Int J Multimed Info Retr
DOI 10.1007/s13735-017-0124-0

REGULAR PAPER

ACSIR: ANOVA Cosine Similarity Image Recommendation in vertical search

D. Sejal[1,2] · T. Ganeshsingh[1] · K. R. Venugopal[1] · S. S. Iyengar[3] · L. M. Patnaik[4]

Received: 10 November 2016 / Revised: 21 January 2017 / Accepted: 17 March 2017

Abstract In today's world, online shopping is very attractive and grown exponentially due to revolution in digitization. It is a crucial demand to provide recommendation for all the search engine to identify users' need. In this paper, we have proposed a ANOVA Cosine Similarity Image Recommendation (ACSIR) framework for vertical image search where text and visual features are integrated to fill the semantic gap. Visual synonyms of each term are computed using ANOVA p value by considering image visual features on text-based search. Expanded queries are generated for user input query, and text-based search is performed to get the initial result set. Pair-wise image cosine similarity is computed for recommendation of images. Experiments are conducted on product images crawled from domain-specific site. Experiment results show that the ACSIR outperforms iLike method by providing more relevant products to the user input query.

Keywords Content-based image retrieval (CBIR) · Image recommendation · Vertical search

1 Introduction

The number of Internet users is increasing rapidly an account of revolution in Internet and digitization. The availability of multimedia is also increasing rapidly, and hence, it demands fast, efficient and reliable methods of information retrieval. The most popular tools are Search Engines which provide an interface to accept input queries from users and retrieve relevant information.

The image search engines are broadly classified into two categories: (i) horizontal search engine that provides links to relevant content and (ii) vertical search engine is used to perform domain-specific search and provides actual content rather than links. Text-based and content-based search approaches are widely used for these types of search engines to retrieve images. The text-based image search is completely dependent on the occurrence of input query terms either in metadata or in surrounding text of images. This approach is widely used because of its lower computation cost and faster response time, whereas it fails to retrieve the images which are relevant but do not have the term in the surrounding text. In the content-based image search, visual features of query-image are compared with that of images present in database. It performs content-based matching; hence, it retrieves images that are relevant irrespective of the query term. This method requires higher time complexity and has slower response time.

Sometimes, search engines fail to retrieve information as per user wish because of various reasons: (i) improper input query, (ii) lack of users' understanding about the input search query, (iii) wrongly tagged images present in database. Hence, users are unable to achieve the desired output. This gap between users' search intention and understanding of object is called as semantic gap and is common in most of the image search engines.

Motivation Various hybrid approaches have been proposed for reduction of semantic gap [1–5]. These approaches use two-step search and rank algorithms. First, images are retrieved with text-based search. Secondly, images are ranked by computing similarity between images using visual fea-

✉ D. Sejal
sej_nim@yahoo.co.in

1 Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bangalore 560001, India

2 T. John Institute of Technology, Bangalore, India

3 Florida International University, Miami, Florida, USA

4 National Institute of Advanced Studies, Bangalore, India

Published online: 31 March 2017

tures. Here, text and visual features are used independently and are not semantically correlated. But, these hybrid models are capable of reducing semantic gap. As text-based image search is very popular, if visual meaning of terms is integrated with actual representation of image, retrieval performance can be improved.

Contribution In this paper, ANOVA Cosine Similarity Image Recommendation (ACSIR) framework for vertical image search is proposed. This framework does not consider user feedback session as in IR_URFS_VF [6] to recommend images. Various product images with its description are crawled from domain-specific site. Visual features of all the images are computed, normalized and stored in the database offline. For each term in the product description, weight is computed using ANOVA p value by considering visual features. This p value is used to identify semantically similar terms. For each pair of terms, cosine textual similarity measure is used to calculate pair-wise term similarity. For, given user input query, expanded queries are generated by considering term similarity and text-based search is performed to get initial results. Pair-wise image cosine similarity is computed on these results; images are ranked based on similarity score and used to recommend the images.

Organization This paper is organized as follows: Sect. 2 presents survey on various image retrieval methods. The proposed ACSIR framework and algorithm are presented in Sect. 3. Section 4 discusses data collection, experiment setup and performance evaluation. Conclusions are given in Sect. 5.

2 Related work

In this section, different image search techniques are reviewed and can be categorized as (i) content-based image retrieval (CBIR), (ii) annotation-based image retrieval (ABIR), (iii) text + visual (hybrid) method for image search and (iv) image search with reduced semantic gap.

2.1 Content-based image retrieval (CBIR)

Initially, images are annotated manually and text-based image search is performed on annotated images. The process of manual annotation for large volume of images involves intense labor, probability of wrongly annotating images and limitation of keywords to describe images. Hence, image search performance is degraded. CBIR techniques are developed to use image visual features for image retrieval. Images are indexed by their visual features such as color, shape or texture. Many CBIR approaches have been proposed by many researchers on different datasets [7–13]. Methods proposed in [14,15] can be used to compute image similarity for retrieval. The CBIR methods are effective but requires more computation time, and for low-level features, the performance is uncertain.

2.2 Annotation-based image retrieval (ABIR)

In ABIR, images are tagged or metadata in terms of keywords are assigned automatically. It is a multiclass image classification problem, and new images are annotated automatically using machine learning techniques. Since automated annotations are used for retrieval, user input query terms are consistent with annotations. Many ABIR methods are developed for automatic image annotations [16–23]. The key challenge in ABIR is to frame classifier (specific and related). Further, the method is only text-based search, where image visual features are not taken into consideration.

2.3 Text + visual (hybrid) method for image search

Various image search techniques by combining text and visual features are developed to overcome the problems of CBIR and ABIR. Cui et al. [5] have proposed a hybrid method to re-rank Google search results. An intention category model is used to integrate visual features adaptive to the input image. Image features are combined in each category by applying similarity measure to re-rank images. Experiments are performed on Google and Microsoft Live Image Search. Generic classifier [24] is developed to classify images. In this method, textual features are combined by analyzing existence of query terms in image metadata and web pages. Later, image histogram is used for visual representation.

