Arbitrary Category Classification of Websites Based on Image Content

Anton Akusok[1], Yoan Miche[1], Juha Karhunen[1], Kaj-Mikael Björk[3], Rui Nian[4], and Amaury Lendasse[1,2,3,5]

[1] Department of Information and Computer Science, Aalto University School of Science, FI-00076, Finland
[2] Department of Mechanical and Industrial Engineering and the Iowa Informatics Initiative, 3131 Seamans Center, The University of Iowa, Iowa City, IA 52242-1527, USA
[3] Arcada University of Applied Science, Helsinki, Finland
[4] College of Information Science and Engineering, Ocean University of China, 266003 Qingdao, China
[5] IKERBASQUE, Basque Foundation for Science, 48011 Bilbao, Spain

Abstract. The paper presents a comprehensive methodology for general image-based classification. It allows for classification of websites with abstract target classes, handles large datasets and is tolerant to noise. The approach uses local image features and their color descriptors to obtain build image representations by a modified k-NN algorithm. Image representations are refined into image and website class predictions by a two-stage classifier based on Extreme Learning Machine. The results are good even in the presence of noise and for abstract target categories; and are excellent for the most important classes.

1 Introduction

Analysis of web content is an old but challenging task, emerged with the first Internet search engines. Being able to describe or classify a webpage is essential for various tasks like returning relevant search results [18], finding similar pages [26] or blocking unwanted or dangerous websites [28] like phishing ones.

Traditional webpage analysis rely on text processing methods [27] (text body of a page, address, keywords and links). But with the increase of bandwidth, storage and processing power, image data found heavy usage in webpages, being a native to humans powerful expressive format. A modern user will probably be surprised seeing a text-only webpage without any visual design.

Image data, while being an important source of information in the web, is hard for machine processing due to its extreme variability. Image understanding is still extremely challenging, but a simpler and more feasible task is image classification with several classes of interest. Existing classification methods include target-specific ones [54], which cannot be generalized on arbitrary classification. Examples of these are adult content detection methods based on the amount of skin colour in the image [39]. Other methods are very elaborate and aim at

image understanding with object extraction and recognition [50, 11]. They are common in regular competitions like PASCAL VOC [16] or common benchmark datasets like Caltech101 [17]; otherwise they are impractical due to a complicated adaptation to another problem, a lack of ready-made toolboxes, and long training and running times.

This work aims at a web content filtering problem. This is an integral part of an automated Internet security framework. Internet security domain is not new, but the need of its automation is recent. While computer networks are penetrating into all parts of human activity, and the potential danger of their misuse raises (up to a physical treat imposed by, for instance, hacked computer-based elevator control systems or car safety systems), a demand for capable safety measures remains high. Due to a large volume of information produced every day (both content and malicious software), human experts are unable to respond to every case in time. One possibility to address this task is to pre-process data automatically, giving machine decision for easy or well-known cases, and leaving much lower number of cases for human analysis. This work addresses dangers or inconveniences that web content of some kind may give to people, by creating a tunable web content filtering system. The work is done in collaboration with F-Secure Corp.[6] under Data-to-Intelligence[7] (D2I) TeKeS project, and is partially supported by a Nokia Foundation grant[8].

Web content filtering is done by estimating whether a particular website belongs to one of the offensive classes, and blocking it if it does. The scientific part of the method is thus a multi-label classification of websites. The available dataset has 20 target classes — 19 offensive ones, which reflect concepts like "Hate" or "Beer", and a benign class called "Unknown". A large dataset of URLs (web addresses) for these classes is provided by F-Secure Corp. A list of classes with the number of corresponding urls is given in Table 1.

A typical approach for classification of web pages is a text-based classification. This is done indeed on the same dataset (by a different party[9]), but in the context of the problem, text-based approach has numerous drawbacks. First, it performs poorly on websites with little or no text; but such websites are present in important categories like "Adult". Second, a text-based classifier is learned for one language. It requires adaptation to cover multiple languages, which is problematic as the number of languages is high, and for many of them there is little training data. An image-based classifier could overcome these difficulties, by being invariant to the language of website and able to use training data from all languages altogether.

Image-based classifier proposes its own challenges, which are considered in this work. First, correct class labels are known only for websites. An assumption is made that all images in a website have the same label, which obviously does not always hold. Images which have different class than their website are called

[6] F-Secure `http://www.f-secure.com/en/web/home_global/home`
[7] Data to Intelligence `http://www.datatointelligence.fi`
[8] Nokia Foundation `http://www.nokiafoundation.com`
[9] Åbo Akademi `http://www.abo.fi/institution/en/informationsteknologi`

Table 1: Details of the dataset of website urls, provided by F-Secure Corp.

Class	Websites	Images
Adult (english)	6801	216446
Beer	5913	33331
Casino	3651	22513
Cigarette	1939	13734
Cigars	3845	25224
Cults	3282	17976
Dating	4703	32091
Jew related	3479	16696
Marijuana	5397	43980
Occults	5105	20139
Prescription drugs	6042	42433
Racism white supremacy	400	2794
Racist groups	4667	29626
Religion	5438	18092
Spirits	2820	18962
Sports betting	3671	16569
Violence	1919	21344
Weapons	2464	27629
Wine	4095	25463
Unknown	3432	34709

semantic noise. The proportion of semantic noise varies significantly per class: the "Adult" class has relatively low amount of irrelevant images, about 5%; and web pages of the "Cults" class mostly contain relevant text and lots of irrelevant images, like avatars of people talking at forums, with a proportion of image semantic noise close to 70%.

Image understanding and machine vision is an active research frontier in Machine Learning, however very few problems are successfully solved so far. The next section 2 discusses about modern image classification, image retrieval and object recognition methods. It describes typical approaches to image-based machine learning, and highlights those which form the image processing core of the proposed website classifier. Additional steps of False Positives-optimized classification and merging of image predictions into website labels are discussed in the corresponding sections 3.4, 3.5 of the methodology.

2 Related work

Extracting semantic information from image data is an active research frontier [14, 41]. It includes topics like image classification, retrieval or segmentation; object detection in images; image labeling; and video processing as well. Problems connected with image processing are hard to solve, partially because most images are 2-dimensional projections of a 3-dimensional world, and successful

usage of images by humans is based on an extensive use of prior and context knowledge.

The authors are aware of only two complex image-related tasks, which have satisfactory solution so far. One is a face detection method, which is found in most modern smartphones and digital cameras. It is based on the Viola-Jones face detector [51], which scans an image with a sliding window at different scales. It uses an ensemble of simple classifiers, selected and trained on a large dataset of labeled faces. That sliding window, being a slow method in general, achieves extreme speedup by using integral images [12] with its simple classifiers, which allows for a real-time performance even on relatively slow devices.

The second problem with an accurate solution is the road signs recognition [32]. The approach is similar — an image is scanned with a sliding window of different sizes and shapes, and each window is processed separately by a neural network. Without rolling sum trick, the performance is far from real-time even with accelerated computations on GPU, but the method's accuracy exceeds 99%.

Both these successful cases use the prior knowledge of objects they aim to find: faces or road signs. Another interesting research classified 100,000 classes reasonably fast [14], but again they represent typical objects mostly in canonical views. This will not hold for a general-purpose image classifier of object detector. One well-known annual competition for general image-based classification and detection methods is PASCAL VOC [16]. Other popular publicly available complex image datasets include Caltech-101 [17], Caltech-256 [19], MIRFLICKR-1M [25] and 80 million tiny images [45] dataset. The state-of-the-art methods are found among the winners of PASCAL VOC [11] or recent publication on the other image datasets [2]; links can be found on corresponding websites.

The developed algorithms are large and complex (see [2, 50, 15]), but most of them utilize the same basic building blocks — local image features ("local features" or "image features"). The idea is that useful information in image is not distributed uniformly, and some parts (i.e. corners, see Figure 1) are more important for image understanding than others (i.e. a uniform background).

