TUT MUVIS IMAGE RETRIEVAL SYSTEM PROPOSAL FOR MSR-BING CHALLENGE 2014

J. Raitoharju, H. Zhang, E.C. Ozan, M.A. Waris, M. Faisal, G. Cao, M. Roininen,
I. Ahmad, R. Shetty, S. P.C., S. Uhlmann, K. Samiee, S. Kiranyaz, M. Gabbouj

Tampere University of Technology
jenni.raitoharju@tut.fi

ABSTRACT

This paper presents our system designed for MSR-Bing Image Retrieval Challenge @ ICME 2014. The core of our system is formed by a text processing module combined with a module performing PCA-assisted perceptron regression with random sub-space selection ($\mathrm{P^2R^2S^2}$). $\mathrm{P^2R^2S^2}$ uses OverFeat features as a starting point and transforms them into more descriptive features via unsupervised training. The relevance score for each query-image pair is obtained by comparing the transformed features of the query image and the relevant training images. We also use a face bank, duplicate image detection, and optical character recognition to boost our evaluation accuracy. Our system achieves 0.5099 in terms of DCG_{25} on the development set and 0.5116 on the test set.

Index Terms— Image Retrieval, Relevance Evaluation, Data Partitioning, Face Bank

1. INTRODUCTION

We aim at web-scale image retrieval in MSR-Bing Image Retrieval Challenge @ ICME 2014. The challenge leverages the click data to bridge the semantic and intent gap using a newly released image dataset "Clickture-Lite" [1]. This dataset was sampled from one-year click log of Bing search engine. The data is organized by triads of queries, images and clicks as $Clickture = \{\mathcal{K}, \mathcal{Q}, \mathcal{C}\}$, where a query ($\mathcal{Q}$)-image ($\mathcal{K}$) pair is coupled with the number of clicks ($\mathcal{C}$) from the search results. In general, more clicks imply higher relevance between the query and image. Due to the abundance of click data generated by the search engine and its unique contribution in image retrieval, previously people attempted to improve the ranking results using methods including top query modeling, image annotation by query modeling and rank learning [1]. As followed in this paper, we describe our method to tackle this problem. In Section 2 we give a detailed description of our system and in Section 3 we show some experimental results. Section 4 concludes the paper.

2. SYSTEM OVERVIEW

Figure 1 shows a diagram of the overall system used in our master submission. We have four decision making modules, which all rely on results returned by the text processing module. If the text processing module fails to return anything, we resort to random guess in our decision making. In practice the core of our system is a module performing PCA-assisted perceptron regression with random sub-space selection ($\mathrm{P^2R^2S^2}$) and under certain conditions it is assisted by the three other decision making modules: face bank, duplicate detector, and optical character recognizer. All decision making modules return a relevance score and a reliability score to each query-image pair. In most cases both scores are between 0 and 1. Only duplicate image detector may produce relevance scores above 1 to ensure that certain images will be always ranked first regardless of scores assigned to other images. The merging module uses both reliability and relevance scores to decide the final output. All modules are presented in more detail in Sections 2.1 - 2.7 as shown in the figure. Our secondary submissions along with a method we considered, but did not use in our submissions, are briefly presented in Section 2.8.

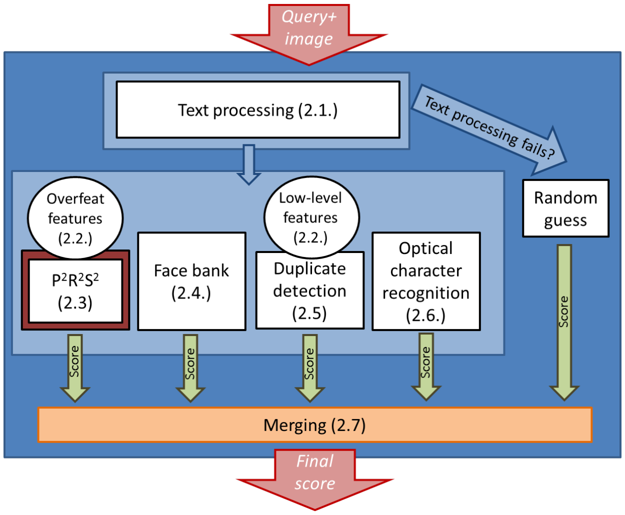


Fig. 1. Overview of MUVIS team's master submission

Thanks to Merope cluster for computational resources and to Juha Herrala and Mika Anttila for their technical support.

