B. Mitschang et al. (Hrsg.): BTW 2017 – Workshopband,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 345

Fake war crime image detection by reverse image search

Alexander Askinadze[1]

Abstract: In the media, images of war crimes are often shared, which in reality come from other contexts or other war sites. In this paper an approach is proposed to detect duplicate or fake war crime images. For this, the bag of visual words model is used in conjunction with localized soft assignment coding and the k-nn classifier. For evaluation, a data set with 600 images of war crimes was crawled. Different distances and parameters were used for evaluation. Unmodified images can be recognized with this approach with 100% accuracy. Rotated and scaled images can also be detected with nearly 100% accuracy. Modifications like cropping or the combination of scaling and cropping ensure significantly smaller accuracy results. The run time was investigated and it was found that about 3000 images per second can be processed on an Intel Core i5 processor.

Keywords: Near-duplicate image detection, image recognition

1 Introduction

It has already been the case that images of war crimes were published and shared on mass media and social networks which originally came from a different context or were taken from previous war sites [Fu12][BB14]. This is often the case in the Near and Middle East, especially in connection with the conflicts in Iraq, Syria and Palestine. For example, an article by the BBC on the Gaza conflict in 2014 has shown that Palestinians in Gaza used fake pictures to share the suffering on social networks:

> Over the past week the hashtag #GazaUnderAttack has been used hundreds of thousands of times, often to distribute pictures claiming to show the effects the airstrikes.
>
> Some of the images are of the current situation in Gaza, but a #BBCtrending analysis has found that some date as far back as 2009 and others are from conflicts in Syria and Iraq. [BB14]

With more and more shared content in the media, it becomes increasingly important to find a way to verify the source of the image. This problem belongs to the field of duplicate image recognition. However, to the author's best knowledge, very few publications can be found in the literature that address the issue of fake war crime image detection. In order to solve this issue, this paper examines how the problem of detecting fake images of war crimes can be solved.

[1] Heinrich-Heine-Universität, Datenbanken und Informationssysteme, Universitätsstraße 1, 40225 Düsseldorf, askinadze@cs.uni-duesseldorf.de

The intended suggestion is a system where a user can query the image database with an image. This is called reverse image search. The search query can be made either as a file upload or through the URL of an image.

In Figure 1, such a scenario is presented where a user submits an image as a search input. The image is transformed into an internal representation and compared with similar representations present in the database. The best results are displayed to the user. If the user recognizes the image in one of these results, meta-information is displayed to the user. Meta-information can contain the first occurrence of this image and already existing URLs. Otherwise, the image is added to the database to participate in further search queries.

Such reverse image search systems are already available, e.g., at Google[2] or tineye[3]. These search engines already provide good search results, but soon fail with rotated images. It is possible that people who share fake images would also be willing to manipulate them so that such search engines do not recognize these images as duplicates. Therefore, a system is required that is invariant regarding rotation, scaling, brightness changes both of original images as well as of cropped images.

The paper is organized as follows. Section 2 gives a brief overview of necessary fundamentals of image processing, which are necessary to transform images into numerical vectors (image representations). Additionally, it presents how the numerical vectors can be compared with each other to recognize duplicates. The evaluation of the proposed approach on the crawled data set for various hyper-parameters is outlined in Section 4. The conclusions are drawn in the final section.