Hybrid image search, clustering algorithms are proposed by combining text and visual features [2,4,25]. In these methods, first the images are categorized using image visual features to form respective clusters. Images from clusters are loaded based on query terms. Image visual features are used to re-rank the images in cluster. It involves cluster maintenance, as single image may belong to more than one cluster.

Two-step hybrid image re-ranking methods are proposed in [26,27]. In these algorithms, user's intention is captured by single click on image results obtained by text-based search. Expanded queries are formed by selected image clicked by user. These expanded queries are used to re-rank the images. In these hybrid methods, text and visual features are not correlated and the visual meaning of the term is not integrated.

2.4 Image search with reduced semantic gap

In image search with reduced semantic gap methods, text and visual features of images are integrated. Here, the semantic meaning is combined with visual representation of term. [3] proposed a framework by fusing text and visual features. Image-tag bipartite graph and image similarity graph based on visual features are combined. Texture features are extracted from co-occurrence matrix, and its results are compared with k-mean clusters to reduce the semantic gap [1]. Semantic gap is bridged by acquiring the visual meaning of

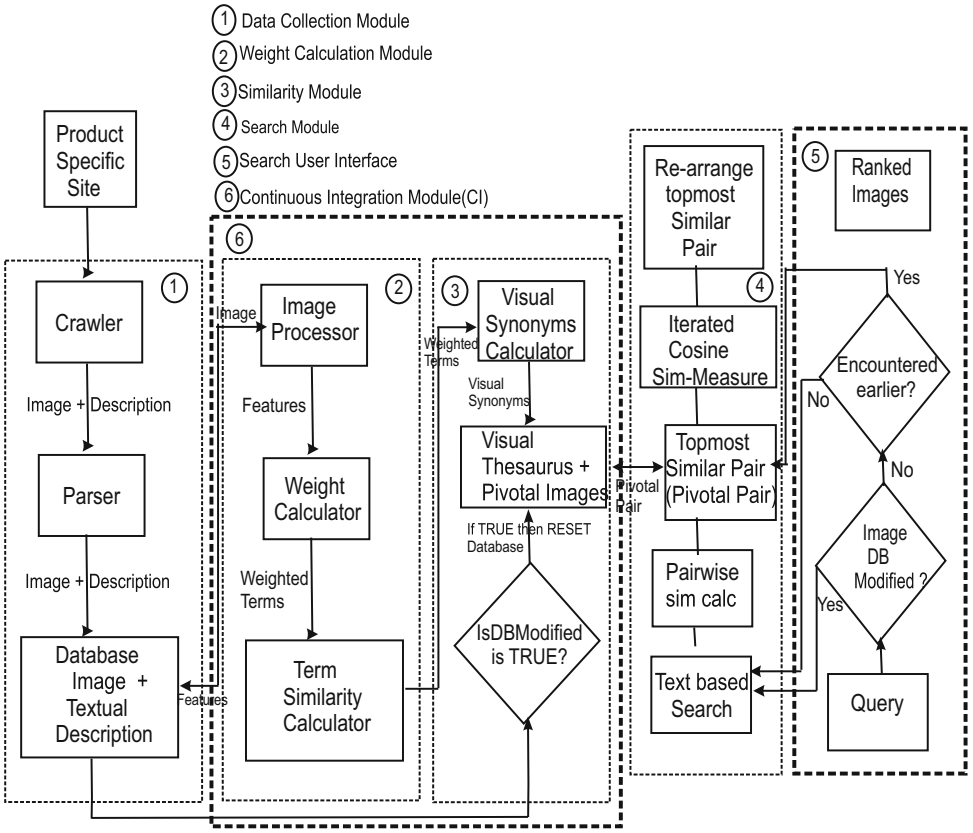

Fig. 1 ACSIR framework

each term [28,29]. The visual meaning is extracted by computing p value using Kolmogorov–Smirnov (K–S) test on image visual features. Expanded queries are formed using computed visual synonyms.

3 ACSIR framework and algorithm

3.1 Problem definition

On a domain-specific website, for given a user input query q, the objective is to recommend products.

3.2 Assumptions

It is assumed that user is online while entering input query.

3.3 ACSIR framework

The ACSIR framework retrieves and recommends images for given user input query in vertical online image search which is shown in Fig. 1. This framework has six different modules: (i) data collection module, (ii) weight calculation module, (iii) similarity module, (iv) search module, (v) search user interface and (vi) continuous integration module. Each module is explained below.

3.3.1 Data collection module

In this module, a pair of customized crawler–parser is used to fetch product-specific pages from an online retailer website.

(a) *Crawler* Crawler is a utility that fetch webpages and extracts images along with metadata from a particular website. In this framework, retailer-specific webpages are used to crawl various product images and their metadata. This metadata contains associated annotations such as product name, price, category that can be used to retrieve images.

(b) *Parser* Non-search-specific product details and stopwords are removed from metadata by using a customized text parser. It extracts images and search-specific product labels by rule-based text parsing. It generates product-specific terms image pairs with higher degree of relevance. These parsed term image pairs are then added into database.

3.3.2 Weight calculation module

Text-based image search is completely dependent on the existence of input query terms in surrounding text. This approach is widely used because of its lower computation cost and faster response time, whereas it fails to retrieve the images which are relevant but do not have the term in the surrounding text. An efficient search engine should retrieve such images. Image visual features can be used to retrieve visually similar images where the query terms are not present in the surround-

ing text. An image processing module is used to retrieve these images that extracts visual features, stores them in database and is computed offline.

(a) *Image processor* Here, variety of widely used image features is extracted and normalized for crawled images.

Features collection Gray-level co-occurrence matrix is used to extract texture features. Thirteen Haralick features [30] are extracted from this matrix. Coarseness of image and direction are identified by three dimensions of Tamura features [31]. Gabor filters [32] (eight directions and five scales) are used to get 40 texture feature vector. Images are divided into (3×3) blocks of YCbCr, and first three color moments are applied to gain 18 color features.

