significantly help in reducing false matches between offers.
- Manufacturer/Brand and Category information are embedded uniformly across different attributes like URL tokens, product titles etc. which help in enriching the product information which we will discuss in detail later.

A significant point worth highlighting here is the lack of feature specifications for products which deters a completely

| Product URL | Title | Manufacturer | Category |
|---|---|---|---|
| http://cameras.pricegrabber.com/digital/Canon-EOS-60D-DSLR-Digital-Kit/m804086745.html | EOS 60D 18 Megapixel Digital SLR Camera **Body with Lens Kit - 18 mm-135 mm** - Black (3" LCD - 7.5x Optical Zoom - Optical - 5184 x 3456 Image - 1920 x 1080 Video - MPEG-4 - HDMI - PictBridge) | Canon | Cameras |
| http://cameras.pricegrabber.com/digital/Canon-EOS-60D-DSLR-Digital/m804086744.html | EOS 60D Black SLR Digital Camera - **Body Only** (18 Megapixel - 3" LCD - 5184 x 3456 Image - 1920 x 1080 Video - MPEG-4 - HDMI - PictBridge) | Canon | Cameras |

Table 1: Sample Product records from a product offer Site

| Product URL | Title | Category | Product Review |
|---|---|---|---|
| http://reviews.cnet.com/Canon_EOS_60D_**with_18_135mm_lens**/4505-6501_7-34157253.html?subj=fdba&part=rss&tag=rb_content%3Brb_mtx_Digital+cameras | Canon EOS 60D **(with 18-135mm lens)** | Digital Cameras#Canon digital cameras | Amazing performance! Pros: The AF - it's so fast and reliable, I use it for BIF's! The Metering is rock solid in just about any lighting condition. It's also very responsive in just about any situation. Do your share, and the 60D will not disappoint you! Cons: Can't think of any at the moment ... |
| http://reviews.cnet.com/Canon_EOS_60D_**body_only**/4505-6501_7-34157106.html?subj=fdba&part=rss&tag=MR_Search+Results | Canon EOS 60D **(body only)** | Digital Cameras#Canon digital cameras | Fantastic Rebel upgrade Pros: screen, image quality, burst speed, buffer, battery life, and IQ are great for a Rebel Cons: too bulky for even a SuperRebel; out-resolves my lenses at most apertures |

Table 2: Matching Product reviews from a review site

unsupervised approach in the presence of noisy/missing information. For example, absence of Lens information or display type leads to failure in identifying such features in the record fields, making the task of designing a robust system challenging. Also, any entity resolution system in the product matching space must have a high precision. To perform accurate matching of product offers to reviews, we present an approach that is robust to missing information, variants of the same product, limited training data and evaluate the performance of the same. Our approach neither assumes any category-dependent features nor does it require category-specific training to solve the problem.

The outline of the paper is as follows. In Section 2, we discuss related work. Section 3 presents the architecture of the product matching system. Section 4 describes the datasets used and the methodology for evaluating the system. We show experimental results in Section 5 and demonstrate the effectiveness of the feature annotation process. We conclude with a discussion of the implications of our findings and directions for future work in Section 6.

## 2. RELATED WORK

Entity Resolution has received a great deal of attention in recent times ; a comprehensive treatise for Entity Resolution in Köpcke and Rahm [14] depicts the set of challenges and various approaches proposed, along with a qualitative evaluation for each approach. A significant portion of work in the research community solve the Entity Resolution problem on the assumption that the input records on which deduplication is performed have well-structured schemas [1, 15, 4] and where corresponding attributes are matched in each representation. Existing systems [10, 5] dedupe on data differing from each other in the formats they support, the blocking techniques used, the matching functions used and the mechanisms used to combine and train them once structured data is available.

On the other hand, very little work has been carried out in the Product Reviews matching space which pose a completely different challenge. Thor [19] learns an adaptive similarity function based on the term relevance of textual fields in product offers. Bilenko et al. [3] propose an adaptive online approach that clusters offers from different merchants by learning a composite similarity function, which is different from our problem of matching product reviews with product offers.

The work most related to our problem is matching reviews present in the form of unstructured text, with structured Product offers as a language modeling approach by Dalvi et al. [9] which is different from the Entity Resolution problem we tackle, between two structured sets of records. Recent work [13] also has a similar approach of matching unstructured offer descriptions to structured record feature specifications.

In a nutshell, the significant challenge here is identifying and extracting the relevant features representing a product from different attributes present in each of the record feeds. We describe an entity resolution framework that addresses this problem in the following section.