The process of obtaining local image features involves two major steps: feature detection and feature extraction. The detection phase finds potential informative regions in image, a good overview is given in [46]. Feature detection phase can be skipped with dense image sampling, but this method results in more local features, and not all of them are useful. Common feature detection methods are Harris-Laplace [37] and Harris-Affine [36]; the former is scale- and rotation-invariant and the latter is also affine-invariant, although detects less stable features across different images of a same scene.

In feature extraction stage, pixel values of a found image patch are transformed to a special fixed-size descriptor vector. Its major property is that two such vectors calculated from two similar patches on different images (two patches of the same object part, which look similar to humans, but may have very different pixel values) lie close in Euclidean space [31]. This property allows matching descriptor vectors for detecting similar objects across images, for instance pedestrians [35, 4]. Different methods of calculating descriptor vectors produce features

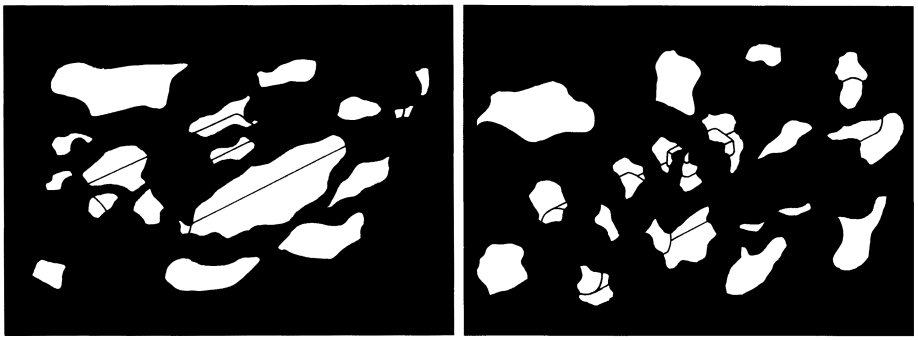


Fig. 1: Local image features: non-informative (left) and informative (right). Informative features capture corners and junctions, while non-informative represent lines or uniform image regions. An object (flashlight) is recoverable with informative image features. Figure from a human image understanding paper [5].

of various dimensionality, but in general it should be large enough for accurate matching [8]. Descriptors are calculated for intensity maps, which are gray-scale images in general; color descriptors are possible by concatenating particular descriptors from intensity maps of colors in different color spaces [47]. Common choices of descriptors are different variants of SIFT [31, 47] and SURF [3], the latter is less precise but much faster. Modern development train descriptors with Convolutional Neural Networks to obtain state-of-the-art performance [41].

General image classification frameworks often use search over image to produce confidence maps [11]. This works in class-specific methods, like search for naked body in Adult images detection [52, 39]. Such methods often use Bag-of-Visual-Words (BoW) approach [44, 21, 2], similar to Bag-of-Words in text processing [6]. The occurrence histogram produced by a BoW method is useful for an SVM classifier with a histogram kernel [10]. A different way is a per-feature analysis [9], based on k-NN feature matching [1]. This approach can match different parts of an image to different training images [7], which is useful for learning with abstract non-uniform classes.

The next section 3 introduces the proposed methodology for learning abstract image classes from large and noisy dataset. It uses color SIFT image features, and a per-feature kNN-based image classification. Predicted image classes are merged together for websites, as websites are the targets for classification in the imposed problem.

3 Image-based classification methodology

Overview of methodology

The methodology is created from an idea of meaningful local regions in images, some of which are relevant to the classification problem [7], as illustrated on

Figure 2. It starts by extracting all local feature descriptors from training images, which inherit the class of their corresponding image. These features define a class distribution in the descriptor space. The first step creates a large set of classified local features (200,000,000 features for the proprietary dataset from F-Secure Corp.).

Next, the created dataset is used for classifying feature descriptors of new images. Classification is based on the Nearest Neighbor idea, that the class of a test feature is similar to the classes of its nearest neighbors. Calculation of the exact nearest neighbor with 200,000,000 samples is infeasible, so a smaller representative subset of training features is used to define the class distribution in the feature space. Samples in this subset are called "centroids" for convenience, in resemblance to the centroids in k-means clustering. These centroids are classified once on a full training set (which takes thousands of core-hours in the local Triton[10] computer cluster). Then several closest centroids are found for each test local feature, and the class description of a test feature is derived from the classes of these centroids. Details are explained in subsection 3.3.

With the classes of test local features, test images can be classified by a general method. Authors use an Optimally Pruned Extreme Learning Machine (OP-ELM [33]), a variation of Extreme Learning Machine [24, 23] (ELM) model which suits well for large-scale machine learning problems [48]. This particular method is adjusted to output confidences in multiple classes, with a possibility to reject image if a confidence level is not high enough. The rejection is important for this particular problem, because low False Positives rate is preferred over a high coverage in the given problem setup: missing an offensive website label sometimes is acceptable, because the method is not perfect; but blocking a benign website can make a user to remove website filtering tool, which is an unwanted situation for the company.

The last step is the classification of websites. It is based on class predictions of images from each website. Such predictions are not binary, but are continuous values where a higher value means higher class likelihood. An assumption is made that only a few of available classes are correct for one website. A hypothesis is proposed that predictions for irrelevant classes for images of that website will be distributed normally with a low mean value, while predictions for correct classes for these images will not come from the same distribution, as they would have a significantly higher value. That hypothesis is tested by a paired t-test with adjustable threshold between mean values of the two normal distributions, and results in a multi-label website classification.

Each major step 1-5 of the methodology from Figure 2 is described in the corresponding sub-sections below.

[10] We acknowledge the computational resources provided by Aalto Science-IT project.

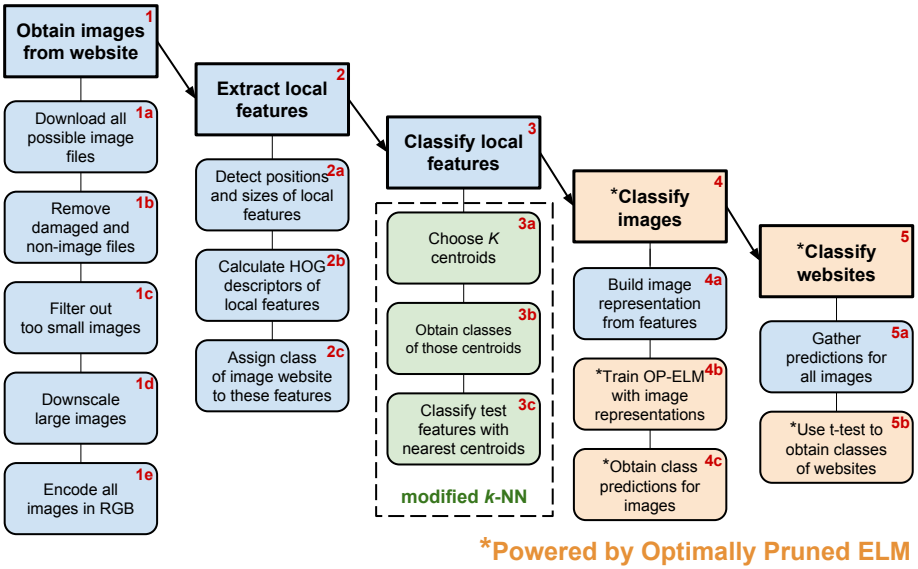


Fig. 2: Diagram of image classification process. Five major steps are given in bold rectangles, with corresponding sub-steps depicted below. See explanation in the text.

3.1 Obtaining images from websites

An input to the methodology is a single website url (and a class of that website for the training set). The data collection stage (Figure 2, 1a-1e) transforms website image data into a uniform representation.

