# The Dual Codebook: Combining Bags of Visual Words in Image Classification

(Bachelor Project)

J.L.Maas, s2363143, J.L.Maas@student.rug.nl,
M.A.Wiering*, E.Okafor*

August 23, 2016

## Abstract

In this thesis, we evaluate the performance of two conventional bag of words approaches, using two basic local feature descriptors, to perform image classification. These approaches are compared to a novel design which combines two bags of visual words, using two different feature descriptors. The system extends earlier work wherein a bag of visual words approach with an L2 support vector machine classifier outperforms several alternatives. The descriptors we test are raw pixel intensities, and the Histogram of Oriented Gradients. Using a novel Primal Support Vector Machine as a classifier, we perform image classification on the CIFAR-10 and MNIST datasets. Results show that the dual codebook implementation successfully utilizes the potential contributive information encapsulated by an alternative feature descriptor, and increases performance, improving classification by 5-18% on CIFAR-10, and 0.22-1.03% for MNIST compared to the simple bag of words approaches.

**Keywords:** Histogram of Oriented Gradients, Bag of Visual Words, Dual Codebook, Machine Learning, Image Classification

## 1 Introduction

In this thesis, we propose the use of a Dual Bag of visual Words model (Dual-BOW) in a relatively conventional framework to perform image classification. Within computer vision, there are many approaches that have been used to evaluate classification performance [1]. The challenge which renders many conventional machine learning techniques unfeasible includes how to correctly recognize an object from an image, which may be rotated, scaled, illuminated, or oriented differently.

A popular approach utilizes what is known as the bag of visual words (BOW) [2], which has been shown to reach good performances on multiple tasks [3] [4], and is also simple in design.

Our goal of this study is to research the additional effect of combining two bags of words, using different local feature descriptors (LFD), to assess the performance increase (if any) of combining essential information encapsulated by using different local feature descriptors.

Two popular, and diverse, benchmarks datasets often used in this field are the MNIST and CIFAR-10 datasets. MNIST [5] consists of 70,000 (60,000 training, 10,000 testing) 28 x 28 pixel images of 10 classes of digits. Though often considered a simplistic dataset, it remains a popular benchmark, and provides plenty research to compare with. CIFAR10 [6] consists of 60,000 (50,000 training, 10,000 testing) 32 x 32 colour images, constructed from 10, more diverse classes (ranging from animals to vehicles).

**Outline** This thesis is organized as follows: Section 2 describes the system design, and covers the implementations of the LFDs, the classifier, dual codebook, and the experiment we performed. Section 3 describes the results from the experiments, and is followed by a thorough discussion in Section 4, and a conclusion in Section 5.

*University of Groningen, Department of Artificial Intelligence

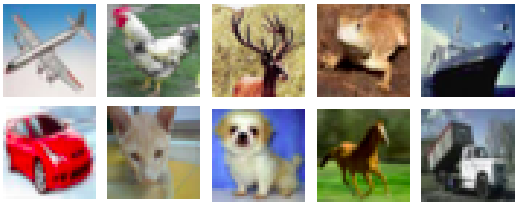

**Figure 1: Samples from the CIFAR-10 dataset.**

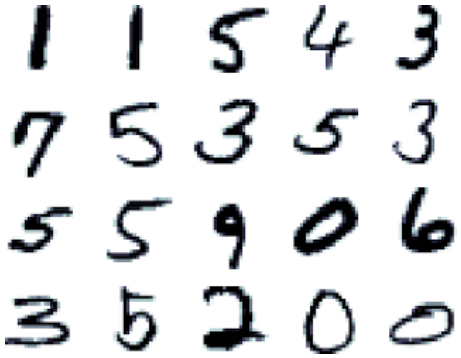

**Figure 2: Samples from the MNIST dataset.**

# 2 System Design

The system design builds upon the framework used in [7], wherein a bag of visual words is used, and the performance of several different local feature descriptors was evaluated. Herein, they also compare the performance of several types of support vector machines.

## 2.1 Datasets

As previously mentioned, the datasets used are CIFAR-10 [6] (see Figure 1) and MNIST [5] (see Figure 2). The methodology used relies on extraction of so called patches, sub-parts of the image, that can be extracted using a sliding window of a fixed size.

For MNIST, the images were rescaled (using cubic interpolation) to an image resolution of 48 x 48 pixels, after which patches of 14 x 14 pixels were extracted.

For CIFAR-10, smaller patchsizes of 8 x 8 were more appropriate, as patch size appeared to have a large impact depending on the dataset used. The image size remained unchanged at 32 x 32 pixels.