## 3. COMPONENTS OF THE FRAMEWORK

A high level overview of the product offers matching framework is as shown in Figure 1. The Product matching framework accepts a set of two (or more[1]) feeds as input, in the form of structured records, each having one or more attributes. Each of the incoming feeds are transformed into

[1]the system can be easily extended to multiple feeds, though we deal with only two feeds currently for practical purposes

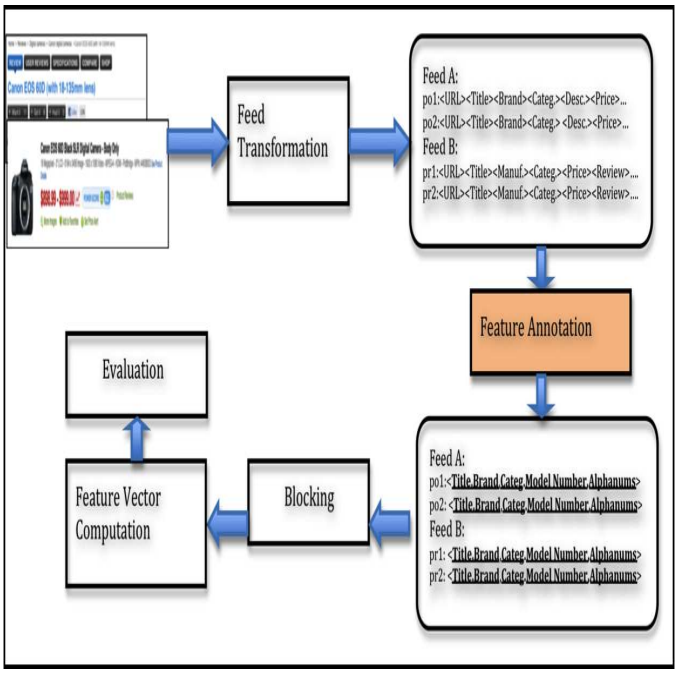


Figure 1: Product Offers - Reviews matching framework

a uniform representation to remove duplicates and aliases of the same record (e.g. different market vendors, different views of user/expert reviews belonging to the same product). We extract key features embedded in the attributes from the transformed record-set that represent the Product Offer/Review. We then construct appropriate similarity measures for each of the equivalent attribute pairs for the records. To avoid computing similarity across all pairs of records, we block on a subset of equivalent fields. For our matching strategy, we take a subset of these records labeled as our "golden" dataset based on the presence of global unique identifiers which are UPCs or MPNs in the product space. We then train our classifier on the labeled feature vectors to generate matching and non-matching pairs. An editorial review of the classified record pairs as matches is the final step in evaluating the status of the matched and nonmatched pairs. We describe each of the modules in the subsequent subsections.

## 3.1 Feature Annotation

Our approach models a dataflow[6] which has a set of records from two record feeds as input, on which **feed-specific pre-processing** is performed. More specifically, we have a set of $n$ records $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ where each record $A_i = \{a_{i1}, a_{i2}, \ldots, a_{ik}\}$ has $k$ attributes from Feed $\mathbb{A}$ and a set of $m$ records $\mathcal{B} = \{B_1, B_2, \ldots, B_m\}$ where each record $B_j = \{b_{j1}, b_{j2}, \ldots, b_{jl}\}$ has $l$ attributes from Feed $\mathbb{B}$. Each feed undergoes conversion to a standardized representation, wherein similar records that refer to the same entity are merged together. This differs from the Record Merging solution in canonicalization[8], where records are simply formed from combining attribute values from different records. From the perspective of Product matching, it is identifying an attribute subset $k' \subseteq k$ for each record in Feed $\mathbb{A}$ also referred to as the Product offer feed and $l' \subseteq l$ for every record in Feed $\mathbb{B}$ referred to as the review feed that uniquely identify the entity.

**Feature Annotation:** Once the attribute subset representing the Product offer and review feeds has been identified, we proceed to annotate the attribute fields. We use the term *product* from now on interchangeably for the product offer and review. Each attribute field value, in its entirety or segmented tokens of the field, is annotated depending on the *most suitable* representation of the product. The *most suitable* representation is the minimal complete set of features that uniquely identify the product. We obtain the minimal set by considering a small domain-specific seed set of attributes $\hat{k} \subseteq k'$ and add all those attributes to the seed set that contain at least one common token with the seed set. Each of the tokens in the newly added attribute set which are common with the seed set are assigned that label of the attribute in the seed set. For example, consider we start off with a seed set {`model number`} for the first product offer in Table 1 . Then we add the attribute set {`Product URL`,`Product Title`} with the common token labeled as `model number`. Thus we identify a set of tokens $t_i \in \mathcal{T}$ and an attribute subset $k'' \subseteq k'$ containing the token set such that

$$\mathcal{T} = \{(l, val(t_i)) | l \in label(k')\}$$

An example of an annotation of a product record from the offer feed and review feed is shown in Table 3 starting from a seed attribute set of {`category`, `brand`, `model number`}. As shown in this example, we find key product features like LCD display size, Optical/Digital zoom information, Flash memory capacity etc. are embedded as a part of the Product URL and title in the offer feed. Also as seen, such product features are absent from all possible attributes in the matched review record. In case of video games or softwares, the only possible key feature present are version numbers of the game/software. Since we aim to compute similarity measures across categories, such inconsistencies need to be captured. We observe that vital information associated with product features are alphanumeric tokens in the attribute subset. We describe a technique to label such tokens in the subsection described below.