A website url is searched for any files that have image extensions. The list of useful extensions is taken from an image processing toolbox for Python language (Python Image Library, PIL[11]), and includes .bmp, .dib, .dcx, .gif, .im, .jpg, .jpeg, .jpe, .pcd, .pcx, .png, .pbm, .pgm, .ppm, .psd, .tif, .tiff, .xbm and .xpm.

Then any damaged or non-image files are deleted. Image files can be detected by a file header, but this does not detect damaged images (due to errors during download, incorrect encoding, or something else). An easy way of finding valid images is to load image content from all files with a toolbox. If an error occurs at any stage, the file is deleted as an invalid image.

Downloaded image content from a website often includes decoration elements like lines or uniform background. They are unlikely to convey any class-relative meaning, and can introduce class noise with their class predictions, thus they are discarded in the methodology. An empirical threshold on minimum file size of a meaningful image is estimated by manually browsing a large collection of downloaded images sorted by file size. This threshold is fixed at 2400 bytes.

[11] `http://www.pythonware.com/products/pil/`

Most image files smaller that 2400 bytes are either decoration elements, or tiny previews of other images, and can be safely discarded.

Another problem is posed by images of a large size. One example was found in a rasterized vector image with dimensions of 6400x6400 pixels. It contained many sharp edges, and produced over 100,000 local features, significantly slowing down the whole method without any benefits in accuracy of predictions. Thus an upper bound in the longest edge is set for all images. The current system sets this bound to 500 pixels; the value can be adjusted for smaller runtime or better performance, but not having such value at all hurt the performance. Down-scaling of images which exceed upper size boundary is done with an anti-aliasing algorithm, with keeping original image proportions.

The final step of getting image data is to encode all images in RGB colorspace. This step is performed on all images, even the ones which are already in RGB — because they often include an alpha channel, which makes them different for the feature extractor. A low compression level is chosen for JPEG algorithm to prevent occurrence of visual artifacts.

An important remark to say is that although website images are pre-processed as explained before, some of them will be irrelevant to the problem. These images form *semantic noise*, which is empirically estimated to range from 5% for "easy" classes like *Adult* up to 70% for "difficult" classes like *Cults*. In "easy" classes like *Adult*, most information is presented by picture and video content, a favorable case for an image-based methodology. In "difficult" classes, however, information is transferred mostly in textual form. For instance, a major part of images from the *Cults* category is formed by avatars of people who chat on forums. Such images don't help much in classification; but an abundance of textual information makes text-based classifiers good complimentary methods for these areas. Highly variable quality of initial image content is an unavoidable difficulty for image-based methods, that is why a rejection option for low-confidence predictions is added to final classifiers of the proposed methodology.

3.2 Extraction of local image features

Local image features are "meaningful", or "informative" regions of an image [30, 3]. Thinking about a picture of an airplane in the sky, a patch of a uniform blue sky is not very informative, whereas a patch containing an airplane is. Local features usually contain corners, edges or strong changes in color and contrast [46]. These features are specifically made to be invariant to image transformations (scaling, rotation) and noise (for instance from different encoding), so they are useful for finding similar objects in different images [31, 38, 49]. These features can also be used for image classification, by finding image patches similar to those from some particular class [7].

Harris-Laplace image feature detector

Edge and corner detection in image uses derivative (or gradient) of image intensity map, usually denoted by $\mathbf{I}$. Edge detection is straightforward, as edges

correspond to local maximum in matrix **I**. Smoothing with Gaussian kernel suppresses noise and makes edges easier to detect.

Corner detection is more elaborate. A corner is an area of image with high curvature. It corresponds not only to actual corners, but also to junctions, highly textured surfaces, occlusion boundaries, etc. A continuous *cornerness* parameter is used to specify the likelihood of an image pixel to be in the center of a "corner". In general local image features framework, meaningful areas have high cornerness, so a corner detector is used as local image feature detector.

Mathematically, corners are found by analyzing two orthogonal image gradients. Low values of both gradients correspond to a uniform area with low cornerness and informativity. One large gradient and its low counterpart correspond to edges; and both large gradients correspond to corners. Two orthogonal gradients for pixel neighborhoods are found with the second derivative matrix **M**: they are given by eigenvalues of this matrix, and their scale is given by corresponding eigenvectors.

$$\mathbf{M} = \sigma_D^2 g(\sigma_I) * \begin{bmatrix} \mathbf{I}_x^2(\mathbf{x}, \sigma_D) & \mathbf{I}_x(\mathbf{x}, \sigma_D)\mathbf{I}_y(\mathbf{x}, \sigma_D) \\ \mathbf{I}_x(\mathbf{x}, \sigma_D)\mathbf{I}_y(\mathbf{x}, \sigma_D) & \mathbf{I}_y^2(\mathbf{x}, \sigma_D) \end{bmatrix}, \tag{1}$$

where

$$\mathbf{I}_x^2(\mathbf{x}, \sigma_D) = \frac{\partial}{\partial x} g(\sigma_D) * \mathbf{I}(\mathbf{x}) \tag{2}$$

$$g(\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{3}$$

where $\mathbf{x}$ is a query image, $\mathbf{I}(\mathbf{x})$ is an intensity map of that image, x and y are x-axis and y-axis of the pixel grid of an image. Image gradients (local image derivatives) are computed with the *differentiation scale* σ_D. Then the gradients around an image point are smoothed with the *integration scale* σ_I.

Because directions of gradients are irrelevant to cornerness, and only their magnitudes matter, there is a simplified approach for detecting high cornerness. Recall that matrix determinant is equal to a product of eigenvalues, and the trace equals their sum. Cornerness can be estimated as a difference between determinant and trace of matrix M.

$$\text{cornerness} = \det(M) - \lambda \operatorname{trace}(M) \tag{4}$$

The typical value for λ is 0.04 [46]. This formula is faster than finding the exact eigenvalues. Process of estimating cornerness is shown on Figure 3.

The *characteristic* scale of found corners is detected by an extremum of some scale function. The size of the region obtained that way is independent of the image resolution, as illustrated on Figure 4. Scale detection in Harris-Laplace method is initialized by running the Harris corner detector multiple times on different scales, and then selecting an optimum region among interleaving ones between multiple scales with the Laplacian operator, as explained in details in [29].

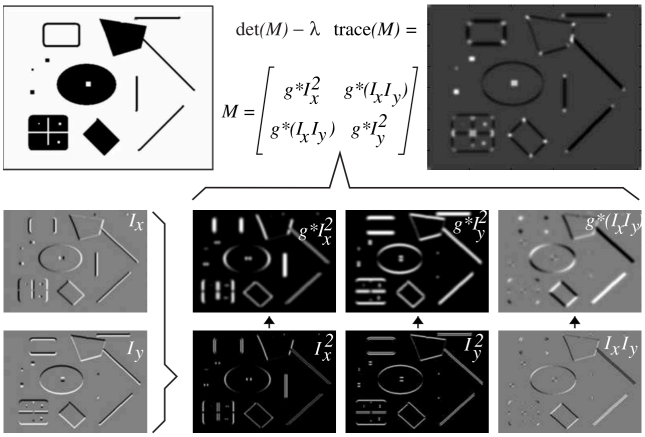


Fig. 3: Estimating cornerness with second derivatives of intensity matrix M. Illustration from [46].

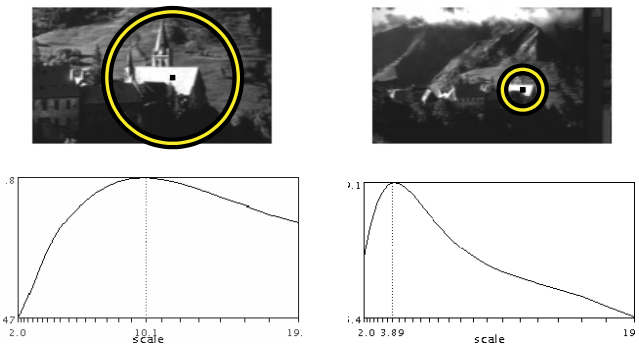


Fig. 4: The characteristic scale calculated with the Laplacian operator, and the corresponding regions, for two images of the same scene at different resolution. Illustration from [46].