Normalization Various image acquisition methods are used to capture images; as a result few features may dominate. Hence, image comparison accuracy is affected. For example, higher light intensity may affect color features. Low light and other weather factor may cause deformation in image areas resulting inconsistent image features. Hence, feature values need to be optimized to avoid deviation in visual meaning of images. The *min–max* method is used to normalize feature in (0, 1) range by Eq. 1

$$\mathbf{feature(X)_{norm}} = \frac{feature(X) - feature(min(X))}{feature(max(X)) - feature(min(X))} \tag{1}$$

Here, $feature(X)_{norm}$ is the normalized feature value of Xth feature. $Feature(min(X))$ and $feature(max(X))$ are the minimum and maximum values for Xth feature in image pool.

(b)*Weight calculator* Text-based image retrieval method fails to retrieve images which do not contain query terms in image description but are visually similar. Hence, term similarity is computed to find visually similar words to increase coverage of images and efficiency of search. These words are used to construct extended queries for given input query. Term similarity between two terms is computed by applying cosine term similarity to statistical p value computed by ANOVA method.

ANOVA p value retrieval Consider universal image pool U containing all images present in dataset. Unique terms are extracted from image description present in U by removing stop-words and irrelevant terms to form term dictionary T. For each term T_i in T, images are retrieved to form positive sample set S_p by performing text-based search. Here, i = *1,2...n*, where n is the size of T. The inverse of S_p is calculated to form a negative sample set $S'_p \cdot S'_p = U - S_p$. The method to calculate p value of each term is shown in Function 1.

Function 1: ANOVA p-value Calculator

Function: ANOVA p-valueCalculator
Data: For term T_i in T with size n, consider S_p and S'_p as positive and negative sample set with size M and N, respectively. Generate p-value vector of size F_c for each term T_i.

begin
  for i = *1 to* n **do**
    Compute S_p and S'_p for term $T(i)$
    for k = *1 to* F_c **do**
      for j = *1 to* M **do**
        $\mathbf{feature_{pos-mean}(X_{jk})} = \frac{\sum_{j=1}^{M} feature(X_{jk})}{M}$
    for k = *1 to* F_c **do**
      for j = *1 to* N **do**
        $\mathbf{feature_{neg-mean}(X_{jk})} = \frac{\sum_{j=1}^{N} feature(X_{jk})}{N}$
    Compute p-value with
    $Calculate\text{-}pvalue(feature_{pos-mean}, feature_{neg-mean})$

In Function 1, $\mathbf{feature_{pos-mean}(X_{jk})}$ and $\mathbf{feature_{neg-mean}(X_{jk})}$ are the mean vector of S_p and S'_p, respectively. For every extracted feature set for each term Function 2 is invoked to compute p value for each feature.

In Function 2, SST_{pos} and SST_{neg} are the total sum of squares of mean features of positive and negative sample, respectively. SSE is the error sum of squares. The function *cumulativeprobability(F)* is the cumulative distribution function of regularized beta function.

Mapping p values: The computed p values using Function 1 are potentially small in range of 10^{-15} to 10^{-55} which yields difficulty in comparison. Hence, p values are inverted and normalized using *min–max* function as in Eq. 1. These inverted-normalized p values are used as weight vector W of the query terms such that the smaller the p value, the higher the weight.

(c) *Term similarity calculator* Here, computed p values are used to identify semantically similar terms. For each pair of terms (T_i, T_j) where i ≠ j, cosine textual similarity measure is applied to calculate pair-wise term similarity using Eq. 2.

$$Term_{similarity}(T_i, T_j) = \frac{\sum_{k=1}^{F_c}(FPM_{(X_k,T_i)} * W_{(X_k,T_i)}) * (FPM_{(X_k,T_j)} * W_{(X_k,T_j)})}{\sqrt{\sum_{k=1}^{F_c}(FPM_{(X_k,T_i)} * W_{(X_k,T_i)})^2} * \sqrt{\sum_{k=1}^{F_c}(FPM_{(X_k,T_j)} * W_{(X_k,T_j)})^2}} \tag{2}$$

Function 2: Calculate p-value

Function: Calculate p-value
Data: Consider $feature_{pos-mean}(X)$, $feature_{neg-mean}(X)$ vectors. Generates p value for each feature.

begin

Let $vector_{size}$ = number of features in $feature_{pos-mean}$ = number of features in $feature_{neg-mean}$

for i = *1 to* $vector_{size}$ **do**

Compute

$$P_{pos} = \sum_{j=1}^{vector_{size}} feature_{pos-mean}(X_j)^2$$

$$Q_{neg} = \sum_{j=1}^{vector_{size}} feature_{neg-mean}(X_j)^2$$

$$A_{pos} = [\sum_{j=1}^{vector_{size}} feature_{pos-mean}(X_j)]^2$$

$$B_{neg} = [\sum_{j=1}^{vector_{size}} feature_{neg-mean}(X_j)]^2$$

$$SST_{pos} = P_{pos} - A_{pos}$$

$$SST_{neg} = Q_{neg} - B_{neg}$$

$$SST = SST_{pos} + SST_{neg}$$

$$SSA_{pos} = \frac{(P_{pos})^2}{M} - \frac{(A_{pos})^2}{(M * Fc)}$$

$$SSA_{neg} = \frac{(Q_{neg})^2}{N} - \frac{(B_{neg})^2}{(N * Fc)}$$

$$SSA = SSA_{pos} + SSA_{neg}$$

$$SSE = SST - SSA$$

$$F = \frac{(SSA/2)}{(SSE/Fc)}$$

p-value = $1 - CumulativeProbability(F)$

Here, $FPM_{(X_k,T_i)}$ is the feature mean vector of positive sample set S_p of term T_i. $W_{(X_k,T_j)}$ is the inverted-normalized weight vector of term T_i.

3.3.3 Similarity module

Pair-wise term semantic similarity score is computed with term similarity calculator. This similarity score is computed by using the visual features of images retrieved with text-based search. Hence, terms are semantically and visually similar. Term dictionary is constructed based on similarity score.