For each dataset, we tested the classification performance of 10,000 images, and the classifier used (explained in Section 2.5) trained on 50,000 and 60,000 images for CIFAR-10 and MNIST respectively.

## 2.2 Local Feature Descriptors

We designed our system with flexibility in mind, as such that it enables swapping different local feature descriptors*, allowing different patch sizes, and implementation methodologies.

### 2.2.1 Raw Pixel Intensities

The raw pixel intensities method directly uses the RGB intensities of the pixels within a patch, and is the default feature descriptor used for a conventional visual bag of words approach. Simple as it may be, its successes in several tasks have shown its potential [8], and show that raw pixel intensities within patches can be used to represent interesting features. Nevertheless, the feature vector length can grow very large when larger patches are used, especially in colour images (which is the case for the 3-channel CIFAR-10 dataset, as opposed to the single-channel MNIST dataset).

In our experiments, for MNIST, the patch size of 14 x 14 pixels results in a patch-feature length of 196 elements. For CIFAR-10, however, we need to track three colour channels of a 8 x 8 pixel patch, which results in a patch-feature length of 192.

After computing the patch-feature vector, it is standardised. Though we included modules for performing different levels of pre- and postprocessing, we settled on using only standardisation where appropriate.

Standardisation of a vector is performed by computing the mean:

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

*For the experiment, however, only two local feature descriptors were used. We also intended to include a local binary patterns feature descriptor, but at the time did not possess the computational resources to include it in our research.

of its elements. Then, the deviation is computed by:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(\overline{x} - x_i)^2}{n} + e}$$

After which the standardised vector is obtained by updating the vector values:

$$x_i' = \frac{x_i - \overline{x}}{\sigma}$$

We used this standardisation scheme on several occasions within the design.

#### 2.2.2 Histogram of Oriented Gradients

The Histogram of Oriented Gradients [9] (known as HOG) has been a popular feature descriptor for a long while, and knows several different uses [10] [7]. To compute the descriptor, gradient components are computed for the horizontal and vertical gradient ( $G_x$ and $G_y$ respectively ) for every pixel in the patch. Though multiple masks can be used, the simple kernel $[-1, 0, +1]$ bears preference [11]. The gradients are computed as:

$$G_x = f(x+1, y) - f(x-1, y)$$

$$G_y = f(x, y+1) - f(x, y-1)$$

where $f(x, y)$ is the pixel intensity at coordinate x,y. The final Magnitude $M(x, y)$ (intensity of change) and orientation $\theta(x, y)$ (direction of change) are computed as:

$$M(x, y) = \sqrt{G_x^2 + G_y^2}$$

$$\theta(x, y) = \tan^{-1}\frac{G_y}{G_x}$$

After computing the magnitudes and orientations for every pixel, the patch is segmented into four quadrants[†]. Within each quadrant, the magnitudes of all pixels are binned using linear interpolation (thus the binned magnitude is distributed over the neighbouring bins) into a histogram by the corresponding orientations, which produces the Histogram of Oriented Gradients. After computing the histograms of all four quadrants, these are concatenated to produce the feature vector representing the patch.

For our experiment, we used 9 bins to represent orientations in a range of $0 - 180^o$ (thus a bin width of 20 degrees). Since the patch sizes do not determine the HOG's feature vector size, the feature vector length for MNIST is 36. For the tri-colour channel CIFAR-10, it is 108.

For MNIST, a patch size of 14 x 14 pixels is reduced to 12 x 12 to cope with padding, after which HOG is computed for four 6 x 6 pixel cells. For CIFAR-10, a patch size of 8 x 8 pixels is reduced to 6 x 6 for the same reason, and the HOG is computed for four 3 x 3 pixel cells.

As with the raw pixel intensities local feature descriptor, the HOG feature vector is also standardised.

### 2.3 Kinds of Codebooks

In this section, we will explain the 2 types of codebooks we used in our project.

#### 2.3.1 Classic Codebook

The Bag of Visual Words has been a popular tool in computer vision and classification [2], wherein an image can be represented by regarding the patches that it is composed of. Using this methodology, one can create a bag of words by applying an unsupervised algorithm (such as K-means clustering [12]), on a random collection of patches, extracted from images from the training set.

The resulting centroids are intended to represent generalized patches, or visual words, and as a whole act as a dictionary (which we refer to as a codebook within the context of this paper), representing which visual elements are acknowledged to exist and occur in the data [7].