Finally, the detected corner regions with corresponding locations, scales and orientations are considered to be the local informative regions of an image. The main benefit of the Harris-Laplace detector is a large portion of similar regions found on images with the same scene but different resolutions and noise levels [38] (for instance from different encoding). The detected regions are still composed from very different pixel values, and an encoding method is required for their comparison, which is invariant to noise and particular pixel brightness. One such commonly used method, called SIFT (Scale-Invariant Feature Transform) is described below.

SIFT image feature descriptor The Scale Invariant Feature Transform [30] is a local image feature descriptor, based on histograms of oriented gradients [13] (HOG). On its input, SIFT has position and size (scale) of a local feature. The image feature orientation is defined as the orientation of an average gradient of that image patch.

The SIFT method starts by placing a 16×16 regular square grid at an image feature, aligned with that feature's orientation. Then local gradients are calculated for each cell of that grid, based from values of pixels inside the grid cells. Magnitudes of those gradients are weighted with a Gaussian kernel to give more importance to gradients near the center of the feature. Then these weighted gradients are accumulated into 4×4 orientation histograms with 8 discrete orientations each. Values of histograms are obtained as vectors by reading them counter-clockwise. Finally, vectors for each histogram are concatenated to get the whole SIFT descriptor vector. Its length is thus 16 histograms $\times$ 8 directions $= 128$ features. An example of SIFT descriptor calculation is shown on Figure 5.

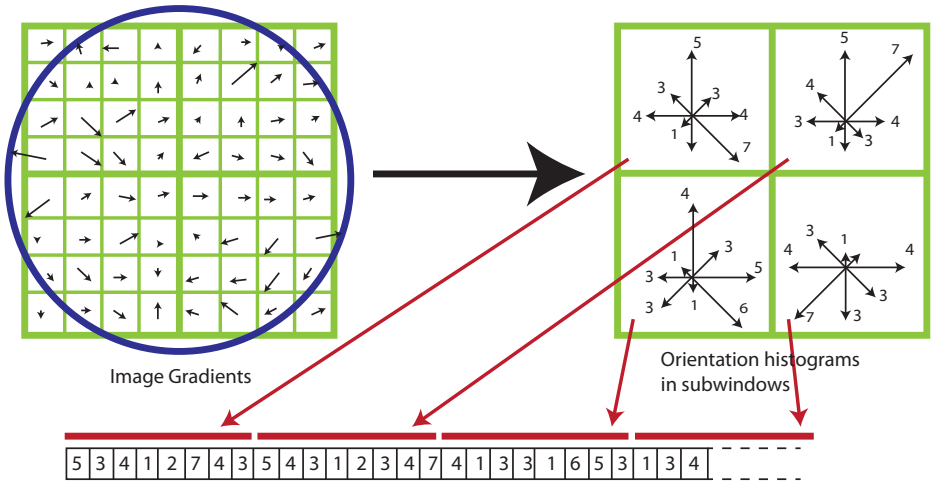


Fig. 5: Example of SIFT descriptor calculation; reduced dimensionality is used for better visibility. First, local gradients are obtained for each cell of 8×8 grid (16×16 in full SIFT). These gradients are weighted with a Gaussian kernel, denoted by a circle. Then an oriented histogram is calculated on 2×2 grid (4×4 in full SIFT); a length of each arrow in a histogram equal to the sum of gradients in the same direction. Finally, histogram values are read counter-clockwise and concatenated into a SIFT descriptor.

The original SIFT descriptor uses only image intensities, and is suitable for gray-scale images. Color SIFT descriptors are obtained by concatenating particular SIFT vectors for each color in some color space. Recent surveys [47, 8]

suggest that a weighted opponent color space (cSIFT descriptor) is a good choice. This type of image feature descriptor is used in the methodology.

The next two subsections introduce the method of utilizing cSIFT image descriptor for image classification. It is based on k-Nearest Neighbor classification of image descriptors, so that a test image is compared to whole training classes, not on an image-per-image basis.

3.3 Nearest Neighbor classification of local features

A major assumption behind the proposed image-based classifier is that local image features convey class-related meaning. That is, there is a distribution of image classes in the local feature space (384-dimensional space for cSIFT features). This distribution can be estimated, and allows for the classification of local features of a test image.

A suitable method of estimating the class distribution is k-Nearest Neighbors (k-NN). For each local features of a test image, the k closest features from the training set are found using a space metric (L_p norm is a common choice, with $p = 1$ or $p = 2$). An estimated class of a test feature is obtained by, for instance, a majority vote between classes of k closest training features.

Unfortunately, an exact k-NN method is infeasible with the amount of features in the training set (200,000,000 for F-Secure dataset). The feature dimensionality of 384 makes approximate nearest neighbor computations ineffective as well [53]. Thus a reduced version of k-NN method is used, where all features of the training set are represented by their smaller representative subset. Samples of that subset are called *centroids*, as they are often found as centroids of a k-means clustering algorithm with a large number of clusters. These centroids with corresponding classes store the information about class distribution in the local feature space. In practice, centroids can be selected randomly from a dataset. The effect of k-means versus random selection, and the optimal number of centroids are discussed in the Experiments section 4.1.

Given a smaller set of centroids, it is feasible to find the closest k centroids of each class for each local feature of a test image, with corresponding distances. Then for each class, distances to the closest centroids are pooled together for all the local features of an image. Three parameters are extracted per class: the smallest distance, the average distance and the standard deviation of all distances. It produces 20 classes * 3 parameters = 60 numbers. These 60 numbers create an image representation vector, which has the same dimensionality for images of any shape and with any number of local features. Feature vectors of images are used in general purpose OP-ELM classifiers, as explained in the next section.

3.4 Optimally Pruned ELM for image and website classification

The previous step provides labels for particular local features, and image labels can be inferred from them by a simple majority vote. But such approach discards possible interactions between classes, and the effect of an uneven number of

training samples per class. Also, an increased tolerance against False Positive predictions is desired for the specific problem, even at the cost of decreased coverage. All these constraints are satisfied by a general classifier algorithm. The algorithm of choice in the article is a modification of an Extreme Learning Machine.

An Extreme Learning Machine (ELM) is a batch training method for a Single Layer Feed-forward Neural network (SLFN). It was initially proposed by Guang-Bin Huang in [24, 23], and is proven to be a universal approximator [22]. An ELM network is trained in two steps. First, weights and biases of the input-to-hidden layer are initialized randomly. Then input samples are projected to the hidden layer using these random weights, and a non-linear function is applied to the hidden layer outputs. Second, the hidden-to-output weights are found from a linear equation system between hidden layer data representation and the corresponding target outputs. The solution of such over- or under-determined linear system in a general case (where number of hidden neurons may not be equal to the number of samples) is obtained with an ordinary least squares method. Such solution for a back-propagation Multilayer Perceptron [20] is impossible because of the dependence between input-to-hidden and hidden-to-output layer weights; thus a back-propagation MLP uses iterative gradient descend method. Ordinary least squares solution is several orders of magnitude faster, with a comparable accuracy [33], and has the only parameter to tune: a number of hidden nodes.

A formal description of ELM is following. Consider a set of M distinct training samples $(\mathbf{x}_i, \mathbf{y}_i),\ i \in [\![1, M]\!]$ with $\mathbf{x}_i \in \mathbb{R}^{d_1}$ and $\mathbf{y}_i \in \mathbb{R}^{d_2}$. Then a SLFN with N hidden neurons has the following output equation:

$$\sum_{i=1}^{N} \beta_i \phi(\mathbf{w}_i \mathbf{x}_j + b_i),\ j \in [\![1, M]\!],$$

with ϕ being the activation function (a hyperbolic tangent is a common choice, but a mixture of non-linear and linear activation functions is possible), $\mathbf{w}_i$ the input weights, b_i the biases and β_i the output weights.