(a) *Visual synonym calculator* For each term T_i, maximum similarity score max_{sim} for pairs (T_i,T_j) is selected. This max_{sim} is used to set a selection threshold $Th_{selection}$ calculated using Eq. 3.

$$Th_{selection} = T_h * max(Term_{similarity}(T_i, T_j)) \quad (3)$$

Here, T_h is a range threshold set for selection of similarity score. All the term pairs with similarity score exceeding $Th_{selection}$ are selected as visually similar semantic synonyms of term T_i. This process is repeated for each term T_i in T.

Table 1 shows example of term dictionary generated using term similarity calculator for ACSIR method. The visual synonyms are compared with iLike method in which synonyms are computed using K–S method.

3.3.4 Search module

This module recommends ranked images for given user input query. The query is processed as follows:

(a) *Text-based search* Images are retrieved from image database for given input query q and for each term of q and stored in S_{actual} and $S_{expanded}$, respectively. For q, visual synonyms are selected from term dictionary and expanded queries q' are formed. Images are retrieved for query q first followed by q' and stored in S_{actual} and $S_{expanded}$, respectively.

(b) *Pair-wise image similarity calculator and ranking images based on similarity score* Textually retrieved images need to be ranked based on visual meaning of terms to achieve higher degree of relevance. Two-step image similarity measure is applied on S_{actual} and $S_{expanded}$ to rank the images based on visual similarity. Pair-wise image similarity for first and second iterations is computed by Function 3 and Function 4, respectively.

For each pair of images (I_i,I_j) in S_{actual}, image similarity is computed by cosine similarity measure on visual features by using Eq. 4.

$$Image_{similarity}(I_i, I_j) = \frac{\sum_{k=1}^{F_c} feature_{(X_k,I_i)} * feature_{(X_k,I_j)}}{\sqrt{\sum_{k=1}^{F_c}(feature_{(X_k,I_i)})^2} * \sqrt{\sum_{k=1}^{F_c}(feature_{(X_k,I_j)})^2}} \quad (4)$$

In Function 3, $Image_{pivot}(I_x,I_y)$ indicates the most relevant visual meaning of textual query q.

Table 1 Term dictionary example

Keyword	iLike—visual synonyms	ACSIR—visual synonyms
Black	Fila, GAS, Stamp, Gray, Dark, Carlton, Brick	Dark, Gray, Franco, Carlton, FILA
Shoes	Unisex, Adi, Adidas, Carlton, Puma	Unisex, Jack, Gliders, Adidas, Red-chief, Loafers, Footwear
Footwear	Slip-on, Slippers, Floater, Casual, Sole	Shoes, Sandals, Slip-ons, Slippers, Floater, Loafers
Flip-flops	Footwear, Slip-on, Disney, Flat, Lightweight	Slip-on, Slippers, Sandals, Footwear, Disney, Sole, Adidas
Dark	Light, Pale, Black, Purple, Maroon	Maroon, Purple, Green, Black, Gray, Navy, Teal, Shaded
Shirt	Slim Fit, Check, Checked, Coffee, Light, Colored	Slim Fit, Aron, Arrow, Casual, Formal, Rare, Checked, Coffee
Runner	Run, Water, Cross, Running, Mustard, Air	Air, White-and-Maroon, Running, Sports, Footwear, Shoes
Occasion	Mainline, Resistant, Semi-Formal, Brogue, Comfort	Exhibit, Proterra, Burnish, Comfort, Casual, Ethnic
Leather	Siliciano, London, Lea, Formal, Carlton	Globalite, Formal, Black, Dark, London, Carlton
Comfort	Semi-Formal, Mainline, Two Tone, Burgundy, Occasion	Burnish, Occasion, Burgundy, Proterra, Formal, Sports

Function 3: Cosine Single Iteration

Function: CosineSingleIteration
Data: Consider Images in S_{actual}. Generates Pivotal Image pair $Image_{pivot}(I_x,I_y)$.

begin
- Let m = size of S_{actual}
- **for** i = *1 to* $m-1$ **do**
  - **for** $j = i + 1$ *to* m **do**
    - Calculate pair-wise Image Similarity between Image(I_i,I_j) using Eq. 4
    - Add Image Similarity Score for each pair into $Image_{score}[\,]$
- Select Image pair with Maximum Similarity Score from $Image_{score}[\,]$ as pivotal Image $Image_{pivot}(I_x,I_y)$ and add it to Lookup Buffer LB

Images are re-ranked with reference to $Image_{pivot}(I_x,I_y)$ as shown in Function 4.

3.3.5 Search user interface

Search user interface is provided to enter textual search query q. Once user enters input query, ranked images are recommended using Function 5.

In Function 5, *isDBModified* is the flag value set by the continuous integration (CI) server. It indicates addition of new images in database above the *DBConsistency* threshold. This threshold is set by the online retailer to trigger CI server module to update visual synonyms and pivotal images in database. *isEncounteredEarlier* flag is used to check the occurrence of the query in past.

3.3.6 Continuous integration module

Continuous integration is the practice in software engineering to continuously test system integrity. CI servers are the applications deployed on a server, which runs periodically.

Function 4: Cosine Second Iteration

Function: CosineSecondIteration
Data: Consider Images in S_{actual}, $S_{expanded}$ and Pivotal Image pair $Image_{pivot}(I_x,I_y)$. Generates Image list with Similarity Score

begin
- Let m = size of S_{actual}
- **for** i = *1 to* m **do**
  - **if** $i \neq x$ *and* $i \neq y$ **then**
    - Calculate pair-wise Image Similarity between I_x and I_i in S_{actual} using Eq. 4 and store in $ImageScore_{actual,x}[\,]$
    - Calculate pair-wise Image Similarity between I_y and I_i in S_{actual} using Eq. 4 and store in $ImageScore_{actual,y}[\,]$
- Merge $ImageScore_{actual,x}[\,]$ and $ImageScore_{actual,y}[\,]$, then sort in descending order based on Similarity Score and store in $Image_{actual-sort}[\,]$
- Let $Image_{ranked}[\,]$ = ranked images
- **for** *each pair of Images* (I_i,I_j) *in* $Image_{actual-sort}[\,]$ **do**
  - **if** I_i *is not present in* $Image_{ranked}[\,]$ **then**
    - Add I_i in $Image_{ranked}[\,]$
  - **if** I_j *is not present in* $Image_{ranked}[\,]$ **then**
    - Add I_j in $Image_{ranked}[\,]$
- Let $S'_{expanded} = S_{expanded} - S_{actual}$ be the images that are not matched with pivotal images.
- Compute ranked images for $S'_{expanded}$ with $CosineSecondIteration(S'_{expanded},0,Image_{pivot}(I_x,I_y))$

In this framework, visual synonyms and pivotal images are periodically tested and updated by checking *DBConsistency* threshold.