2.1. Text Processing

Query texts are normally short English phrases which are written in lose grammatical forms and possibly with typographical errors. A large number of query texts contain geographical terms, person names, and other identity names. Given a probing query text, the text processing module finds the most relevant query texts from the training dataset and returns the images associated with these query texts.

The text processing module first converts each query text into a unique semantic ID –a set of word stems– in order to merge different forms of the same query into one entry. This procedure includes the following steps:

- Split the query text into words and perform part-of-speech tagging. After the tagging, nouns, verbs, adjectives and adverbs are kept. All other types of words are discarded.
- Lemmatize the words using WordNet engine [2] so that different forms of a word are represented by their stems.
- Remove meaningless words for image retrieval. Our blacklist includes “image”, “picture”, “free”, etc.

When searching in the training database, we also try to find all combinations of synonyms of the probing query text. For example, the module should find “Christmas picture” if the probing query text is “Xmas picture”. We use WordNet engine to find all synonyms of a word.

If the exact semantic ID is not found in the training data set, we try to find queries whose semantic IDs are supersets of the semantic ID of the probing query text. These expanded queries are reliable expansions of the probing query text. For example “drone aircraft” and “crashed drones” are reliable expansions of the query text “drone”. If reliable queries are not found, we will try to find queries whose semantic IDs partially overlap with the semantic ID of the probing query word set. This type of expansion may find queries that are semantically different from the probing query text. These queries are considered as unreliable expansion to the probing query text. For example “man’s face” and “man’s face looking up” are unreliable expansion of the query text “man’s face looking down”.

If neither the exact match nor the expansion procedure finds relevant queries in the training dataset, we apply the Hunspell [3] text autocorrection to correct possible typos and do the query search and expansion again. If even autocorrection does not help to find any relevant queries in the training dataset we consider that text processing has failed and we resort to random guess in our decision making.

2.2. Features

2.2.1. OverFeat

We use the OverFeat [4] convolutional network-based image features extractor and classifier for extracting features from the dataset images. OverFeat has been trained on the ImageNet dataset for classifying between 1000 image categories, and the authors provide two networks with slightly different topologies. We use the 1000-dimensional output layer of the smaller network as a descriptor for the dataset images, i.e., each image is described by its correlation with the ImageNet classes. Prior to feature extraction the input images are resized to match the network input resolution of 231x231 pixels by first uniformly scaling the image so that the smaller image dimension equals 231, and then cropping the larger dimension equally from both sides to match the required resolution. This way of resizing ensures that aspect ratio will not distort, but some information on the image borders is lost. The method has produced satisfactory results in experimental testing. In our system OverFeat features are used by the $\mathrm{P^2R^2S^2}$ module as described in Section 2.3.

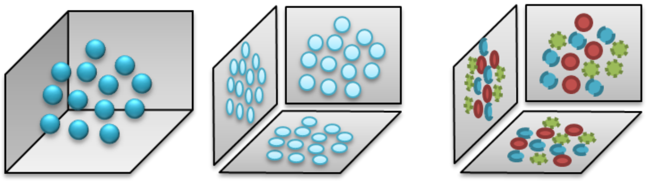
2.2.2. Low-level features

We also use low-level features to detect whether the query image is a duplicate or near duplicate of a training set images. After some experimental testing we selected Local Binary Patterns (LBP) [5] and Color Structure Descriptor (CSD) [6] features for this purpose.

2.3. PCA-assisted Perceptron Regression with Random Subspace Selection ($\mathbf{P^2R^2S^2}$)

The core of our system is a module performing PCA-assisted perceptron regression with random sub-space selection ($\mathrm{P^2R^2S^2}$). $\mathrm{P^2R^2S^2}$ aims at digging deep into the big data and reaching the information hidden among the huge number of samples and data dimensions. The main idea is not to investigate the dataset as a whole, but partition it into smaller entities and reveal the unseen. Partitioning is applied on both feature space and data samples and it is done in a fully randomized way.

Traditional data mining approaches tend to investigate further into the dimensions of the sample space, identifying and selecting the most informative features, and finding out the correlation between different dimensions [7, 8]. $\mathrm{P^2R^2S^2}$ approaches this problem first by forming different subspaces of the original feature spaces. The subspaces are formed using random selection among dimensions without replacement. We form N sub-feature spaces each consisting of D dimensions of the original feature space. N and D are set so that resulting feature spaces are overlapping. Each dimension is included in at least one subspace. Unlike the traditional feature selection methods, sub-feature space formation in $\mathrm{P^2R^2S^2}$ does not aim at decreasing the final feature space



a) Original data b) Sub-feature spaces c) Sample partitioning

Fig. 2. Example of partitioning feature space and data samples.

dimension, but increase it, yet still keeping the investigation in smaller dimensions. Each randomly generated new feature space is later thoroughly investigated using principal component analysis (PCA) -assisted perceptron regression.