### 3.1.1 Alphanumeric Token Labeling

The first step in labeling alphanumeric tokens is obtaining all occurences of numeric/alphanumeric tokens from the attribute subset $k''$ containing the token set $\mathcal{T}$. We add this new set of tokens to the token set, and repeat this process to find alphanumeric token occurences from other attributes in $k' \notin k''$. We then proceed to label each distinct alphanumeric token $a_i$ by identifying a small window of word-tokens (typically 3-4) $l^j_{a_i}$ for each occurence of the token in an attribute field $j$. Since each alphanumeric token can be assigned more than one label from all possible attributes in which they occur, we consider a bag of word-tokens $L_{a_i} = \langle l^j_{a_i} \rangle$ for each alphanumeric token. The complete annotated token set is then given by

$$\mathcal{T}' = \mathcal{T} \cup \mathcal{A} \quad \text{where}$$
$$\mathcal{A} = \{(\langle l^j_{a_i} \rangle, v_{a_i}) \mid v_{a_i} = val(a_i)\}$$

Consider the first sample Product offer in Table 1 and two possible labelings for the same in Tables 4 and 5 below. A closer look suggests that the labeling obtained from Table 4

| Feed | Attribute Name | Attribute Value |
|---|---|---|
| Product Offer Feed | Product URL | http://cameras.pricegrabber.com/$\overbrace{\text{camcorders}}^{\text{category}}$/$\overbrace{\text{Canon}}^{\text{brand}}$-VIXIA-HF-$\overbrace{R11}^{\text{model number}}$-Dual-Flash/m771532551.html |
| | Product Title | VIXIA HF $\overbrace{R11}^{\text{model number}}$ Digital Camcorder (Flash Memory, Memory Card - $\overbrace{16:9}^{\text{alphanum}}$ - $\overbrace{2.70\text{”}}^{\text{alphanum}}$ Active Matrix TFT Color LCD - $\overbrace{20x}^{\text{alphanum}}$ Optical/ $\overbrace{400x}^{\text{alphanum}}$ Digital - $\overbrace{32}^{\text{alphanum}}$ GB Flash Memory) |
| Review feed | Product URL | http://reviews.cnet.com/$\overbrace{\text{digital}-\text{camcorders}}^{\text{category}}$/$\overbrace{\text{canon}}^{\text{brand}}$-vixia-hf-$\overbrace{r11}^{\text{model number}}$/4505-6500_7-33949019.html |
| | Product Title | $\overbrace{\text{Canon}}^{\text{brand}}$ VIXIA HF $\overbrace{R11}^{\text{model number}}$ |
| | Category Desc. | $\overbrace{\text{Digital Camcorders}}^{\text{category}}$#$\overbrace{\text{Canon}}^{\text{brand}}$ digital camcorders |

Table 3: Matched Product Offer-Review records

is a better match for the first Product offer in Table 1 than the labeling obtained from Table 5.

| Feature Label | Feature Value |
|---|---|
| Brand | Canon |
| Category | Cameras |
| EOS | 60D |
| Megapixel | 18 |
| Digital SLR Camera Body | |
| Body with Lens Kit | 18 mm-135 mm |
| LCD | 3” |
| Optical Zoom | 7.5x |
| Image | 5184 x 3456 |
| Video | 1920 x 1080 |

Table 4: Labeling sequence 1

| Feature Label | Feature Value |
|---|---|
| Brand | Canon |
| Category | Cameras |
| EOS | 60D |
| Megapixel | 18 |
| Digital SLR Camera Body | |
| Body with Lens Kit | 18 mm |
| Body with Lens Kit | 135 mm |
| LCD | 3” |
| Optical Zoom | 7.5x |
| Image | 5184 x 3456 |
| Video | 1920 x 1080 |

Table 5: Labeling sequence 2

Since there can exist multiple labelings for an alphanumeric token sequence in the key attributes of the product, the challenge is identifying the optimum label for the sequence. We consider the optimum labeling problem of the token sequence as an unsupervised graph labeling problem using probabilistic models. We adapt Conditional Random Fields (CRFs) since they allow dependencies on sequence tokens with tractable performance. We describe the proposed CRF model as follows.