Once the codebook is constructed, it can be used to represent a new image. This is done by partitioning a given image $N$ into $S$ (non-overlapping) segments, of equal size. Within every segment, $n$ patches are extracted using a sliding window of a custom size and shift. The derived set of patches are then described by feature vectors using the appropriate local feature descriptor.

Hereafter, the activations are computed in the following fashion. For every patch-feature $p_i \in$

[†]It should be noted that this window can also be regarded as simply the patch itself if its configured window size equals the patch size, or allow overlap if desired. However, for our research, we limited to only full patch sizes, and different levels of segmentations.

$\mathbb{R}^n$ from the collection of patches within a segment, distances are computed to each word $w_j \in \mathbb{R}^n$ from a codebook $C^l = \{w_1, w_2 ... w_K\}$ (where $l \in \{IMG, HOG, DUAL\}$ denotes the appropriate feature descriptor), using a distance function $d(p_i, w_j)$.

In our experiment, we used the Euclidean distance as distance function:

$$d(p_i, w_j) = \sqrt{\sum_{x=1}^{n} \left(p_i^x - w_j^x\right)^2}$$

to represent the distance from a patch $p$ from an image, to centroid $w$ from the codebook, over all elements of its feature vector length.

Computing the distance to all words allows us to compute the mean distance of patch $p_i$ to all words:

$$\bar{d}(p_i, w) = \frac{\sum_{j=1}^{K} d(p_i, w_j)}{K}$$

Hereafter, we can compute the new activations according to the Soft-Assignment function[4], by updating the activation vector $a_j \in \mathbb{R}^K$, which denotes the activations of the codebook centroids, with respect to the patches within the segment. For every patch $p_i \in \mathbb{R}^n$, the activation value $(a_j)$ of word $w_j$ is updated by:

$$a_j = \begin{cases} a_j, & \text{if } \epsilon \leq 0 \\ a_j + \epsilon, & \text{if } \epsilon > 0 \end{cases}$$

Where $\epsilon = \bar{d}(p_i, w) - d(p_i, w_j)$ (and corresponds to a similarity measure between a patch and a word).

Repeating this procedure for every patch within segment $s \in \mathbb{R}^S$ gradually generates its activations vector:

$$A_s(K) = \{a_1, a_2, ... a_K\}$$

To create the final feature vector, $x_N^l$, representing a given image $N$, using codebook $l$ (and its corresponding local feature descriptor), the activations of all $S$ segments of the image are concatenated:

$$x_N^l(s) = \{A_1; A_2; ... A_S\}$$

and standardised once.

The resulting final feature vector can be used as training and testing data for any classifier of choice.

Obviously, computational complexity in this approach grows with feature descriptor size, and the number of centroids used. The dimensionality of the final feature vector of the image, corresponds to $S * K$, where $S$ corresponds to the number of segments the image is partitioned in, and $K$ the number of centroids in the codebook used. The codebooks were generated using 200,000 patches randomly extracted from the dataset used.

We created the codebooks using conventional K-means clustering (Lloyd's Algorithm), with 150 iterations. For both raw pixel intensities (IMG / BoW) and the histogram of oriented gradients descriptor (HOG-BOW), we performed runs using 400, and 800 centroids, wherein images are partitioned into 9 segments (3 x 3), thus resulting in feature dimensionalities of 3,600 and 7,200 for 400 and 800 centroids respectively.

### 2.3.2 Dual Codebook

We propose the combination of both the raw pixel intensities and HOG features to develop a dual codebook. This enigma of combining features within the scope of the visual bag of words approach knows little prior research [13]. In essence, the dual codebook is the combination of two codebooks, which may have been generated either using the same local feature descriptor (possibly under a different configuration), or an entirely different one. The configuration of the second codebook is not bound by those used in the first, and thus may also operate with a different number of centroids.

In this fashion, given two codebooks $C^{IMG}$ and $C^{HOG}$ (generated using raw pixel intensities, and the histogram of oriented gradients respectively), an image $N$ is represented by computing the activations, $x_N^l$, for both codebooks towards this image. The activation vectors obtained, $x_N^{IMG}$ and $x_N^{HOG}$ are then concatenated:

$$x_N^{DUAL} = \left\{x_N^{IMG}; x_N^{HOG}\right\}$$

to create the final feature vector of the image under the dual codebook approach.

This approach effectively allows combination of two different local feature descriptors, which can aid classification accuracy by inclusion of potentially essential information which may be encapsulated by the one, but not the other feature descriptor.