3.4 Algorithm

In this section, ANOVA Cosine Similarity Image Recommendation in vertical search (ACSIR) is presented in

Function 5: Search Criteria

Function: SearchCriteria
Data: Consider Input Query q, $isDBModified$ Flag, $isEncounteredEarlier$ Flag. Recommends Top-k Ranked Images

begin
- **if** *isDBModified flag is TRUE and isEncounteredEarlier flag is FALSE* **then**
  - Perform Text based search to retrieve Image Datasets S_{actual} and $S_{expanded}$.
  - Retrieve Pivotal Image pair $Image_{pivot}(I_x, I_y)$ using $CosineFirstIteration(S_{actual})$
  - Generate Image list with Similarity Score using $CosineSecondIteration(S_{actual}, S_{expanded}, Image_{pivot}(I_x, I_y))$
- **else**
  - **if** *isDBModified flag is FALSE* **then**
    - **if** *isEncounteredEarlier flag is TRUE* **then**
      - Load Pivotal Image Pair $Image_{pivot}(I_x, I_y)$ from Lookup Buffer LB for Query q
      - Generate Image list with Similarity Score using $CosineSecondIteration(S_{actual}, S_{expanded}, Image_{pivot}(I_x, I_y))$
    - **else**
      - Perform Text based search to retrieve Image Datasets S_{actual} and $S_{expanded}$.
      - Retrieve Pivotal Image pair $Image_{pivot}(I_x, I_y)$ using $CosineFirstIteration(S_{actual})$
      - Generate Image list with Similarity Score using $CosineSecondIteration(S_{actual}, S_{expanded}, Image_{pivot}(I_x, I_y))$
  - Recommend Top-k images from $Image_{ranked}[\,]$

Algorithm 1: ANOVA Cosine Similarity Image Recommendation in Vertical Search

Input : User Input Query q, Image Database Img_{db} with Image Img and its description Img_{desc}, $isDBModified$ Flag, $isEncounteredEarlier$ Flag
Output: Top-k Recommended Images

begin
- Offline :
- Compute Visual features for each image Img in Img_{db}
- Let T = Unique terms present in Img_{desc}
- Compute ANOVA p values using $ANOVAp\ valueCalculator(T)$
- For each pair of terms(T_i, T_j) compute term similarity $Term_{similarity}(T_i, T_j)$ using Eq. 2.
- Construct Term Dictionary using Visual Synonym Calculator
- Online:
- **for** *input query q* **do**
  - Recommend Top-k images using $SearchCriteria(q, isDBModified, isEncounteredEarlier)$

Algorithm 1. It has two phases: (i) pre-processing (offline) and (ii) image recommendation (online). In offline phase, (i) visual features are extracted and stored for all images, (ii) for each term, ANOVA p values for every feature are computed, (iii) textual term similarity is computed for each pair of term, and (iv) term dictionary is generated using term pairs with maximum similarity score. In online phase, (i) for given user input query, expanded queries are generated using the synonyms present in terms of dictionary computed offline. (ii) Text-based search is performed for base and expanded queries to generate actual and expanded image datasets, respectively. (iii) These retrieved images are re-ranked and recommended based on image similarity score.

4 Experiments

4.1 Data collection

In this experiment, 5582 images are crawled from *e*-commerce website *myntra.com* using customized crawler. This website is an online product vendor, selling products like apparels, footwear, electronics and appliances. The site provides annotated images with categories, price, product description and image name. On crawling, visual features of each image are extracted using image processor and are stored in an offline database. A dictionary of unique keywords is formed by removing stop-words and irrelevant tags from product description. This dictionary contains 589 keywords, created using custom text parser. Feature and keyword extraction process is repeated for newly added images to maintain updated dataset.

4.2 Experiment setup

Proposed method ACSIR works on text-based image retrieval with visual meaning of content. Hence, iLike [29] is used as a baseline for comparison. Both the methods follow text-based image retrieval along with visual synonyms. In ACSIR and iLike, visual synonyms are computed offline using ANOVA and K–S method, respectively. p value is computed using JavaNPST [33] and the Apache Commons Mathematics Library [34] for ANOVA and K–S method, respectively. In ACSIR, *DBConsistency* threshold is set to 5% as per our choice; *isDBModified* and *isEncounteredEarlier* flags are set to false initially. For a given input query top-5 synonyms were loaded to form expanded queries for both methods.

4.3 Performance evaluation

In this section, image recommendation results are compared and discussed for both iLike and ACSIR methods. The exper-

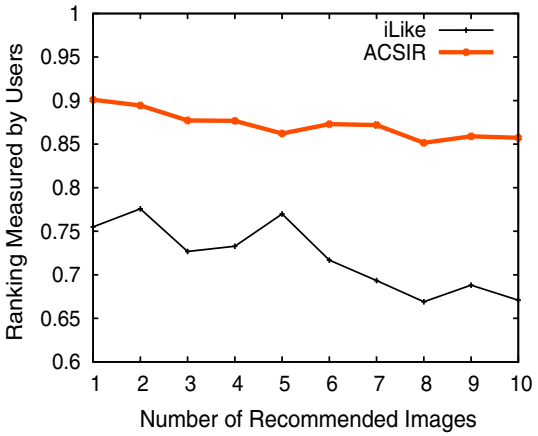

Fig. 2 User evaluation for image recommendations ranking with ACSIR and iLike method