### 3.1.2 Optimum labeling model

To obtain the optimum labeling for the token sequence $\mathcal{T}'$, we consider $\mathcal{T}'$ to be composed of two kinds of information, the observed information in the form of token values, and the unobserved information in the form of token labels. Each alphanumeric token is labeled with `alphanumeric token label` or `alphanumeric token value`. Since each <alphanumeric label,alphanumeric value> pair depend on the observation of the sequences and the labels of the tokens & vice-versa, the undirected graph modeled by a CRF best captures this inter-dependency. The conditional probability of latent variables given the observed variables is given by:

$$\Pr(y|x) = \frac{1}{Z} \prod_{C(x,y)} \Phi(C(x,y)) \quad (1)$$

where $x$ and $y$ are the set of observed variables and the set of unobserved variables respectively, and $\Phi(C(x,y))$ is the clique potential for the clique $C(x,y)$. $Z$ is called as the partition function defined as:

$$Z = \sum_{y} \prod_{C(x,y)} \Phi(C(x,y)) \quad (2)$$

We can write Equation 1 as

$$\Pr(y|x) = \frac{1}{Z} \exp \sum_{i} \gamma_i f_i(x,y) \quad (3)$$

where

$$\Phi(C(x,y)) = \exp \sum_i \gamma_i f_i(x,y)$$

where $\gamma_i$ $f_i(x,y)$ are the $i$th binary feature and corresponding weight. The optimum labeling of the unobserved variables can be found using the standard message passing algorithm.

### 3.1.3 Parameter learning

Parameter learning involves learning the optimum weights $\gamma_i^*$ which is done by solving the maximum likelihood function

$$\mathcal{L}(\gamma_i) = \sum_{j=1}^{T} \sum_i \gamma_i f_i(x_j, y_j) - \log(Z) \quad (4)$$

where $T$ is the training set. As mentioned earlier, we undertake an unsupervised approach for parameter learning based on the EM algorithm, where the E-step estimates the probability of labeling the hidden parameters, and the M-step maximizes the current weight update using an iterative method like conjugate descent or LBFGS. The algorithm is described in Algorithm 1. However, since this formulation is non-convex and can get stuck in a locally optimal solution, we incorporate suitable priors for choosing the initial parameter values.

**Algorithm 1** EM based LBFGS optimization for CRF Learning

**Input:** $T$ - A set of Training examples consisting of co-referent and non co-referent pairs.
$K$ - Number of iterations.
$\gamma_i^0$ - Initial parameter set.
**Output:** $\gamma_i^K$ - Final Parameter set.
$\gamma_i^* \leftarrow \gamma_i^0$.
**repeat**
  **E-step:**
  **for** $j = 1, 2, \ldots, T$ **do**
    $\Pr(y'|x_j) = \frac{1}{Z} \exp \sum_i \gamma_i^* f_i(x_j, y')$;
  **end for**
  **M-step:**
  **for** $k = 0, 1, \ldots, K-1$ **do**
    **for** $j = 1, 2 \ldots, T$ **do**
      $\hat{y}_j^k = \nabla f(x_j + \alpha_j p_j) - \nabla f(x_j)$ ;
      $\gamma_i^{k+1} \leftarrow \gamma_i^k + \sum_{y'} f_i(x_j, y') \Pr(y'|x_j) - f_i(x_j, \hat{y}_j^k)$;
      $k \leftarrow k+1$
    **end for**
  **end for**
  $\gamma_i^* \leftarrow \gamma_i^K$
**until** Convergence is achieved

## 3.2 Feature Vector Construction

The annotated records from both feeds are matched using a *feature-specific* similarity function defined between equivalent features of each record of a feed. Consider a feature set $\mathcal{F} \subseteq k, l$ constructed for each of the product offers. We then define a comparison feature vector

$$F^*\langle R_1, R_2 \rangle = \langle f_i(R_1, R_2) \rangle \quad (5)$$

where each $f_i(R_1, R_2)$ is an appropriate similarity function defined for the corresponding feature. An important consideration while defining similarity functions is differentiating the scoring metric for a nominal feature e.g. product model numbers and alphanumeric tokens from that of a generic field e.g. title. Invariably, mismatches for nominal attributes receive binary scores as against generic fields.