In case where the SLFN perfectly approximates the data, the relation between inputs $\mathbf{x}_i$ of the network, target outputs $\mathbf{y}_i$ and estimated outputs $\hat{\mathbf{y}}_i$ is:

$$\sum_{i=1}^{N} \beta_i \phi(\mathbf{w}_i \mathbf{x}_j + b_i) = \hat{\mathbf{y}}_i = \mathbf{y}_i,\ j \in [\![1, M]\!],$$

which writes compactly as $\mathbf{H}\beta = \mathbf{Y}$, with

$$\mathbf{H} = \begin{bmatrix} \phi(\mathbf{w}_1 \mathbf{x}_1 + b_1) & \cdots & \phi(\mathbf{w}_N \mathbf{x}_1 + b_N) \\ \vdots & \ddots & \vdots \\ \phi(\mathbf{w}_1 \mathbf{x}_M + b_1) & \cdots & \phi(\mathbf{w}_N \mathbf{x}_M + b_N) \end{bmatrix},$$

$$\beta = \left(\beta_1^T \cdots \beta_N^T\right)^T, \qquad \mathbf{Y} = \left(\mathbf{y}_1^T \cdots \mathbf{y}_M^T\right)^T.$$

The general case solution for the output weights β with under- or over-determined linear system and imperfect approximation is achieved through the Moore-Penrose generalized inverse [42] of the matrix $\mathbf{H}$, denoted as $\mathbf{H}^\dagger$. This solution minimizes the L_2-norm of the approximation error.

$$\beta = \mathbf{H}^\dagger \mathbf{Y}$$
$$\hat{\mathbf{Y}} = \mathbf{H}\beta = \mathbf{H}\mathbf{H}^\dagger \mathbf{Y}$$

Optimally Pruned ELM (OP-ELM) [33] is an extension of the ELM method, which adds selection of hidden neurons. In practice, any regularized or pruned versions of ELM are desired because they counter possible negative effect of random initialization, which may create hidden nodes with outputs irrelevant to the task. OP-ELM is one of such methods, and it is used in the proposed methodology.

Optimal pruning is done after the second stage of ELM learning (solving a linear system). It is based on a Multi-Response Sparse Regression algorithm [43], which provides ranking of linear model input variables with respect to multiple outputs, from most to least useful ones. The selection of the best number of most useful neurons uses a Leave-One-Out (LOO) estimator of the generalization error, conveniently calculated with PRESS statistics formula [40]. Only the selected number of most useful neurons is retained, the rest hidden nodes are discarded. The procedure reduces noise effect from random initialization, slightly improving prediction quality but greatly improving stability of predictions between different initializations and reducing over-fitting from a too large number of hidden nodes. It also effectively removes the "number of hidden neurons" model parameter, as any number large enough will suffice. See [33] and [34] for more details.

3.5 Combining image predictions to website

The OP-ELM classifier provides prediction for single images. For website classification, these predictions are combined together. Another goal is a multi-label website classification, because the target labels are not mutually exclusive — for instance, the same website can belong to "Spirits" and "Cigarette" classes.

The combination uses a fact that most of the 20 classes will be negative for a given website. The image classifier can predict all negative classes (if website images are non-informative), or one or more classes can be predicted as positive. Per-image prediction of a positive class may vary, and will always have the top confidence (ELM output) value. However, among all website images, predictions for a positive class have a significantly higher average value than predictions for a negative class. This significance level is evaluated formally with a t-test.

An example is shown on Figure 6. Frame (A) shows classifier outputs for a website with 5 images, and 3 possible classes. The exact output values are in part random, so a normal distribution can be fit to them, as on frame (B). If

one class is positive, then a normal distribution fitted to that class will have significantly higher mean than a normal distribution fitted to all other classes, as on frame (C). The frame (D) shows a fitted distribution of a negative class, which does not have a significantly higher mean value.

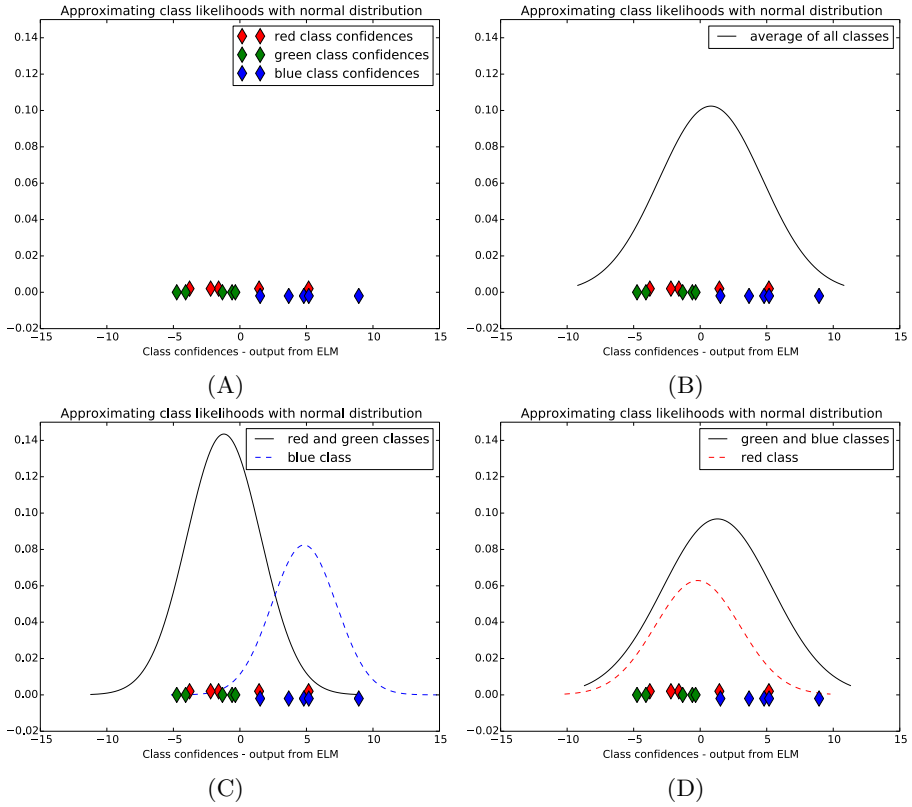


Fig. 6: Example of using t-test from multi-label classification of a website, with 5 images and 3 classes (red, green and blue). A) Classifier outputs for all images and all classes (small y-axis variation added for visibility). B) All outputs can be approximated by a single Gaussian distribution. C) Example of a positive class (blue) for a website - a normal distribution fitted to the blue samples has significantly higher mean than a normal distribution fitted to all except blue samples. D) Example of a negative class (red) for a website.

The t-test makes a parametric website classifier, because minimum thresholds between global mean and particular class means must be found for multi-label classification (see Figure 6, C). A simplified classifier is obtained by another OP-ELM, which uses mean and standard deviation of samples of each class as inputs. One additional useful input is the number of images in a website, which gives

41 input features in total. Using mean and standard deviation values imply the normality of data distribution, and gives a similar method to t-test; but all inner parameters are evaluated automatically from the training set. For a multi-label classification, only threshold on the OP-ELM outputs is to be evaluated; and for a simpler multi-class classification there are no parameters at all, because the largest output is taken as the predicted class. Such simplified multi-class classifier is used in the experiments.

4 Experiments and Results

4.1 Selecting a number of centroids

Image classifier is based on a class distribution in the local feature space. Due to infeasibility of an exact k-NN, only a subset of all training local features is used; samples in this subset are called *centroids*. Centroids can be selected randomly, or taken as centroids of k-means clustering with a high number of clusters. Centroids need to have class; this can be a class of an image they are taken from (for random centroids), or it is selected with a majority vote among k-Nearest Neighbors from training local features. A comparison between random and k-means centroids, with different centroids classification methods, is given on Figure 7. For the experiment, images from the Caltech-101 dataset are used because classes of Caltech are well-defined and certain.