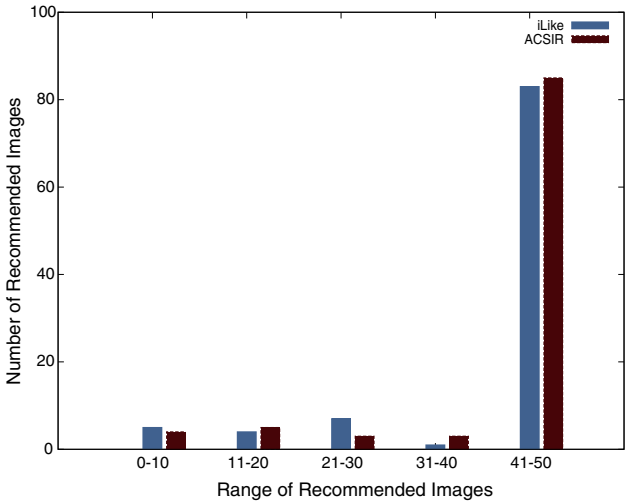

Fig. 3 Coverage of recommended images with ACSIR and iLike method

iments carried out on a 4GB DDR2 RAM, Intel(R) Core(TM) i5 @2.40 GHz processor system. The higher configuration systems may yield better results. Image dataset discussed in data collection is used for evaluation of both the methods. Ranking of top-10 recommended images is considered as a performance metric. Hundred test queries are used for evaluation. Hundred users, including student, research scholars and teaching staff, are invited to evaluate relevance of recommended images. Each user is allocated two queries and asked to evaluate relevance of ranked recommended images with the relevance score between 0 and 1. Here, 0 and 1 indicate totally irrelevant and highly relevant images, respectively. Mean values of users' relevance score are computed for top-1 to top-10 images.

Figure 2 shows ranking relevance score evaluated by the users for image recommendations with ACSIR and iLike methods. It is observed from the graph that the relevance score is in decreasing order from top-1 to top-10 images in

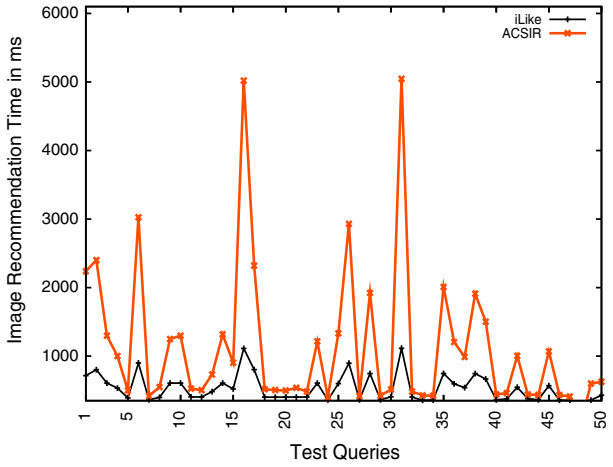

Fig. 4 Time required to recommend images with ACSIR and iLike method with first iteration

both the methods, but recommended images are relevant and ranked in proper order in ACSIR method. The mean of relevance score of ranked images of ACSIR method is better by 15.26% for top-10 images in comparison with the iLike method.

Figure 3 shows number of recommended images displayed as output. It is observed from the graph that the number of images displayed for recommendations is more in ACSIR method than CBIR method if top-50 images are considered.

Figure 4 shows time required to recommend images with ACSIR and iLike method with first iteration for 50 queries. It is observed that time required in ACSIR method is more than the iLike method. The mean image recommendation time for iLike and ACSIR is 393.38 and 845.62 ms, respectively, for 100 test queries. This is because in iLike method, images are recommended based on text-based search, while, in ACSIR method, first images are retrieved using text-based search and re-ranked by computing cosine similarity.

Figure 5 shows time required to recommend images with ACSIR method in six iterations for 50 queries. Figure 6 shows time required to recommend images with ACSIR method in sixth iteration and iLike for 50 queries. The mean image recommendation time from first to six iterations is 845.62, 507.57, 461.48, 423.06, 390.56 and 390.56, respectively, in ACSIR method. It is observed that from fifth iteration onward the image recommendation time is consistent. It is observed from the graph that time required to recommend images with ACSIR method in fifth iteration is comparable with iLike. In ACSIR method, pivotal image is identified in first iteration and stored in the lookup buffer and is accessed second iteration onward. Hence, the image recommendation time gradually decreases.

Figures 7, 8, 9, 10, 11, 12, 13, 14, 15 and 16 show the top-5 recommended images for input queries *dark formal*

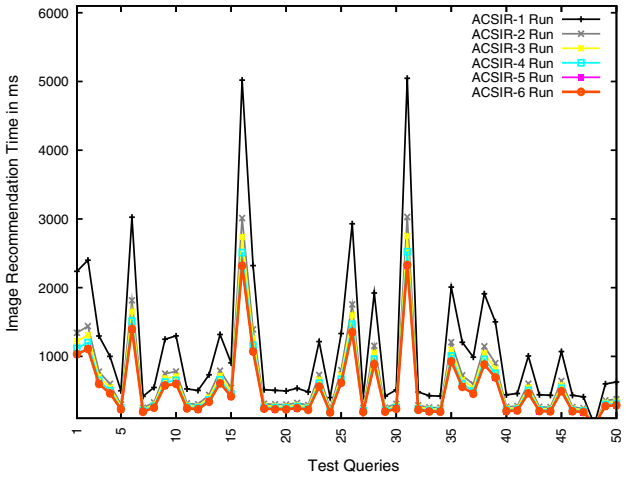

Fig. 5 Time required to recommend images with ACSIR method with six iterations

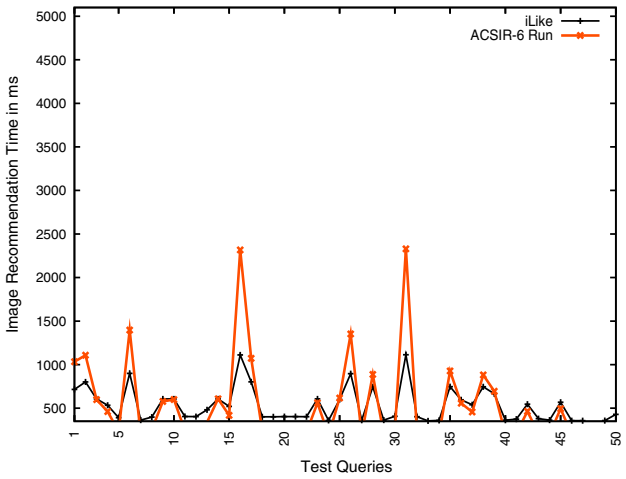

Fig. 6 Time required to recommend images with ACSIR method in sixth iteration and iLike