## 3.3 Training Phase

Entity Resolution in the Product offers space is done by classifying every record pair into matching ($\mathcal{M}$) pairs or non-matching($\mathcal{N}$) pairs. Typically, a comparison feature vector is computed between each of the $n$ records from Feed $\mathbb{A}$ and $m$ records from Feed $\mathbb{B}$. A binary classifier is then trained using a "golden" labeled dataset having each $(R_1, R_2) \in \mathcal{M}$ or $(R_1, R_2) \in \mathcal{N}$. The number of non-matching labeled pairs tend to overwhelm the number of matching labeled pairs due to limited presence of "golden" indicators like UPCs or MPNs. To avoid the bias towards the size of the labeled data, we employ support vector machines(SVMs)[20] as our binary classifier which provide an intuitive notion of similarity wherein the matching and non-matching pairs are separated by a maximal-margin hyperplane.
We train an SVM on the computed similarity feature vectors for the labeled set as shown in Algorithm 2. The SVM classifier learns weights from the training feature vectors(FVs) by using a Kernel function to map them to a high dimensional space. The SVM classifier function output is a decision function depending upon the Kernel function employed and the Lagrangian dual variables $\vec{\alpha}$.

**Algorithm 2** Training algorithm

**Input:** $\mathcal{M}_T : (R_{ip}, R_{jp})_{p=1}^{P}$ - A labeled set of $P$ matched record pairs.
$\mathcal{N}_T$ : $(R_{iq}, R_{jq})_{q=1}^{Q}$ - A labeled set of $Q$ non-matched record pairs.
**Output:** Decision function $f(R_x, R_y) = \text{sgn}(\sum_{i=1}^{p+q} y_i \alpha_i K(F_i, [R_x, R_y]) + b)$
**for** each record pair $\in \mathcal{M}_T$ and record pair $\in \mathcal{N}_T$ **do**
  ConstructFeatureVectors $F\langle \mathcal{M}_T \rangle = F\langle R_{ip}, R_{jp} \rangle_{p=1}^{P}\rangle$;
  ConstructFeatureVectors $F\langle \mathcal{N}_T \rangle = F\langle R_{iq}, R_{jq} \rangle_{q=1}^{Q}\rangle$;
**end for**
Train SVM on $F\langle \mathcal{M}_T \rangle$ and $F\langle \mathcal{N}_T \rangle$ to obtain the SVM model $f(x) = \sum_{i=1}^{p+q} \alpha_i y_i K(F_i, x) + b$
**return** $\text{sgn}(\sum_{i=1}^{p+q} y_i \alpha_i K(F_i, [R_x, R_y]) + b)$ given a Record pair FV $F\langle R_x, R_y \rangle$

Computing the similarity scores for all record pairs ($O(mn)$) is highly inefficient, since typically in the product space, a large number of them are complete non-matches. To avoid computing this large number of similarity pairs, we divide records into disjoint subsets by **blocking** on a set of attributes. We discuss in detail about the blocking strategy adopted in the following section.

# 4. EVALUATION

The objective of the Entity Resolution framework is to return the best set of matched product-offer pairs with as few incorrect matches as possible (high precision) with as many correct matches covered as possible (good recall). Precision is a critical requirement in an Entity Resolution system, since we do not want the wrong review associated with a product. As we will see, different categories of products

show different behaviors due to noisy information associated with them. Hence, to improve system efficiency, we boost Precision by considering variants of the same:

- **Precision at Top-K :** For each record from feed A, we obtain the Top-K matching records from feed B ordered by the weighted sum of the comparison feature scores and check for the corresponding presence of the correctly matched record in the top-K.

- **Precision at Top-K with review :** Once the top-K for a record is obtained, we do an editorial verification of those records not present in the top-K for a random selection of records for various categories.

## 4.1 Datasets

Product offers extracted from an online comparison shopping site and reviews from a well known review site form our record feeds for the Entity Resolution task. A periodic crawl over a two-month duration yielded around 6M product offer pages and 200K review pages on which standard feed-specific transformation mentioned in Section 3 was done. This ultimately resulted in a labeled dataset of 14,000 products matched on the basis of Manufacturer Part Numbers(MPNs) / Universal Product Codes(UPCs) / Stock Keeping Units (SKUs) with the corresponding product reviews. The labeled dataset consists of 6 top-level categories viz. computers (desktops, laptops etc.), electronics (flat panel TVs, cellphones, MP3 players), cameras (e.g. digicams, camcorders), video games(XBOX games, playstation games etc.), appliances (microwaves, ranges) and software (antivirus, games etc.) containing 36 finer-level categories with at least 50 products per category. Though we do not perform any category-specific training set or any category-specific features, we ensure that every top-level category is associated with at least 10% of the total labeled records for training the classifier.

In addition to the co-referent records, we also inject a sizable number of non co-referent pairs that closely resemble each other with only variations in the Model number or in video game categories along with version numbers. We select 12000 such closely mismatched instances belonging to cameras, video games, electronics and softwares each containing an approximate equal amount of non co-referent product-review pairs.