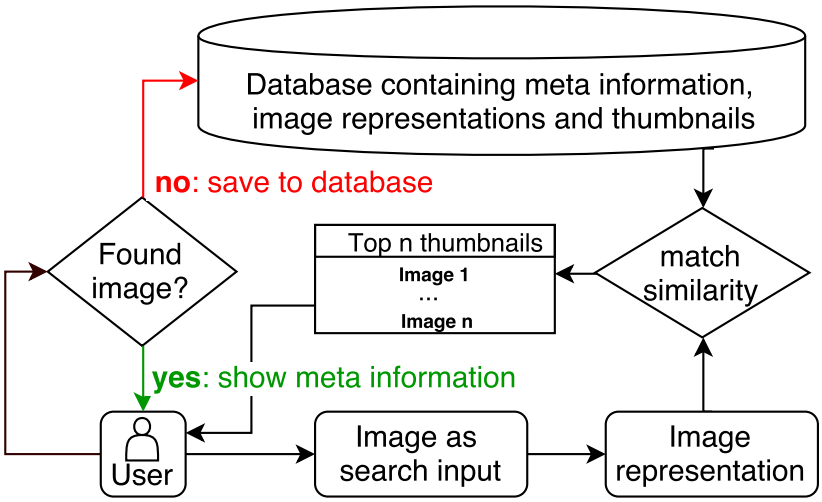


Fig. 1: Typical scenario of an reversed image search

[2] http://images.google.com/
[3] https://www.tineye.com/

2 Method

2.1 Image representation

An image is usually given as a pixel matrix with $n \times m$ pixels. Where n is the number of rows and m the number of columns of the matrix. Each image can have any size $n \times m$. Therefore, it is hard to compare two images in pixel representation. This requires an uniform image representation.

The Bag of Words model *(BoW)*, which is known from text processing, provides a way to represent images uniformly. Here, according to the words in a text, certain image features are selected as visual words. The visual words must be computed from a set of all image features. The set of visual words is called a visual codebook. The codebook creation process consists of the following three steps:

1. First of all, points are selected in an image. This can be done in a number of ways:
   (i) the points are distributed equally, (ii) only points are taken which are found by a point detector (e.g. an edge detector), or (iii) the points are selected randomly in the picture.
2. After the points have been selected, a descriptor describing the local properties of this point is computed for each point. Many methods of image processing use SIFT [Lo99] or SURF [Ba08] descriptors as image features. In [PPS13] the authors compared SIFT and SURF descriptors and came to the conclusion that SURF is faster and has a similar performance to SIFT. Therefore, SURF will be used as a descriptor.
3. Finally, the codebook is calculated from the set of all descriptors of all images. This large number of descriptors must be reduced to a small number of descriptors, which represent the entire quantity as well as possible. This is often implemented with the k-Means algorithm [Ll82]. The cluster centers are then chosen as visual words. The parameter k of the k-means algorithm thus determines the size of the codebook.

After the codebook has been created, the descriptors of each individual image are assigned to one of the visual words. A histogram is used to determine how often each visual word is taken as an assignment. This histogram thus also has the size K depending on the codebook size. This histogram is called *BovW* (bag of visual words) [Cs04]. Let h be the non-zero based BovW histogram, K the codebook size, D the set of all descriptors of an image, and $v_1, ..., v_K$ the visual words. For each descriptor $d \in D$, h is then updated as illustrated in algorithm 1.

Algorithm 1 Standard BovW Computation

for each d in D **do**
  $h[\arg\min_k ||d - v_k||] + +$
end for

Another strategy called *Soft assignment Coding* [Va10] extends this approach. Not only the one code word that is closest to the descriptor ensures the incrementation of the histogram

but all the code words. The weighting of the incrementation depends on the distance of the descriptor and the code words. Let h be the non-zero-based BovW histogram, K the codebook size, D the set of all descriptors of an image, and $v_1,...,v_K$ the visual words. For each descriptor $d \in D$, h is then updated as illustrated in algorithm 2. The parameter β indicates how soft the assignment is.