In our experiment, the dual codebook was evaluated under the same configurations as its singular alternatives, and combines two codebooks of 400 centroids each. This configuration therefore results in a final feature vector with a dimensionality of 7,200. Based on the dual codebook used in this section, the new bag of visual word formed can be referred to as Dual-BOW.

### 2.4 Classifier

For classification, we designed an L2 'primal' support vector machine (one for each class) as described in [14], using a revised objective function:

$$\min_{\omega,b} L = \|\omega\|^2 + C \cdot \sum_N \xi_N^2$$

and output function:

$$g(x_N) = \omega \cdot x_N + b$$

where $x_N = x_N^l$ denotes the centroid activations from the bag of words, using descriptor $l$ , and the error is represented as:

$$\xi_N = \max(0, 1 - y_N \cdot g(x_N))$$

$y_N \in \langle -1, 1 \rangle$ represents whether the target label of example $x_N$ belongs to the class which this SVM represents. Training is done in iterations, and all training data are presented in each iteration. For every iteration, if the output label doesn't correspond to the class ($y_N \cdot g(x_N) < 1$), then the weights are adjusted using the formula:

$$\Delta w_j = -\lambda \cdot (\frac{w_j}{C} - (y_N - g(x_N)) \cdot x_N^j)$$

Where $\lambda$ denotes the learning rate. At the end of every iteration, the bias $b$ is updated to represent the mean error $y_N - g(x_N)$ of all examples where $y_N \cdot g(x_N) < 1$.

We used the L2 primal Support Vector Machine [14], with a learning rate $\lambda$ of 0.0000001, and performed 2000 training iterations before classifying. The initial weight values are 0.000002, and C is set to 2048.

### 2.5 Experiment

In total, for both MNIST and CIFAR-10, we designed 5 experiment configurations. For the single bag of word approaches, and both feature descriptors, we performed runs with codebooks of 400 and 800 centroids, whereas the dual codebook implementation was run with two codebooks of 400 centroids each. We performed 10-Monte Carlo cross validation runs for every of the 5 configurations (BoW-400, BoW-800, HOG-BoW-400, HOG-BoW-800, DUAL-2x400). The results are described in the next section.

## 3 Results

In this Section, we will present the results for both MNIST and CIFAR-10, based on the 10-Monte Carlo cross validation runs, as can be seen in the Table below.

| **Methods** | **MNIST** | | **CIFAR-10** | |
|---|---|---|---|---|
| | *Mean* | SD | *Mean* | SD |
| BoW-400 | 1.85 | 0.14 | 47.59 | 0.42 |
| BoW-800 | 1.71 | 0.10 | 47.96 | 9.00 |
| HOG-BoW-400 | 1.22 | 0.12 | 41.28 | 0.61 |
| HOG-BoW-800 | 1.05 | 0.13 | 54.98 | 12.64 |
| Dual-BoW-2x400 | 0.83 | 0.09 | 36.20 | 2.60 |

**Table 1: Classification Error (in %) on test-sets of MNIST and CIFAR-10, 10-fold Monte Carlo Cross Validations.**

### 3.1 Evaluation of the CIFAR-10 Dataset

The results of classification on the CIFAR-10 dataset are visualized in Figure 3 (see below). As shown in Table 1, the dual codebook reaches commendable classification performance. Though not stellar nor exceeding present state-of-the-art performance [15], the results still reflect the added value of the dual codebook, resulting in a significant performance increase compared to all single codebook variants.

Student's T-tests shows the dual codebook performs better than the Histogram of Oriented Gradients with 400 centroids ($t = 6.01$, $p < 0.05$), outperforms the 800-centroid variant ($t = 4.60$, $p < 0.05$), and surpasses both 400 and 800-centroid raw pixel intensities (conventional BoW) implementations ($t = 13.26$, $p < 0.05$ and $t = 3.97$, $p < 0.05$, respectively).