We perform a two fold cross validation by randomly splitting the dataset and assigning each fold as train and test datasets. We learn the model as described in Section 3.3 on the training set and evaluate on the test set for each run. We use standard evaluation metrics to measure performance during each run defined by

$$\text{Precision} = \frac{\text{Number of correctly identified Co-referent records}}{\text{Total number of identified Co-referent records}}$$

$$\text{Recall} = \frac{\text{Number of correctly identified Co-referent records}}{\text{Total number of Co-referent records}}$$

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

We report the average Precision-Recall values over 5 trials for each of the two fold cross validation run in our evaluation results described below.

## 4.2 Methodology

An effective way of evaluating our approach is comparing it with a **baseline** measure. We employ the TFIDF cosine similarity measure, which we address as TFIDF-Baseline. We add all relevant tokens from the title, brand and category fields from the product offer-review pair and associate each token with a score defined by tf(token)*log(1+idf(token)), where tf(token) is the token frequency and idf(token) is the token inverse document frequency. TFIDF-Baseline acts as an effective baseline, due to its higher weightage for infrequent tokens like Model Number and alphanumeric tokens e.g. Memory capacity. Thus, we construct the Baseline-TFIDF feature vector as the tfidf similarity score for the ⟨title,brand,category⟩ triple for every product offer-review record pair.

To reduce the number of comparisons across product offers to compute the feature vectors, we perform blocking across an attribute set. We use a number of blocking functions to find out the best blocking measure for the attribute subset as mentioned below.

- **Common tokens:** For every attribute subset, if any of the attributes contain a common token, then the record pair is considered, else discarded.

- **Edit distance:** For every attribute subset, consider those record pairs where the edit distance between equivalent attributes are $\geq \gamma$ where $\gamma = [0.2 \cdots 1.0]$ is a sliding similarity measure increasing in steps of 0.2.

- **Token based TFIDF:** For every attribute subset, consider those record pairs where the TFIDF cosine similarity distance between equivalent attribute tokens are $\geq \gamma$ where $\gamma = [0.2 \cdots 1.0]$ in steps of 0.2.

- **Q-gram based TFIDF:** For every attribute subset, consider those record pairs where the TFIDF cosine similarity distance between equivalent attribute Q-grams are $\geq \gamma$ where $\gamma = [0.2 \cdots 1.0]$ in steps of 0.2 ; We consider Q=3 for our evaluation.

We choose appropriate similarity measures for each of the annotated features. The `Title` score is computed using a token-level TFIDF cosine similarity measure. We consider the product `Brand` as a nominal attribute, since the feed records are not user-generated content, and hence variations in the `Brand` field are minimal. `Category` scores are computed using a level two distance metric. The product `Model Number` is again a categorical attribute and hence receives a binary score. In addition, we assign binary scores to each of the `Alphanumeric Tokens` identified from the offer and reviews.

One significant point worth mentioning is, we penalize mismatches more than missing information, since the structured representation of Product reviews are often noisy. Hence, mismatches receive a score of negative weights (typically -1) and missing values are given a score of 0.

# 5. RESULTS AND DISCUSSIONS

Figure 2 shows the precision v/s recall plot averaged over all trial runs as mentioned earlier in Section 4.1 with a blowup of category-wise performance. The overall precision across all offer-review pairs is 99.203% and the average recall obtained is 89.3% across all offers over all categories. Almost

all categories perform with a precision of at least 96%, where cameras account for a minor drop in precision when we encounter an offer *D60 10.2 Megapixel Digital SLR Camera Body with Lens kit - 18 mm-55 mm (2.5" LCD - 3x Optical Zoom - 3872 x 2592 Image)* with the corresponding review of the type *Nikon D60 Black Gold Special Edition (with 18mm-35mm lens)* where the noted difference in the alphanumeric score of the lens range is not suitably captured by the classifier. However, the recall varies from 67% for LCD panels to 98% for significant categories like Mice/trackballs. The noticeable recall reduction for LCD panels is due to absence of alphanumeric tokens present from any of the attributes in the Review records. Since Mice information usually contain model numbers and alphanumeric tokens in at least one of the attribute fields, the classifier accounts for the high recall.