Algorithm 2 Soft assignment Coding

for each d in D **do**

for k:=1 to K **do**

$$h[k] \leftarrow h[k] + \frac{exp(-\beta||d - v_k||)}{\sum_{j=1}^{K} exp(-\beta||d - v_j||)} \tag{1}$$

end for
end for

In [LWL11] the authors have found that it is difficult to find a good choice for β that can handle both large and small distances well. Therefore, it is suggested that not all the visual words should be used, but only l neighbors of a descriptor. This strategy is called *localized soft-assignment coding*. Let h be the non-zero-based BovW histogram, K the codebook size, D the set of all descriptors of an image, and $v_1,...,v_K$ the visual words. For each descriptor $b \in D$ let $N_l(b)$ be the l nearest neighbors of b. The histogram h is then updated as illustrated in algorithm 3. The authors of [LWL11] say that the parameter β can be tuned

Algorithm 3 Localized soft assignment Coding

for each b in D **do**

for k:=1 to K **do**

$$\hat{d}(b, v_k) \leftarrow \begin{cases} ||b - v_k||, & \text{if } v_k \in N_l(b) \\ \infty, & \text{otherwise} \end{cases}$$
$$h[k] \leftarrow h[k] + \frac{exp(-\beta\ \hat{d}(b, v_k))}{\sum_{j=1}^{K} exp(-\beta\ \hat{d}(b, v_j))} \tag{2}$$

end for
end for

by cross-validation. To create the BovW histograms, *localized soft assignment coding* and the parameters $\beta = 10$ and $l = 5$ given in [LWL11] will be used.

2.2 Similarity Search

For the classification of images, many different classifiers such as *k-NN*, *SVM*, *decision trees* or *neural networks* can be used [CHV99] [BZM07] [KSH12]. In the near-duplicate detection of images, there are just as many classes as images. With the BovW histograms a

uniform image representation has been found. Now a similarity search is enough. Therefore, the k-nearest neighbors (*k-nn*) algorithm will be used.

If the image being searched for is present in the database, the distance of the image representations would be 0 and the 1-nn classifier would recognize the correct image. If the image being searched for is a modified version of an image present in the database, the distance of the BovW histograms should be greater than 0. In this case, it would be appropriate to show the user the k nearest neighbors images whose distances are smallest from the input image.

In order to perform a similarity search, a distance must be calculated for all existing image representations. For n images in the database, a linear run time $\mathcal{O}(n)$ is obtained. Since most distance functions compare the corresponding dimension of two vectors, the run time also depends on the length of the BovW histograms. For a length m of BovW histograms and n images in the database, a $\mathcal{O}(nm)$ run time can be expected.

Because the result can depend on the selected distances, different distances will be investigated. This includes the following distances:

- Manhattan distance. For two n-dimensional vectors x, y the distance is defined by:

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i| \tag{3}$$

- Euclidean distance. For two n-dimensional vectors x, y the distance is defined by:

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4}$$

- The cosine distance can be derived from the cosine similarity. For two n-dimensional vectors x, y the distance is defined by:

$$d(x,y) = 1 - \frac{\sum_i^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2}\sqrt{\sum_{i=1}^n y_i^2}} \tag{5}$$

3 Evaluation

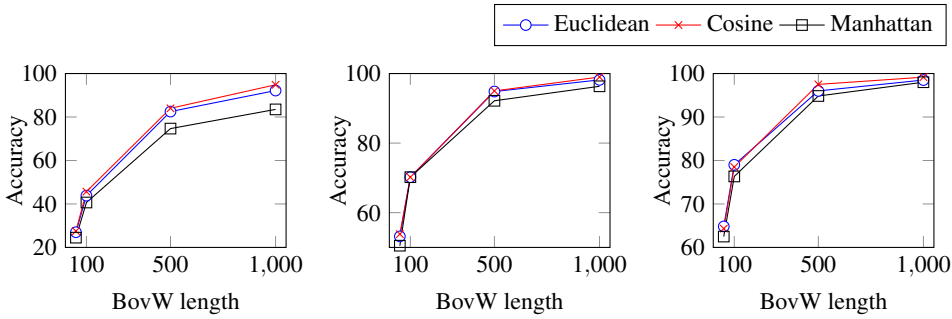
To perform an evaluation, an image data set was created by crawling Google image results for the search terms "Syria war crimes", "Iraq war crimes", and "Gaza war crimes". The search results had to be cleansed initially because they often contained images of politicians or caricatures. Of each of the search requests, 200 images were taken so that the data set consists of a total of 600 images.