As figure shows, local feature classification performance grows for both training and validation set as the number of centroids increases. It is probably limited by an exact Nearest Neighbor approach, which is computationally infeasible except for toy data. Among centroids selection method, k-means outperforms random selection on the training set, but on a validation set the accuracy with k-means grows slower for large numbers of centroids. Taking into account a significant computational cost of obtaining k-means centroids even on the given toy example with 53000 training local features (not to mention the 200,000,000 local features of the realistic web images dataset), the random sampling centroids selection method is chosen.

The best method for centroids classification also depends on the experiment scale. At less than 4000 centroids, re-calculating the class of centroids with a majority vote between k-NN of a training set improves accuracy. Larger values of k ($k = 201$ on the figure) provide better accuracy than smaller values ($k = 3$ or $k = 1$). This approach is often used in conjunction with bag-of-visual-words and SVM classification in the literature [2]. However, the best classification accuracy is reached with a high number of randomly selected centroids, using original classes of corresponding local features. The original centroids' classes and a high number of randomly chosen centroids are therefore used in the experiments.

4.2 Classification results

The experiments are performed on a large dataset of images and websites, provided by F-Secure Corp., with 20 target classes (19 offensive and one benign,

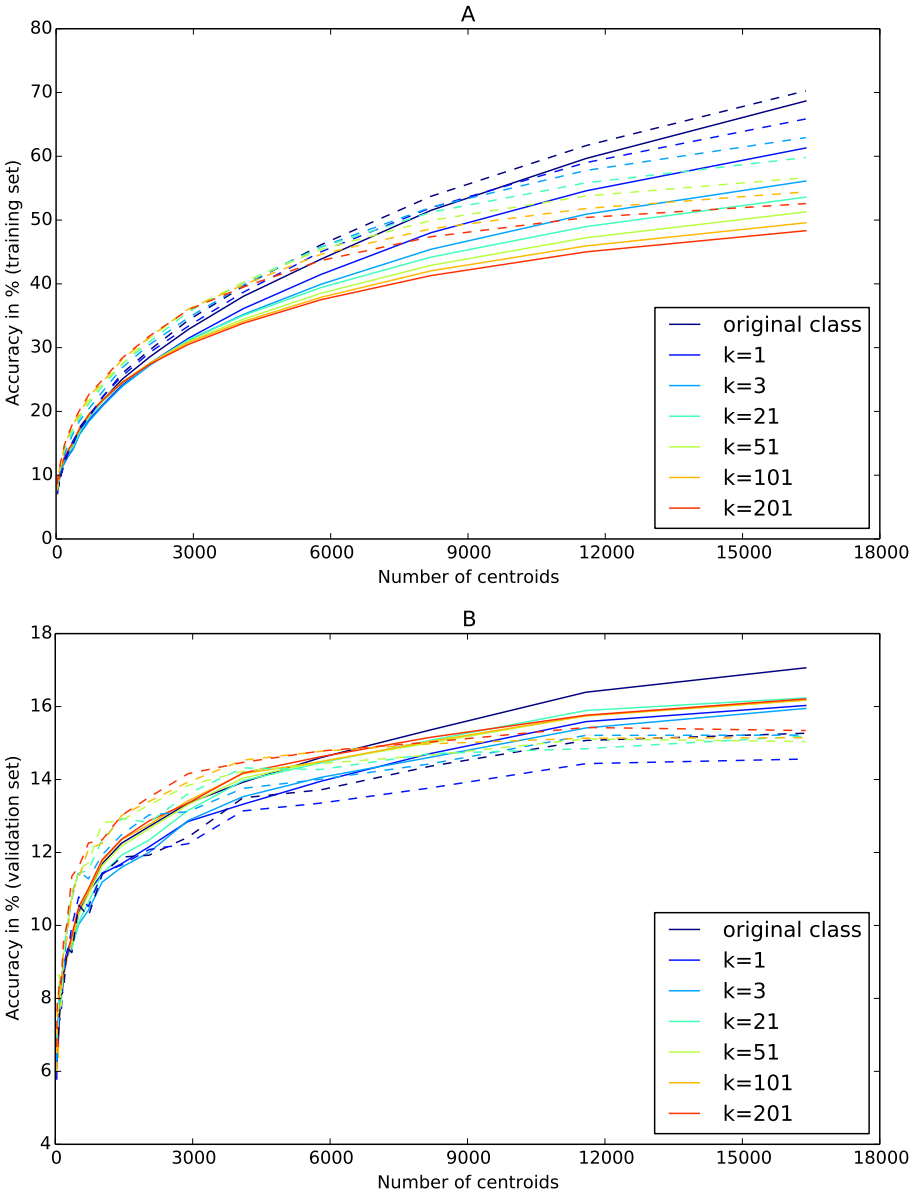


Fig. 7: Classification accuracy of each local image feature, for a training set (A) and a validation set (B), using 1-NN on a set of centroids. Solid lines correspond to randomly chosen centroids, dashed lines to centroids obtained by a k-means algorithm (initialized by the same randomly chosen centroids). Dark blue label correspond to original class of local features, which are chosen to be centroids. Other colors correspond to centroids, classes of whose are obtained by the majority vote between k-NN on all local features from the training set, values of k shown on the legend. Dataset used is 20 classes of Caltech-101 with 5 images for training and validation, around 53000 training local features in total.

called "Unknown"). For each run, 2000 images per class (40000 total) and 200 websites per class (4000 total) are selected for training, and 10000 images per class (200000 total) and 1000 websites per class (20000) for validation. Images and websites are split randomly into training and validation set; the "Racism white" class has less validation images and websites due to a smaller amount of available data. All experiments are performed 10 times, and averaged results are reported.

Centroids are selected randomly from the available image descriptors. There are 2^{20} centroids in total, equal amount from each class (roughly 53000 per class). Prediction accuracy is tested with different amount of centroids, and a smaller set of centroids is always a subset of larger ones. The maximum number of centroids 2^{20} is limited by the computational complexity of processing the dataset (more than 10,000 core * hours for the given number of centroids), and the practical limitation on runtime for website class predictions at F-Secure Corp. (target runtime is less than one minute per website).

The final results for image and website classification are presented on Figure 8. Classification performance grows steadily as the number of centroids increases, possibly reaching the maximum with an exact k-NN (all training local features are centroids). The benign class is especially hard to predict, because it includes all possible images except from the other 19 offensive classes, and thus has a high intro-class variability. Mis-predicting a benign website as an offensive is effectively a False Positive in the website classification methodology, because it will deny access to a good website and can make a user unhappy about the web filtering tool.

4.3 Classification analysis

Confusion matrices of the best experimental setup for website and image classification are shown on Figure 9. Image classification performance is good for per-image prediction, but it becomes even better when images from one website are processed together by another ELM. For image classification, some classes like benign ("Unknown") and "Prescription drugs" get 1%-5% of samples misclassified as other classes. Combining image predictions to websites reduces these errors.

One of the major mis-classification are "Unknown" images predicted as "Adult" (5.2%). These images are checked — several of them are really adult ones (maybe from advertising on websites related to adult), the others are shown on Figure 10 (A). The system could indeed match some parts of these images to parts of images of the "Adult" class to make mis-classification. A case of minor misclassification of "Cigarette" images as "Adult" is checked for another example (Figure 10, B).