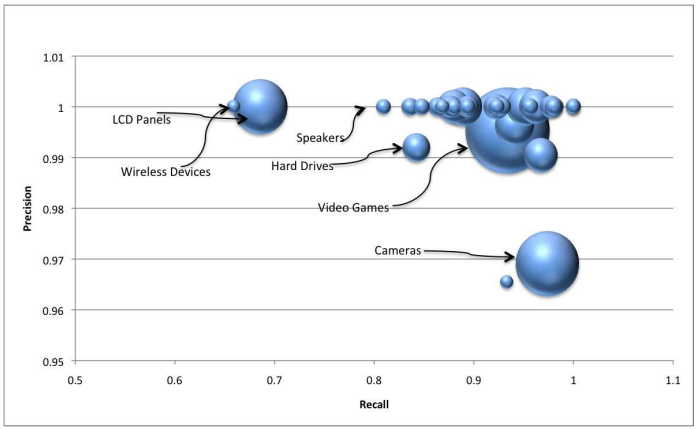


Figure 2: Performance of Feature Annotation

We also perform 5 runs of 2-fold cross validation for the Baseline measure. The overall precision is found to be 72.577% and the recall is 95.3%. Figure 3 shows the F-measure performance of Baseline-TFIDF against the F-measure of the Feature Annotation method. We note several interesting observations of the Baseline measures. Baseline-TFIDF performs well on categories where Model numbers or key product features do not perform a significant role. An interesting observation is the effect of injected mismatches on the Precision drop. A large amount of injected mismatches in Cameras and Video games differ only on the basis of model numbers in the title with most of the other information being the same (e.g. Brand and Display information), or games belonging to different categories but with the same version number. Since the classifier learns the title TFIDF score irrespective of importance of a model number from a LCD size, it maximizes the matching of such records resulting in a large number of spurious matches leasing to low precision. However, since a significant amount of records in Video games do not have version numbers, the precision is not too bad for the same.

## 5.1 Performance comparison of different measures

We attempt to understand the effect of different thresholds of blocking on the robustness of our feature annotation framework and compare it with Baseline-TFIDF. Figure 4 shows the effect when Token-based blocking on category attributes is employed for various thresholds. As seen, the Precision of the Baseline-TFIDF increases marginally to 75% while the recall drops, and then again tapers down to 72.5%. Feature Annotation is much more robust, with near to 99% precision with decreasing recall. Finally, we also see that Top-K (K=3 here for our experimental settings) Feature Annotation with review results in negligible increase in precision. Hence, for all practical purposes Feature Annotation measures are just as effective as Top-K Feature Annotation with review, and hence we report only Feature Annotation scores wherever applicable.

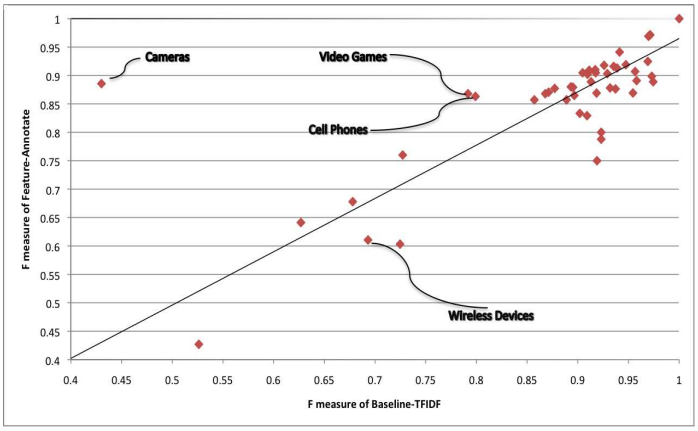


Figure 3: Feature Annotation F measure vs Baseline-TFIDF F measure

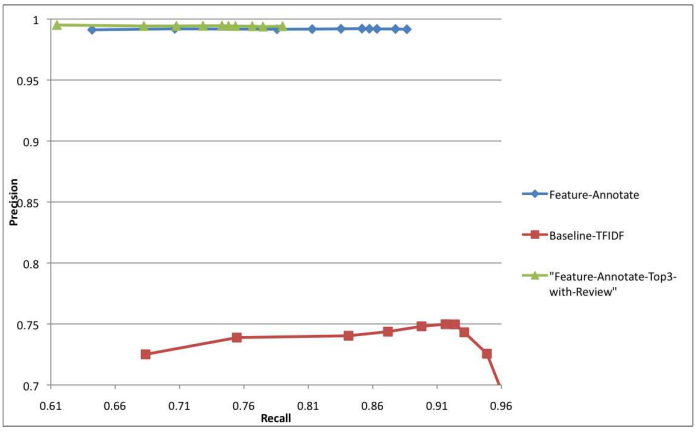


Figure 4: Performance Comparison of different measure at various blocking thresholds

## 5.2 Effect of each feature on Feature Annotation Precision

We highlight the impact of each feature on the overall precision in Figure 5. As seen, when only title scores are considered, precision increases linearly with recall. A possible reason for this is due to the presence of uniform representation of titles in the co-referent pairs more than the non co-referent pairs, which makes the classifier learn accurately as more and more co-referent pairs are added. Again, we see this behaviour being depicted when Model numbers are incorporated along with alphanums, since model num-

bers show the correctness of a matching pair probably better than the alphanum tokens themselves.