It is the Intention to examine how well the war crimes can be detected in different manipulation stages. The investigated manipulation stages include: original images, scaled images, rotated images, cropped images as well as cropped and simultaneously scaled images.

To get the results for different parameters, a grid search will be performed. The examined parameters include: distances (Euclidean, Manhattan, Cosine), length of BovW histograms (50,100,500,1000) and number of search results (1,5,10) for the user in which the image could be recognized. So in total, $3 \times 4 \times 3 = 36$ different combinations. In addition, the training time for different lengths of the BovW histograms and the search run time for the 36 combinations was measured. The overall measurements results of all runs are summarized in Table 1. A detailed visualization of the results, which will be discussed in the following, can be found in Figs. 2, 3, and 4. The visualization of the training and search time can be found in Figs. 5 and 6.

As shown in Table 1, an unchanged image is recognized in almost every run with a precision of 100%. The results for rotated images (rotation around 90 degrees) are slightly less accurate but also around the 100%.

For the evaluation of scaled images, the originals were scaled to 50% of the original size. The results for different number of search results are shown in Figs. 2a, 2b, and 2c. If only a single search result is displayed to the user, the accuracy at large lengths of the BovW histograms is at both the Euclidean and Cosine distances close to 95%. If 10 possible image match candidates are displayed to the user, the accuracy increases to 100%

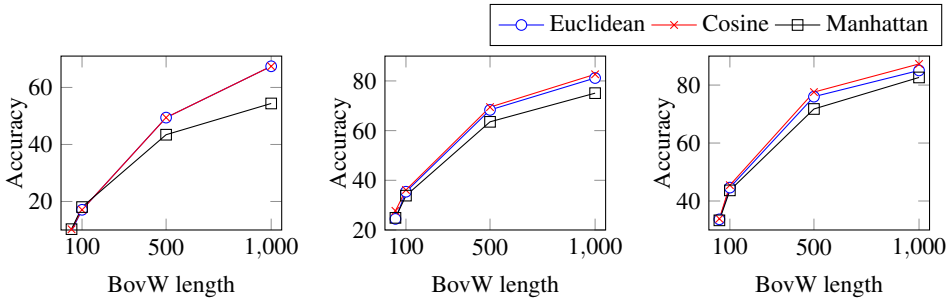


(a) 1 search result (b) 5 search results (c) 10 search results

Fig. 2: Accuracy values for **scaled** image queries for different count of search results

The results for cropped images for a different number of search results are shown in Figs. 3a, 3b, and 3c. To evaluate cropped images, the original image were divided into four equal parts. Each part was used as a search query, and so there were a total of 2400 search requests. With increasing length of the BovW vectors, the accuracy increases significantly in all three cases. While the accuracy results of all runs with Euclidean and Cosine distance are approximately equal, the results of the runs with the Manhattan distance are smaller. As shown in Figure 3a, the highest achieved accuracy value for a single search result is 67%. The tendency increases, which motivates the use of a larger length of BovW histograms. If ten search results are displayed to the user, the accuracy of the image being found is 87.29%, as shown in Figure 3c. Here too, a larger length of BovW histograms is expected to increase the accuracy.