Next, $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ divides also the sample set into smaller partitions. Using a bagging-like approach, we select randomly with replacement P partitions consisting of S samples each. The objective of sample set partitioning is to reduce the amount of samples per examination, enable parallelization and increase stability and accuracy of the applied learning method [9]. Similar to the previous step, this step also increases the total number of samples to be examined, but decreases the number of samples per examination. A simplified example of partitioning feature space and data samples is shown in Figure 2

The objective of $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ is to generate reproducible and evocative representations. Data representations can be obtained in various ways, such as clustering, modelling, or codeword generation. $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ builds representation in a way, which does not only rely on a distance defined in the original feature space, but by its nature, also investigates the subdimension relations of the given space. For each partition, we train a regressor to represent the behavior of the samples in the corresponding partition. Regressors, instead of clusters and models, are trained in a supervised manner. In other words, for each sample used in the training of a regressor, a desired output must be presented. Since the amount of different class labels may be huge, we set the outputs using unsupervised data investigation. This also allows further investigation of the feature space, independent of the semantic relations indicated by the corresponding label.

Supervision in an unlabeled dataset is possible using the assistance of PCA. The desired output of a given sample is obtained by the PCA projection of the corresponding sample vector [10]. If the projected value is greater than 0, the output is set to 1, else it is -1. However these output values are just for initialization. The output values together with the distribution of samples at hand may not lead to a successful regressor training. In that case, an iterative approach is followed. The training samples are evaluated using the trained regressors and the corresponding responses for each sample are generated. The generated response is again thresholded and new output values are generated in order to be used in the next iteration of the training. In other words, if the response of a sample is greater than 0 after training and propagation, the desired output corresponding to that sample is set to be 1 in the next training iteration. After training, the mean squared error is computed and if it is lower than a predefined value, training is assumed to converge. Each converged regressor is stored. We use the first V principal vectors to generate projections and store each converged regressor.

When a new sample vector is presented to the trained system, its feature vector is divided into sub-vectors using the same randomly generated sub-feature spaces previously used in training. Then each sub-vector is propagated through corresponding regressors and the responses are concatenated. The concatenation occurs also for different subspaces. In other words, a sample vector of 1000 is first divided into N different vectors of D dimensions, then each D dimensional subvector is propagated through P regressors with outputs of V dimensions. So the initial 1000 dimensional vector is transformed into a $DxVxP$ dimensional vector. Corresponding vectors are used in retrieval, matching by a selected distance function.

For MSR-Bing Image Retrieval Challenge, we first extracted OverFeat features from all the training set images. During the test set evaluation, we extracted OverFeat features from the test image and using the $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ module we transformed the feature vectors of the test image and the set of associated training images returned by the text processing module. We used the L1 distance to compute the similarity of feature vectors and the relevance score for each query-image pair was then determined using the following approach: First, the L1 distances between the transformed feature vectors of the query image and given training image are calculated. Any example with a click count lower than 2 is discarded, unless those are the only examples at hand. The weighted average of the distances of the closest three vectors is calculated. The weights used in this calculation are obtained by natural logarithm of the click counts of the corresponding training images. Finally, this average distance is converted to a relevance score using a negative exponential function. The reliability score of the $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ is determined based on the similarity of the query texts of the test image and the closest associated training images.

2.4. Face Bank

The objective of our face bank module is to enhance the query-image relevance evaluation, when a face is detected in the query image. We trained several multi-block LBP based face detectors to obtain pose-invariant face detection [11]. Detectors are trained for the following yaw-angles: $\theta = \{0°, \pm 30°, \pm 60°\}$. A maximum-take-all voting strategy is used to merge the output of each detector and obtain the final face localization.

We created a face bank for 2531 well-known celebrities selected from www.pose24.com. For each celebrity, we collected 20 images with different facial pose angles to ensure a better recognition across all face angles. If a face is detected in an image, it is compared with every image in the face bank and a feature vector is formed as a histogram of relevant matches i.e. feature vectors have bins for all the 2531 individuals. Relevance of two faces is evaluated using the face recognition module provided in Intel Perceptual Computing [12]. This approach produces a face feature vector also for face images of persons not in the face bank and it may provide a way to compare similarities of such face images with respect to their similarities with the face bank celebrities.