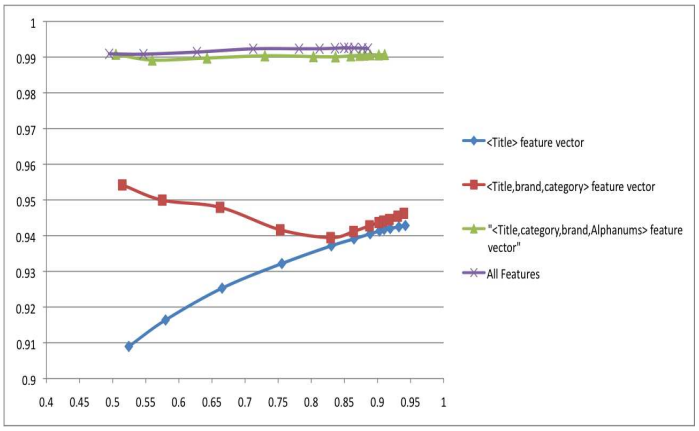


Figure 5: Impact of features on Feature Annotation precision

## 6. CONCLUSIONS AND FUTURE WORK

In summary, the main contributions of the paper are as follows:

- We motivate matching product offers with reviews highlighting the principal issues involved.
- We introduce a novel *feature annotation* approach for extracting key features embedded in product feed attributes in an unsupervised category independent manner.
- We employ different evaluation metrics on a real world product offer-product review dataset for the Entity Resolution task and demonstrate extensively the performance of our approach.

We believe that the framework can be adapted to any system wherein structured records like product discount deals needs to be matched with offers from shopping sites or restaurant locations with restaurant reviews. Another interesting aspect to be considered is the scalability of the approach. In that respect, we would like to take this problem setting forward is an online scenario similar to [13] for streaming product offers and reviews.

## 7. REFERENCES

[1] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom. Swoosh: a generic approach to entity resolution. *VLDB J.*, 18(1):255–276, 2009.

[2] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *DMKD'04*, pages 11–18, 2004.

[3] M. Bilenko, S. Basu, and M. Sahami. Adaptive product normalization: Using online learning for record linkage in comparison shopping. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 58–65, Washington, DC, USA, 2005. IEEE Computer Society.

[4] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 39–48, New York, NY, USA, 2003. ACM.

[5] P. Christen. Automatic training example selection for scalable unsupervised record linkage. In *Proceedings of the 12th Pacific-Asia conference on Advances in knowledge discovery and data mining*, PAKDD'08, pages 511–518, Berlin, Heidelberg, 2008. Springer-Verlag.

[6] P. Christen. Development and user experiences of an open source data cleaning, deduplication and record linkage system. *SIGKDD Explor. Newsl.*, 11:39–48, November 2009.

[7] W. W. Cohen, H. Kautz, and D. McAllester. Hardening soft information sources. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 255–259, New York, NY, USA, 2000. ACM.

[8] A. Culotta, M. Wick, R. Hall, M. Marzilli, and A. McCallum. Canonicalization of database records using adaptive similarity measures. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 201–209, New York, NY, USA, 2007. ACM.

[9] N. Dalvi, R. Kumar, B. Pang, and A. Tomkins. Matching reviews to objects using a language model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 609–618, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[10] M. Elfeky, V. Verykios, and A. Elmagarmid. Tailor: A record linkage tool box. In *Proceedings of the 18th International Conference on Data Engineering*, ICDE '02, pages 17–, Washington, DC, USA, 2002. IEEE Computer Society.

[11] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210, 1969.

[12] M. A. Hernández and S. J. Stolfo. The merge/purge problem for large databases. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, SIGMOD '95, pages 127–138, New York, NY, USA, 1995. ACM.

[13] A. Kannan, I. Givoni, R. Agrawal, and A. Fuxman. Matching unstructured product offers to structured product specifications. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, New York, NY, USA, 2011. ACM.

[14] H. Köpcke and E. Rahm. Frameworks for entity matching: A comparison. *Data Knowl. Eng.*, 69:197–210, February 2010.

[15] S. Minton, C. Nanjo, C. A. Knoblock, M. Michalowski, and M. Michelson. A heterogeneous field matching method for record linkage. In *ICDM'05*, pages 314–321, 2005.

[16] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records:

Computers can be used to extract "follow-up" statistics of families from files of routine records. *Science*, 130:954–959, 1959.

[17] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *KDD*, pages 269–278, 2002.

[18] S. Tejada, C. A. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 350–359, New York, NY, USA, 2002. ACM.

[19] A. Thor. Toward an adaptive string similarity measure for matching product offers. In *GI Jahrestagung (1)*, pages 702–710, 2010.

[20] V. Vapnik. *Statistical learning theory*. Wiley, 1998.

[21] W. E. Winkler, W. E. Winkler, and N. P. Overview of record linkage and current research directions. Technical report, Bureau of the Census, 2006.