shirt, *disney flip-flops*, *navy blue*, *yellow green* and *white red footwear* for iLike and ACSIR methods, respectively. It is observed from Fig. 7 that images (a) and (d) are darker in color but brighter in shade, while images in Fig. 8 are all in dark shade. In Fig. 9, images (b) and (d) are disney flip-flops but do not contain disney characters on them, while images in Fig. 10 contain disney characters also. In Figs. 11, 12, 13, 14, 15 and 16, recommended images are color specific. It is observed that recommended images in Figs.12, 14 and 16 possess higher degree of color relevance in comparison with images in Figs. 11, 13 and 15.

The ACSIR method has few advantages over iLike method which are mentioned below; ACSIR outperforms iLike by ranking images in proper order relevant to user input query.

1. Visual synonyms are computed for each term using ANOVA and K–S method in ACSIR and iLike, respectively. One-way ANOVA uses F-statistics on samples with equal variances, normal distribution and independent errors. If $p4$ value is small, reject the null hypothesis which indicates that all means are same for different groups. K–S method uses D-statistics by computing cumulative frequency. If D-statistics is greater than critical value, then the null hypothesis is rejected. The time required to compute p value with ANOVA is much lower than K–S test.
2. It is observed from Table 1 that the order of the synonyms computed with ACSIR is more relevant to keyword than iLike. To support this statement, images are recommended for query *black* as shown in Figs 17 and 18. It is observed from Fig. 18 that images cover more categories

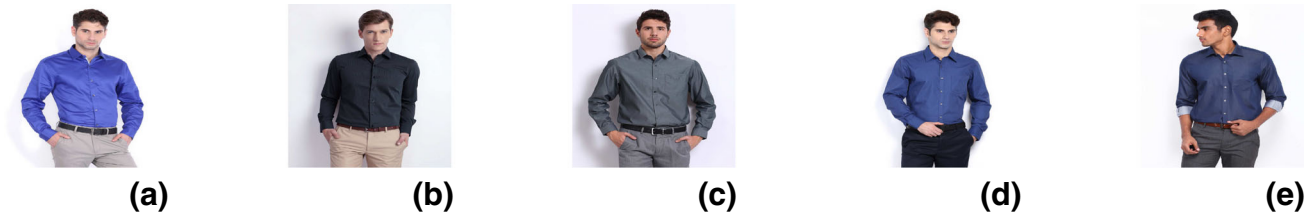

Fig. 7 Top-5 image recommendations for query *dark formal shirts* with iLike method

Fig. 8 Top-5 image recommendations for query *dark formal shirts* with ACSIR method

(a) (b) (c) (d) (e)

Fig. 9 Top-5 image recommendations for query *disney flip-flops* with iLike method

(a) (b) (c) (d) (e)

Fig. 10 Top-5 image recommendations for query *disney flip-flops* with ACSIR method

(a) (b) (c) (d) (e)

Fig. 11 Top-5 image recommendations for query *navy blue* with iLike method (color figure online)

(a) (b) (c) (d) (e)

Fig. 12 Top-5 image recommendations for query *navy blue* with ACSIR method (color figure online)

(a) (b) (c) (d) (e)

Fig. 13 Top-5 image recommendations for query *yellow green* with iLike method (color figure online)

(a) (b) (c) (d) (e)

Fig. 14 Top-5 image recommendations for query *yellow green* with ACSIR method (color figure online)

Fig. 15 Top-5 image recommendations for query *white red footwear* with iLike method (color figure online)

Fig. 16 Top-5 image recommendations for query *white red footwear* with ACSIR method (color figure online)

Fig. 17 Top-5 image recommendations for query *black* with iLike method

Fig. 18 Top-5 image recommendations for query *black* with ACSIR method

and preserve semantic meaning of query term, whereas images in Fig. 17 cover images with semantic meaning only.

3. In iLike, text-based search is performed by forming expanded queries using visual synonyms, whereas in ACSIR, first images are retrieved using expanded queries with visual synonyms and a two-step image re-ranking algorithm using pair-wise cosine similarity is applied. Hence, ACSIR method recommends more relevant images to the input query.
4. ACSIR framework facilitates mechanism for handling newly added images.

5 Conclusions

In this paper, we have proposed ANOVA Cosine Similarity Image Recommendation (ACSIR) framework in vertical image search. Various category products with its description are crawled from *myntra.com* website. Visual features are computed, normalized and stored in the database for all the images offline. Weight of the each term present in description is computed using ANOVA p value by combining text-based search and visual features of the images. Visual synonyms are computed using term similarity. Expanded queries are generated for user input query, and text-based search is per-

formed. Cosine similarity is computed between two images, and images are recommended based on similarity score. Experiments are conducted on crawled image data from *myntra.com* website, and results are compared with iLike method.

Relevance score is used to evaluate quality of ranked images, which is evaluated manually with the help of users. The accuracy of relevance score of ACSIR increases by 15.26% for top-10 recommended images in comparison with iLike. The mean image recommendation time for iLike and ACSIR is 393.38 ms and 845.62 ms, respectively, for 100 test queries with first iteration. In ACSIR method, pivotal image is identified in first iteration and stored in lookup buffer and is accessed second iteration onward. Hence, the image recommendation time gradually decreases. For ACSIR, the mean image recommendation time from first to six iterations is 845.62, 507.57, 461.48, 423.06, 390.56 and 390.56, respectively. From fifth iteration onward it is comparable with iLike.