In the test evaluation of MSR-Bing Image Retrieval Challenge, for a given query text, we downloaded the precomputed face feature vectors for the associated images returned by the text processing module and computed the Euclidean distances between the query images and associated training images. If a match is detected high relevance and reliability scores are returned. The final relevance score of our system is based on the face bank only when it is highly confident about positive match. The overview of the face bank module is shown in Figure 3.

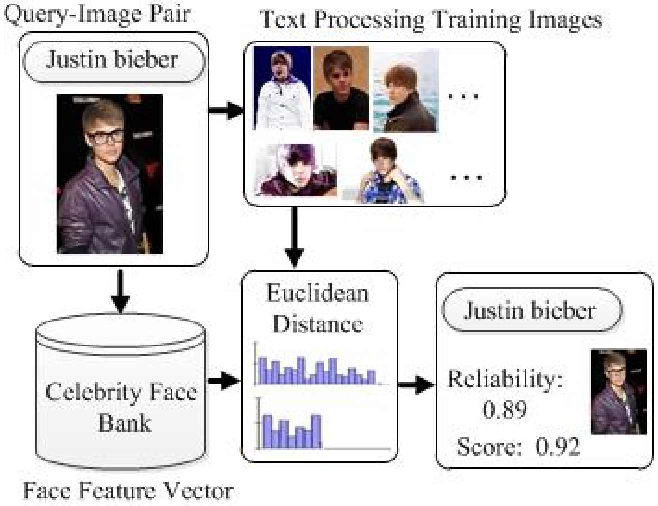


Fig. 3. Overview of relevance evaluation using the face bank

2.5. Duplicate Image Detection

We assume that if the query image is a (near)duplicate of an image associated with the same or a near query text in the training set, the query-image pair should be evaluated to be an excellent match. Therefore, to enhance the query-image relevance results, we created a duplicate image detector which was used to identify whether the query image is a duplicate or a near duplicate of a relevant image from the training set. We used the Euclidean distance over LBP and CSD features to measure the image similarity. We consider images to be near duplicates if both distances are below a given threshold Δ i.e.

$$D_{LBP}(qimg, timg) \& D_{CSD}(qimg, timg) \leq \Delta, \quad (1)$$

where $D_{LBP}(qimg, timg)$ is the Euclidean distance of LBP features vectors extracted from the query image, $qimg$, and the training image, $timg$, and similarly $D_{CSD}(qimg, timg)$ is the Euclidean distance of CSD feature vectors. Duplicate images are first searched only among the images associated with training query texts having the exact semantic ID and a click count higher than 1. If a duplicate among those images is detected, the module returns reliability score of 1 and the click count of the duplicate image as the relevance score. If there are no training queries having exactly the same semantic ID with the query text, duplicate images are also searched among the images associated with reliable query extensions. Both relevance and reliability scores are now set according to the Euclidean distance. In the merging phase, a tight reliability score threshold for using this module's relevance score is defined, so that chances of considering false (near) duplicate images to be excellent matches are small.

2.6. Optical Character Recognition

We also perform Optical Character Recognition (OCR) over the ranking images using Tesseract [13], which is a widely used OCR toolbox. All detected texts are compared with the query text and if the texts are overlapping, the OCR modules returns high relevance and reliability scores.

2.7. Merging Results

The merging algorithm assembles the results of all the modules to determine the final relevance score. The relevance evaluations of duplicate image detector, face bank, and OCR are exploited only when their reliability score is high. Otherwise, the system uses the relevance score from $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ module. Each module has its own relevance score range. The order of the range, from large values to small values, is: duplicate image detector, face bank, OCR, and $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$.

The OCR module gives high reliability and relevance scores if the probing image contains the query text. However, we assume that end users prefer more graphically appealing images to textual dominant images. Thus, we use the OCR text module only if the reliability score of the $\mathrm{P}^2\mathrm{R}^2\mathrm{S}^2$ module is low. Details of the merging algorithm are given in Algorithm 1. The threshold values used in the merging algorithm are empirically determined using the development dataset.