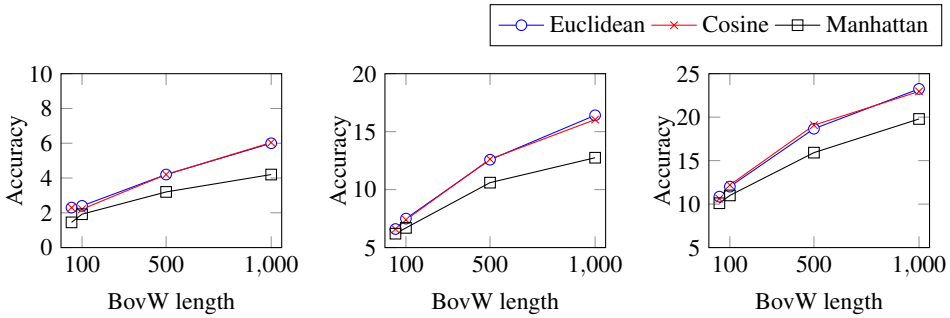
In order to test the limits of this approach, quarters were cropped from the images and then scaled to half of the original size. The accuracy results for different numbers of search



(a) 1 search result (b) 5 search results (c) 10 search results

Fig. 3: Accuracy values for **cropped** image queries for different count of search results

results are shown in Figs. 4a, 4b, and 4c. Compared to the results of scaled and cropped images, the results for the combination of the two manipulations are clearly poorer. With a single search result, only 6% accuracy can be achieved. With 10 search results, the accuracy increases to about 25%, but remains comparatively small. Here too, the two distances, Euclidean and Cosine, show significantly better values than the Manhattan distance.

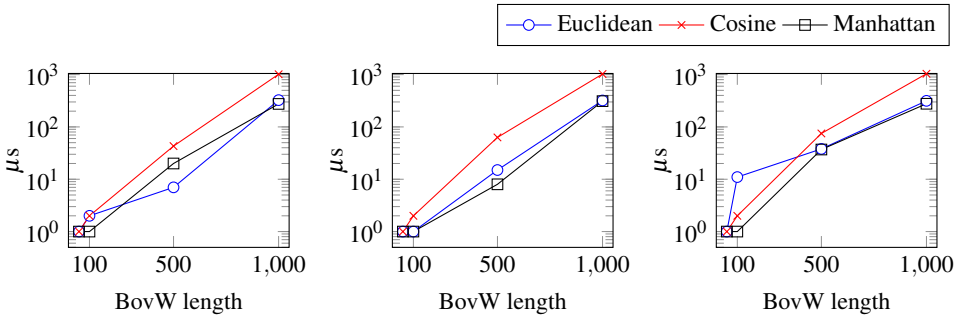


(a) 1 search result (b) 5 search results (c) 10 search results

Fig. 4: Accuracy values for **cropped and scaled** image queries for different count of search results

Search times per image for different numbers of search results are shown in Figs. 5a, 5b, and 5c. It is clear that the Cosine distance caused a significantly longer duration compared to the Euclidean and the Manhattan distance. The Manhattan and the Euclidean distance have similar values. From the fact that the Euclidean distance gives better results than the Manhattan distance, the use of the Euclidean distance is recommended. For BovW lengths of 1000 and the Euclidean distance, the search for an image takes about $\frac{1}{3}$ ms. In a database with n images, a search time of approximately $\frac{n}{3}$ ms can thus be expected on a computer with a Intel Core i5-3740 processor meaning that about 3000 images per second can be processed for example. The values were calculated during processing on a single core. Since this task is highly parallelizable, significantly faster running times can be implemented.

The training time depends on the running time of the k-means algorithm. This depends in turn on the number of data points and the number k of the clusters. In Figure 6, the training



(a) 1 search result (b) 5 search results (c) 10 search results

Fig. 5: Search time per image for different count of search results

time for different number of clusters (or the resulting BovW histograms) is illustrated. The running time increases significantly with the number of clusters. For 1000 clusters the running time is approximately 1 hour. In our investigation, all surf descriptors were used for clustering. The run time could be significantly reduced if only one random subset of descriptors is used by each image.