References

1. Idrissi N, Martinez J, Aboutajdine D (2009) Bridging the semantic gap for texture-based image retrieval and navigation. J Multimed 4(5):277–283
2. Feng F, Wang C, Yao Y, Deng K, Zhang L, Ma WY (2006) IGroup: a web image search engine with semantic clustering of search results. In: Proceedings of the 14th annual ACM international conference on multimedia, pp 497–498
3. Ma H, Zhu J, Lyu MRT, King I (2010) Bridging the semantic gap between image contents and tags. IEEE Trans Multimed 12(5):462–473
4. Luo B, Wang X, Tang X (2003) World Wide Web based image search engine using text and image content features. Electron Imaging 2003:123–130
5. Cui J, Wen F, Tang X (2008) Real time google and live image search re-ranking. In: Proceedings of the 16th ACM international conference on multimedia, pp 729–732
6. Sejal D, Abhishek D, Venugopal KR, Iyengar SS, Patnaik LM (2016) IR_URFS_VF: image recommendation with user relevance feedback session and visual features in vertical image search. Int J Multimed Inf Retr 5(4):255–264
7. Gudivada VN, Raghavan VV (1995) Content based image retrieval systems. Computer 28(9):18–22
8. Zachary JM, Iyengar SS (1999) Content based image retrieval systems. In: Proceedings of IEEE symposium on application-specific systems and software engineering and technology, pp 136–143
9. Stanchev PL (2001) Content-based image retrieval systems. In: Proceedings of CompSysTech'2001, p 1
10. TANG LJ, DUAN L, GAO W (2001) Content based image retrieval system. Appl Res Comput 7:1–4
11. Smeulders AWM, Worring M, Santini S, Gupta A, Jain. Ramesh (2000) Content-based image retrieval at the end of the early years. IEEE Trans Pattern Anal Mach Intell 22(12):1349–1380
12. Datta R, Li J, Wang JZ (2005) Content-based image retrieval: approaches and trends of the new age. In: Proceedings of the 7th ACM SIGMM international workshop on multimedia information retrieval, pp 253–262
13. Akakin HC, Gurcan MN (2012) Content-based microscopic image retrieval system for multi-image queries. IEEE Trans Inf Technol Biomed 16(4):758–769
14. Ramachandra A, Abhilash S, Raja KB, Venugopal KR (2012) Feature level fusion based bimodal biometric using transformation domine techniques. IOSR J Comput Eng (IOSRJCE) 3(3):39–46
15. Lavanya BN, Raja KB, Venugopal KR, Patnaik LM (2009) Minutiae extraction in fingerprint using gabor filter enhancement. In: International conference on advances in computing, control, and telecommunication technologies, pp 54–56
16. Akbas E, Vural FTY (2007) Automatic Image Annotation by Ensemble of Visual Descriptors. In: Proceedings of CVPR'07, IEEE conference on computer vision and pattern recognition, pp 1–8
17. Bartolini I, Ciaccia P (2010) Multi-dimensional keyword-based image annotation and search. In: Proceedings of the 2*nd* international workshop on keyword search on structured data, pp 5–10
18. Wang C, Jing F, Zhang L, Zhang HJ (2007) Content-based image annotation refinement. In: Proceedings of CVPR'07, IEEE conference on computer vision and pattern recognition, pp 1–8
19. Li J, Wang JZ (2008) Real-time computerized annotation of pictures. IEEE Trans Pattern Anal Mach Intell 30(6):985–1002
20. Wang C, Jing F, Zhang L, Zhang HJ (2006) Image annotation refinement using random walk with restarts. In: Proceedings of the 14*th* annual ACM international conference on multimedia, pp 647–650
21. Makadia A, Pavlovic V, Kumar S (2008) A new baseline for image annotation. Comput Vis ECCV 2008:316–329
22. Verma Y, Jawahar CV (2012) Image annotation using metric learning in semantic neighbourhoods. Comput Vis ECCV 2012:836–849
23. Wang C, Blei D, Li FF (2009) Simultaneous image classification and annotation. In: Proceedings of CVPR 2009, IEEE conference on computer vision and pattern recognition, pp 1903–1910
24. Krapac J, Allan M, Verbeek J, Jurie F (2010) Improving web image search results using query-relative classifiers. IEEE Conf Comput Vis Pattern Recognit (CVPR) 2010:1094–1101
25. Ben-Haim N, Babenko B, Belongie S (2006) Improving web-based image search via content based clustering. In: Proceedings of international conference on computer vision and pattern recognition workshop, p 106
26. Tang X, Liu K, Cui J, Wen F, Xiaogang Wang (2012) Intentsearch: capturing user intention for one-click internet image search. IEEE Trans Pattern Anal Mach Intell 34(7):1342–1353
27. Fan J, Keim DA, Gao Y, Luo H, Li Zongmin (2009) JustClick: personalized image recommendation via exploratory search from large-scale flickr images. IEEE Trans Circuits Syst Video Technol 19(2):273–288
28. Chen Y, Yu N, Luo B, Chen X (2010) iLike: integrating visual and textual features for vertical search. In: Proceedings of the international conference on multimedia, pp 221–230
29. Chen Y, Sampathkumar H, Luo B, Chen XW (2013) iLike: bridging the semantic gap in vertical image search by integrating text and visual features. IEEE Trans Knowl Data Eng 25(10):2257–2270
30. Haralick RM, Shanmugam K, Dinstein I (1973) Textural features for image classification. IEEE Trans Syst Man Cybern 3(6):610–621
31. Tamura H, Mori S, Yamawaki T (1978) Textural features corresponding to visual perception. IEEE Trans Syst Man Cybern 8(6):460–473
32. Moreno P, Bernardino A, Santos-Victor J (2005) Gabor parameter selection for local feature detection. In: Marques JS, Pérez de la Blanca N, Pina P (eds) Pattern recognition and image analysis. IbPRIA 2005. Lecture notes in computer science, vol 3522. Springer, Berlin, Heidelberg
33. Derrac J, García S, Herrera F (2015) JavaNPST: nonparametric statistical tests in Java. arXiv preprint arXiv:1501.04222
34. Math C (2016) The apache commons mathematics library. https://commons.apache.org/proper/commonsmath/