2.8. Secondary Submissions and Other Considered Methods

As recommended by the challenge organizers, in our master submission we evaluated every query-image pair individually without comparing the test images assosicated with a certain query text. In our second submission we tried to exploit the mutual relations of the test images in a simple way. We assumed that, if there is a relevant image to the query text in

Algorithm 1 Merging Algorithm

```
given: relevance and reliability scores of $P^2R^2S^2$ module, face bank, duplicate image detector, and OCR
if duplicate image detector reliability score $\geq$ threshold_1
    return duplicate image detector relevance score
else if face bank reliability score $>$ threshold_2 and facebank relevance score $>$ threshold_3
    return face bank relevance score
else if $P^2R^2S^2$ reliability score $\leq$ threshold_4 and query text is found by OCR
    return OCR relevance score
else
    return $P^2R^2S^2$ relevance score
```

the given query image set, there are probably more images similar to that relevant image. We also assumed that the irrelevant images in each query image set are selected randomly, so the probability of having two similar irrelevant images in a query image set is quite low. Using these assumptions, the $P^2R^2S^2$ module compared the transformed feature vector of each image in the test query image set with feature vectors of the rest of images in the test set instead of comparing it with the feature vectors of training set images as in the master submission. Otherwise the comparison was conducted as explained in Section 2.3. Face bank and duplicate image detector were not changed for this submission. Only the threshold values in the merging algorithm were slightly changed. The OCR module was not applied. Over the development set this method was clearly more successful than our master method (See Section 3.4). However, this method is applicaple only if the test set follows the assumptions given above. It is not a general image retrieval solution and, therefore, we decided not to use it as our master submission. Our third submission was otherwise similar to our master submission, but instead of transforming the OverFeat features using $P^2R^2S^2$ we directly compared the OverFeat features.

We also worked on Learning to Rank. We adopted AdaRank [14] as an iterative ranking algorithm to generate the ranking list over the Bing data. A decision stump acted as the weak ranker, and the data was organized listwisely. We selected 20 iterations for the algorithm to fullfil the discriminative power of our AdaRanker, while avoiding the extensive computation in further iterations. We used NDCG@5 as the measure in training. We performed training using the queries from the development dataset together with their associated training images. The relevance for training each query-image pair was determined based on the similarities between the query text and training text and their click counts. The results obtained in time for MSR-Bing Image Retrieval Challenge were not as good as expected and therefore we keep this method still under development.

3. EXPERIMENTAL RESULTS

3.1. Datasets and Evaluation

We trained $P^2R^2S^2$ module using 100K images and their click counts randomly selected from the training set. We could not use more images for training due to system limitations. We tested our system using the development set, which contains 80K query-image pairs, 1000 queries, and almost 80K images.

We evaluated the performance of our methods using Discounted Cumulated Gain (DCG) measure. To compute DCG, for each query text the images are first ranked according to the relevance scores. DCG for each query is then computed as

$$DCG_{25} = 0.01757 \sum_{i=1}^{25} \frac{2^{rel_i} - 1}{log_2(i+1)}, \tag{2}$$

where $rel_i = \{Excellent = 3, Good = 2, Bad = 0\}$ is the manually judged relevance of the query-image pair. The evaluation metric is computed as the average of DCG scores for all the queries.

3.2. Partial Results on Individual Modules

After the aforementioned text processing procedure, there are only 24 queries in the development dataset and 4 queries in testing dataset that we cannot find any relevant queries from the training dataset.

Using the development set, we computed DCG score over only those queries, where the face bank detected matching faces in the query image and the relevant training images. The average DCG for random guess was 0.5961, while DCG with the face bank was 0.6611. It should be noted here that the face bank was only used for images (1894 cases) where a positive face match was detected. The total number of images paired with these queries was 3103 and the rest of the query-image pairs were still evaluated randomly.

Similar to the face bank results, we evaluated the results obtained using duplicate image detection. There were 42306 query-image pairs considered in this case, but again the output of duplicate detector was used only when a duplicate image was detected (3484 cases). The average DCG for random guess was 0.6533, while DCG with the duplicate image detector was 0.6859.

Also for the OCR module we conducted similar testing. In this case, OCR was used for 143/8654 query-image pairs. The average DCG for random guess was 0.3820, while DCG with OCR was 0.3881. In this case, the result with OCR is within the random score variance (0.0077), but we think that the module has potential to work better on a different test data.

3.3. Final Parameter Settings

Based on experimental results some of which are given here, we set the final system parameters as given in Table 1.