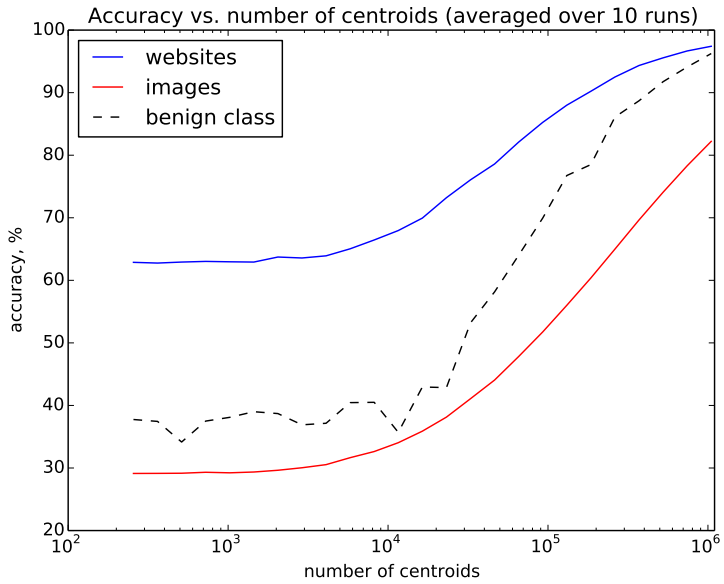


Fig. 8: Website, image and benign websites classification accuracy for the F-Secure Corp. dataset. Number of centroids is given on a logarithmic scale, from 256 to 2^{20}. Images and websites accuracy is averaged over all 20 classes; results are for validation. Random guessing accuracy is 5%.

4.4 Runtime

With a high amount of centroids, time of calculating the distances between image descriptors and all centroids dominates the program runtime. With 384 features in image descriptors, approximate nearest neighbor methods do not work (they are ineffective for more than 20 features in the data [53]). An average image has 300 local features. The build-in Python language function (`cdist()` from `scipy.spatial.distance` module) computes distances between 300 local features and 1050000 centroids in 155 seconds. A custom C language implementation, which uses the 8-bit integer numbers (features are represented as 8-bit integers) and runs in parallel on a quad-core i7 CPU, finishes in 17 seconds. Even faster implementation is possible with accelerators like GPU cards. Finding closest k centroids is a fast task without full sorting (see `partition()` function in `numpy` Python module). Extracting local features from an image takes on average between 1 and 5 seconds. Computational time of distance calculation is linear in the number of centroids.

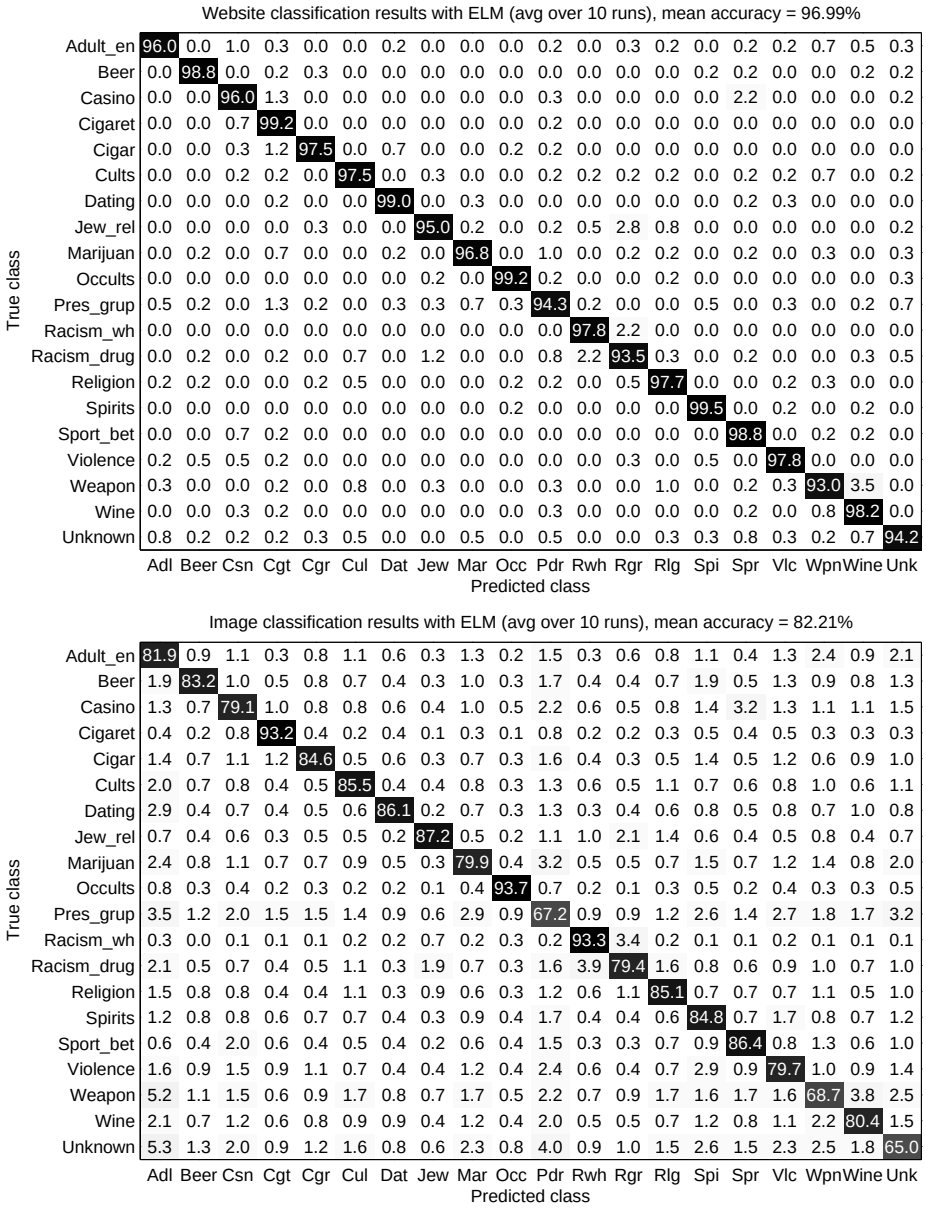
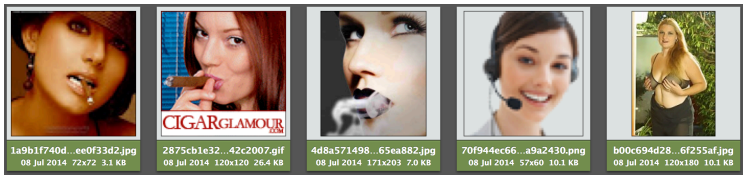


Fig. 9: Confusion matrices for websites and images for the F-Secure Corp. dataset with 2^{20} centroids. Numbers are percentages, averaged over 10 runs for validation data. Enhancing particular images' predictions into website classes enhances the accuracy.

(A)



(B)

Fig. 10: Some of the images from benign websites (A) and cigarette websites (B), predicted as "Adult".

5 Conclusion

The paper proposes a website classification methodology, which is based on image content only. Such method is independent of the language, benefits from having multiple images in a single website, and extends easily to new classes.

It starts from detecting local features in images and calculating their descriptors. Then a number of centroid descriptors are chosen randomly from images of each class. The k closest centroids are found for all descriptors of training and test images, and image representations are built. An OP-ELM image classifier is trained with representations of training images. Predictions of this classifier for images of each training website are used to get features for the next OP-ELM website classifier. Classes of images from a test website are predicted with the first classifier, then website features are build and used to obtain the website class predictions from the second classifier.

A major challenge is to deal with large data volumes, both in methodology and programming. It is solved successfully by the proposed modified version of k-NN classifier and a special image representation. This method is easily to parallelize and extend to new classes. The prediction accuracy comes from an OP-ELM image classifier, a fast and accurate method suitable for large datasets processing. It is non-parametric, very fast for prediction and fast for re-training if any classes should be added or removed from the system. The second stage OP-ELM predicts classes of websites as a whole, which gives a significantly higher prediction accuracy compared to images. Another benefit of using OP-ELM is the ease of incorporating additional input information like a number of images in a website. The proposed methodology is suitable for other Image Classification tasks and the major contribution of this paper comes from the cascade of mod-

els that is used. Using a cascade of models (modified k-NN classifier and two OP-ELM classifiers) allows optimizing the ratio Performance/(Computational Time). It is possible for the users to modify this ratio in order to give more importance to the Performance or the Computational Time. Both OP-ELM classifiers are very efficient, very fast and can be tuned easily. The modified k-NN classifier can be tuned in order to reduce drastically the computational time.