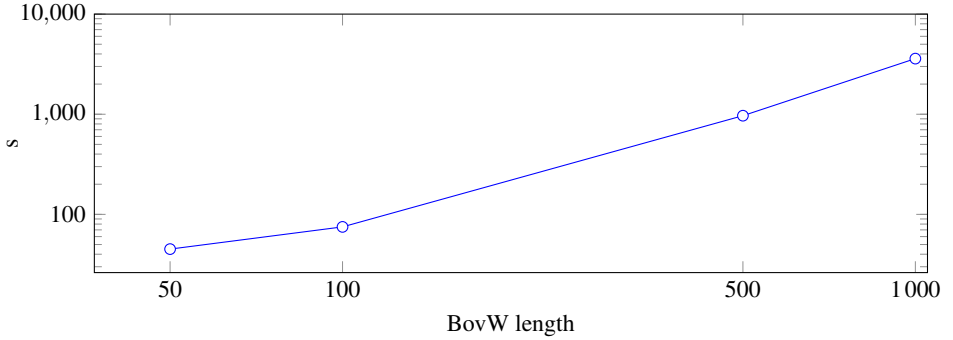


Fig. 6: Training time (k-Means Clustering) for the whole data set

4 Conclusion

In this paper, an approach to detect duplicate or fake war crime images is proposed. There is a considerable amount of literature on duplicate image detection, but no studies have been published on this special task. This approach consists of using the bag of visual words model in conjunction with localized soft assignment coding and the k-nn classifier. With this approach, unmodified images can be recognized with an accuracy of 100%. Rotated and scaled images can also be detected with an accuracy of nearly 100%. Modifications like cropping or the combination of scaling and cropping ensure significantly smaller accuracy results.

Future work will focus on improving the accuracy of strongly modified images. It also needs to be examined how new images can be used to improve the model without causing a completely new training. In a second, about 3000 images can be compared with respect to similarity. This can already be used in a web service. However, the run-time should be optimized for use with larger databases.

Tab. 1: Evaluation results (ordered by results of scaled and cropped image queries)

Run	Settings			Accuracy in %					run time	
	Distance: **E**uclidean, **M**anhattan, **C**osine	BovW length	Count search results	not scaled images			scaled images		train time in s, search time in μs per image	
				original	rotated	cropped	original	cropped	train	search
1	*E*	1000	10	100	99,5	85,04	98,5	23,25	3593	315
2	*C*	1000	10	100	100	87,29	99,16	22,96	3593	1037
3	*M*	1000	10	100	100	82,63	98	19,79	3593	277
4	*C*	500	10	100	100	77,58	97,5	19,08	967	75
5	*E*	500	10	100	99,67	76	96	18,67	967	38
6	*E*	1000	5	100	99,5	81,17	98,17	16,41	3593	320
7	*C*	1000	5	100	100	82,67	99	16,04	3593	1025
8	*M*	500	10	100	100	71,75	94,83	15,92	967	37
9	*M*	1000	5	100	100	75,08	96,33	12,75	3593	311
10	*C*	500	5	100	100	69,46	95	12,625	967	63
11	*E*	500	5	100	99,5	68,33	94,83	12,58	967	15
12	*C*	100	10	100	100	45,46	78,5	12,2	76	2
13	*E*	100	10	100	99,67	44,625	79	12,01	76	≤ 1
14	*M*	100	10	100	100	43,67	76,33	11	76	≤ 1
15	*E*	50	10	100	99,33	33,67	64,83	10,86	45	≤ 1
16	*C*	50	10	100	99,33	33,88	64,33	10,63	45	≤ 1
17	*M*	500	5	100	100	63,54	92,17	10,6	967	8
18	*M*	50	10	100	99,67	33,33	62,5	10,1	45	≤ 1
19	*E*	100	5	100	99,5	35,4	70,2	7,5	76	≤ 1
20	*C*	100	5	100	99,67	36,1	70,2	7,4	76	≤ 1
21	*M*	100	5	100	99,83	33,83	66,5	6,7	76	≤ 1
22	*E*	50	5	100	99,17	24,46	53,17	6,6	45	≤ 1
23	*C*	50	5	100	99	24,7	53,83	6,6	45	≤ 1
24	*M*	50	5	100	99,17	24,88	50,5	6,2	45	≤ 1
25	*C*	1000	1	99,5	99,17	67	94,83	6,04	3593	1015
26	*E*	1000	1	99,5	98,5	67,42	92,17	6	3593	325
27	*E*	500	1	99,33	98,5	49,42	82,5	4,2	967	7
28	*C*	500	1	99,5	99	49,2	84	4,2	967	43
29	*M*	1000	1	99,67	99,33	54,38	83,5	4,2	3593	275
30	*M*	500	1	99,33	99,17	43,42	74,67	3,2	967	20
31	*E*	100	1	99,5	97,67	17,08	43,83	2,4	75	2
32	*E*	50	1	99,67	94	9,6	27	2,3	45	≤ 1
33	*C*	50	1	99,33	94,33	10,25	27,5	2,3	45	≤ 1
34	*C*	100	1	99,67	97,5	17,63	45,66	2,21	75	2
35	*M*	100	1	99,5	98,5	18,04	40,67	1,92	75	≤ 1
36	*M*	50	1	99,83	94,17	10,29	24,5	1,45	45	≤ 1