Table 1. Parameter values used in our master submission

Param.	Explanation	Value
Δ	Similarity threshold for duplicate images	0.001
N	Number of sub-feature spaces	100
D	Sub-feature space dimension	25
P	Number of partitions/regressors	20
S	Number of samples per partition	7500
V	Number of principal vectors used	20

3.4. Overall Results

We evaluated several different versions of our system over the development set to evaluate the influence of each part. Beside our official submissions, we afterward submitted some more versions to be evaluated on the test set also. The results are shown in Table 2. Random score was obtained using random guess only, Master, Sub2, and Sub3 results have been obtained using our master, second, and third submission as explained in this paper. OverFeat results are obtained setting the relevance score according to the L1 distance of OverFeat features of the query image and the closest relevant training set image. For OverFeat2 results we set the relevance score according to the L1 distance of OverFeat features of the query image and other test images connected to the same query text. In other words, OverFeat2 was similar to our second submission, but it was not using $\mathrm{P^2R^2S^2}$ to enhance features. $\mathrm{P^2R^2S^2}$ and $\mathrm{P^2R^2S^2 2}$ are similar to our master and second submission, but face bank, duplicate image detection, and OCR are not used. For PCA results we replaced $\mathrm{P^2R^2S^2}$ with principal component analysis. The additional results on the test set show that we would have obtained a better score without using face bank, duplicate image detector, and OCR. These modules or the merging module may have been overfitted for the development set.

Table 2. DCG scores for different versions of our system over the development and test sets

	Random	Master	Sub2	Sub3
Dev. set	0.4704	0.5099	0.5361	0.5006
Test set	0.4858	0.5116	0.5463	0.5044
OverFeat	OverFeat2	$\mathrm{P^2R^2S^2}$	$\mathrm{P^2R^2S^2 2}$	PCA
0.4974	0.5287	0.5082	0.5359	0.4945
0.5037	0.5406	0.5123	0.5473	0.5042

We also evaluated the number of query-image pairs, where each module was given the priority when deciding the final relevance score. The numbers are given in Table 3, where DID is as an abbreviation for duplicate image detector.

Table 3. Number of query-image pairs where each module was used when setting the final score

Dataset	$\mathrm{P^2R^2S^2}$	Face bank	DID	OCR	Random
Dev.	75080	650	3942	142	112
Test	316338	1158	2519	162	1040

4. CONCLUSIONS

In this paper we introduced our solution to MSR-Bing Image Retrieval Challenge @ ICME 2014. Our results show that the proposed method, $\mathrm{P^2R^2S^2}$ can transform OverFeat features into even more discriminative features and enhance the relevance evaluation.

5. REFERENCES

[1] Xian-Sheng Hua, Linjun Yang, Jingdong Wang, Jing Wang, Ming Ye, Kuansan Wang, Yong Rui, and Jin Li, “Clickage: Towards bridging semantic and intent gaps via mining click logs of search engines,” in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 243–252.

[2] George A. Miller, “WordNet: a lexical database for english,” *COMMUNICATIONS OF THE ACM*, vol. 38, pp. 39–41, 1995.

[3] “Hunspell,” http://hunspell.sourceforge.net/.

[4] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *CoRR*, vol. abs/1312.6229, 2013.

[5] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, Jul 2002.

[6] B.S. Manjunath, J.-R. Ohm, V.V. Vasudevan, and A. Yamada, “Color and texture descriptors,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 703–715, Jun 2001.

[7] Jennifer G. Dy, Carla E. Brodley, and Stefan Wrobel, “Feature selection for unsupervised learning,” *Journal of Machine Learning Research*, vol. 5, pp. 845–889, 2004.

[8] Marco Grimaldi, Padraig Cunningham, and Anil Kokaram, “An evaluation of alternative feature selection strategies and ensemble techniques for classifying music,” in *in Proc. Workshop on Multimedia Discovery and Mining*, 2003.

[9] Leo Breiman and Leo Breiman, “Bagging predictors,” in *Machine Learning*, 1996, pp. 123–140.

[10] Mohammad Rastegari, Ali Farhadi, and David Forsyth, “Attribute discovery via predictable discriminative binary codes,” in *In Vision-ECCV*, 2012.

[11] U. Iqbal, I. D. D. Curcio, and M. Gabbouj, “Semi-supervised person reidentification in videos,” in *9th International Conference on Computer Vision Theory and Applications*, Lisbon, Portugal, Jan. 2014.

[12] “Intel Perceptual Computing,” https://software.intel.com/ en-us/vcsource/tools/perceptual-computing-sdk.

[13] Ray Smith, “An overview of the tesseract ocr engine.,” in *ICDAR*, 2007, vol. 7, pp. 629–633.

[14] Jun Xu and Hang Li, “AdaRank: a boosting algorithm for information retrieval,” in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 391–398.