The website classification method will be deployed for predicting real incoming data, in collaboration with F-Secure Corp. The next development step is to minimize the amount of False Positive predictions with a trade-off between False Positives and coverage, which is important for practical purposes of using website predictions in a web content filtering software.

References

1. G. Amato and F. Falchi. kNN based image classification relying on local feature similarity. *Proceedings of the Third International Conference on SImilarity Search and APplications - SISAP '10*, page 101, 2010.
2. L. Ballan and M. Bertini. Combining generative and discriminative models for classifying social images from 101 object categories. *21st International Conference on Pattern Recognition (ICPR)*, (September):1–4, 2012.
3. H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *Proc. of the 9th European conference on Computer Vision*, pages 404–417, 2006.
4. R. Benenson and M. Mathias. Pedestrian detection at 100 frames per second. *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2903–2910, 2012.
5. I. Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115–47, Apr. 1987.
6. D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.
7. O. Boiman, E. Shechtman, and M. Irani. In defense of Nearest-Neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, number I, pages 1–8. IEEE, June 2008.
8. G. J. Burghouts and J.-M. Geusebroek. Performance evaluation of local colour invariants. *Computer Vision and Image Understanding*, 113(1):48–62, Jan. 2009.
9. X. Cai, B. Xiao, C. Wang, and R. Zhang. A local learning based Image-To-Class distance for image classification. In *2011 First Asian Conference on Pattern Recognition (ACPR)*, pages 667–671, 2011.
10. O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 10(5):1055–64, Jan. 1999.
11. Q. Chen, Z. Song, and Y. Hua. Hierarchical matching with side information for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3426–3433. IEEE, June 2012.
12. F. Crow. Summed-area tables for texture mapping. *ACM SIGGRAPH computer graphics*, 18(3):207–212, July 1984.
13. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 1:886–893, 2005.

14. T. Dean, M. Ruzon, and M. Segal. Fast, Accurate Detection of 100,000 Object Classes on a Single Machine. *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, 2013.
15. B. Divakaruni and J. Zhou. Image Categorization using Codebooks Built from Scored and Selected Local Features. *Proceedings of The 2011 International Conference on Image Processing, Computer Vision and Pattern Recognition (IPCV)*, 1:3–9, 2011.
16. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.
17. L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.
18. R. Fergus and L. Fei-Fei. Learning object categories from internet image searches. *Proc. of the IEEE*, 98(8):1453–1466, 2010.
19. G. Griffin and A. Perona. The Caltech 256. *Vasa*, 2012.
20. S. Haykin and N. Network. Neural Networks: A comprehensive foundation. *Neural Networks*, 2, 2004.
21. J. Hou, J. Kang, and N. Qi. On vocabulary size in bag-of-visual-words representation. *Advances in Multimedia Information Processing-PCM 2010*, pages 414–424, 2010.
22. G.-B. Huang, L. Chen, and C.-K. Siew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *Neural Networks, IEEE Transactions*, 17(4):879–892, 2006.
23. G.-B. Huang, Q.-Y. Zhu, K. Z. Mao, C.-K. Siew, P. Saratchandran, and N. Sundararajan. Can threshold networks be trained directly? *IEEE Transactions on Circuits and Systems II: Express Briefs*, 53(3):187–191, 2006.
24. G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1):489–501, Dec. 2006.
25. M. J. Huiskes and M. S. Lew. The MIR flickr retrieval evaluation. *Proceeding of the 1st ACM international conference on Multimedia information retrieval - MIR '08*, page 39, 2008.
26. Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *Pattern Analysis and Machine Intelligence*, 30(11):1877–1890, 2008.
27. D. Jurafsky and J. H. Martin. *Speech & language processing*. Pearson Education India, 2000.
28. P. Y. Lee, S. C. Hui, and A. C. M. Fong. Neural networks for web content filtering. *Intelligent Systems, IEEE*, 17(5):48–57, 2002.
29. T. Lindeberg. Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116, 1998.
30. D. Lowe. Object recognition from local scale-invariant features. In J. Tsotsos, editor, *Proc. of the Seventh IEEE International Conference on Computer Vision*, volume 2 of *ICCV '99*, pages 1150–1157. Kerkyra, Greece, IEEE, 1999.
31. D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.
32. M. Mathias, R. Timofte, R. Benenson, and L. Van Gool. Traffic sign recognitionHow far are we from the solution? In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.
33. Y. Miche, A. Sorjamaa, and P. Bas. OP-ELM: optimally pruned extreme learning machine. *Neural Networks, IEEE Transactions*, 21(1):158–162, Jan. 2010.

34. Y. Miche, M. van Heeswijk, P. Bas, O. Simula, and A. Lendasse. TROP-ELM: A double-regularized ELM using LARS and Tikhonov regularization. *Neurocomputing*, 74(16):2413–2421, Sept. 2011.
35. K. Mikolajczyk, B. Leibe, and B. Schiele. Local features for object class recognition. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 2:1792–1799, 2005.
36. K. Mikolajczyk and C. Schmid. *Comparison of affine-invariant local detectors and descriptors*. 2004.
37. K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, 2004.
38. K. Mikolajczyk and C. Schmid. Performance evaluation of local descriptors. *IEEE trans. on pattern analysis and machine intelligence*, 27(10):1615–1630, Oct. 2005.
39. M. Mofaddel and S. Sadek. Adult image content filtering: A statistical method based on Multi-Color Skin Modeling. In *2010 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 682–686, 2010.
40. R. Myers. *Classical and modern regression with applications*. Duxbury advanced series in statistics and decision sciences. Duxbury, Pacific Grove, CA, USA, Boston, 2nd edition, 1990.
41. M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. *CVPR*, 2014.
42. C. R. Rao and S. S. K. Mitra. Generalized inverse of a matrix and its applications. *Wiley Series in Probability and Mathematical Statistics (New York)*, 1971.
43. T. Similä and J. Tikka. Multiresponse sparse regression with application to multidimensional scaling. *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005*, 3697:97–102, 2005.
44. J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings. Ninth IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477. Ieee, 2003.
45. A. Torralba, R. Fergus, and W. T. Freeman. 80 Million Tiny Images: a Large Data Set for Nonparametric Object and Scene Recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970, Nov. 2008.
46. T. Tuytelaars and K. Mikolajczyk. Local Invariant Feature Detectors: A Survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.
47. K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE trans. on pattern analysis and machine intelligence*, 32(9):1582–1596, Sept. 2010.
48. M. van Heeswijk, Y. Miche, E. Oja, and A. Lendasse. GPU-accelerated and parallelized ELM ensembles for large-scale regression. *Neurocomputing*, 74(16):2430–2437, 2011.
49. A. Vedaldi and S. Soatto. Features for recognition: Viewpoint invariance for non-planar scenes. *ICCV 2005. Tenth IEEE International Conference on Computer Vision, 2005.*, 2:1474–1481, 2005.
50. V. Viitaniemi. *Visual category detection: an experimental perspective*. PhD thesis, 2012.
51. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 1:I—-511, 2001.
52. X. Wang, C. Hu, and S. Yao. An adult image recognizing algorithm based on naked body detection. *2009 ISECS International Colloquium on Computing, Communication, Control, and Management*, 4:197–200, Aug. 2009.

53. R. Weber, H. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the International Conference on Very Large Data Bases*, pages 194–205, 1998.
54. Q. Zheng, W. Zeng, W. Wang, and W. Gao. Shape-based adult image detection. *International Journal of Image and Graphics*, 6(1):115–124, 2006.