References

[Ba08] Bay, Herbert; Ess, Andreas; Tuytelaars, Tinne; Van Gool, Luc: Speeded-up robust features (SURF). Computer vision and image understanding, 110(3):346–359, 2008.

[BB14] BBC Trending: , #BBCtrending: Are #GazaUnderAttack images accurate? http://www.bbc.com/news/blogs-trending-28198622, 2014. [Online; accessed 21-October-2016].

[BZM07] Bosch, Anna; Zisserman, Andrew; Munoz, Xavier: Image classification using random forests and ferns. In: 2007 IEEE 11th International Conference on Computer Vision. IEEE, pp. 1–8, 2007.

[CHV99] Chapelle, Olivier; Haffner, Patrick; Vapnik, Vladimir N: Support vector machines for histogram-based image classification. IEEE transactions on Neural Networks, 10(5):1055–1064, 1999.

[Cs04] Csurka, Gabriella; Dance, Christopher; Fan, Lixin; Willamowski, Jutta; Bray, Cédric: Visual categorization with bags of keypoints. In: Workshop on statistical learning in computer vision, ECCV. volume 1. Prague, pp. 1–2, 2004.

[Fu12] Furness, Hannah: , BBC News uses 'Iraq photo to illustrate Syrian massacre'. http://www.telegraph.co.uk/culture/tvandradio/bbc/9293620/BBC-News-uses-Iraq-photo-to-illustrate-Syrian-massacre.html, 2012. [Online; accessed 21-October-2016].

[KSH12] Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105, 2012.

[Ll82] Lloyd, Stuart: Least squares quantization in PCM. IEEE transactions on information theory, 28(2):129–137, 1982.

[Lo99] Lowe, David G: Object recognition from local scale-invariant features. In: Computer vision, 1999. The proceedings of the seventh IEEE international conference on. volume 2. Ieee, pp. 1150–1157, 1999.

[LWL11] Liu, Lingqiao; Wang, Lei; Liu, Xinwang: In defense of soft-assignment coding. In: 2011 International Conference on Computer Vision. IEEE, pp. 2486–2493, 2011.

[PPS13] Panchal, PM; Panchal, SR; Shah, SK: A comparison of SIFT and SURF. International Journal of Innovative Research in Computer and Communication Engineering, 1(2):323–327, 2013.

[Va10] Van Gemert, Jan C; Veenman, Cor J; Smeulders, Arnold WM; Geusebroek, Jan-Mark: Visual word ambiguity. IEEE transactions on pattern analysis and machine intelligence, 32(7):1271–1283, 2010.