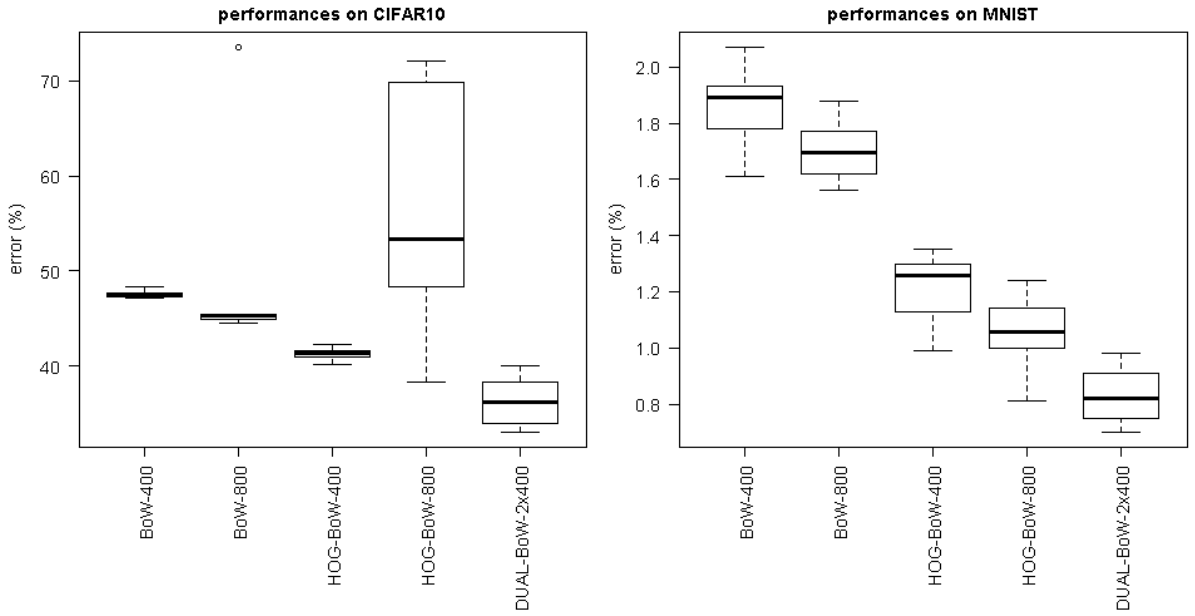


**Figure 3: Error rates for CIFAR-10(left) and MNIST(right)**

Therefore, on CIFAR-10, the Dual-BOW approach, which employs the dual codebook, appears superior to both BOW and HOG-BOW that uses only a single codebook, because it obtains the lowest error rate.

### 3.2 Evaluation of the MNIST Dataset

Though performance improvements may not be as pronounced as those in CIFAR-10[‡] the dual codebook again significantly outperforms all single codebook configurations (see Figure 3 and table 1).

Student's T-test indicate significant improvements over HOG-BoW-400 and HOG-BoW-800 ($t = 8.26$, $p < 0.05$ and $t = 4.50$, $p < 0.05$ respectively). With regard to raw pixel intensities, the Dual-BOW approach significantly outperforms both the BoW-400 ($t = 19.01$, $p < 0.05$) and BoW-800 ($t = 19.97$, $p < 0.05$) implementations.

Thus, the results on MNIST reflects those of CIFAR-10, showing that the Dual-BOW again outperforms conventional BOW approaches utilizing only single codebooks.

## 4 Discussion

In this thesis, we have demonstrated the dual codebook's superiority over comparable single codebook approaches, showing a consistent performance improvement over two substantially different datasets. This implies the capability of successfully combining the essential information encapsulated by different local feature descriptors, improving classification performance.

Though both the datasets and approach used may be considered simplistic to current standards, it does not appear that the dual codebook approach would perform worse with alternative datasets, than single codebook alternatives would [§].

[‡]Even simple KNN-approaches have been known to reach 95% accuracy on MNIST, though anything above 99% can be regarded as decent.

[§]That is, where the dual codebook incorporates the local feature descriptor used for the single codebook alternative. Obviously, the role of a good classifier cannot be neglected in assessing performance.

# 5 Conclusion

Though performance on either dataset is not present state-of-the-art, it should be kept in mind that many of the data-preprocessing enhancements and excessive parameter tuning conventionally performed for these datasets were not applied, as we intended to study the exclusive benefit of the dual codebook approach, with regard to conventional bag of words approaches that utilize only a single codebook. Therefore, these results say little about the limits of the dual codebook approach, which was used in a quite simple configuration in this experiment. Under slightly more computationally demanding configurations of the primal SVM, performance for CIFAR10 for the dual codebook reached scores up to 73.18%, and for MNIST up to 99.3%. However, these results were discarded under the need to perform cross validations with limited computational resources, and time constraints.

With regard to future research, there are many possibilities. We intend to expand the design to an N-codebooks implementation, which will be able to combine N bags of words in order to investigate to what extent this advantage remains.

Additionally, it might be worth investigating the potential value of combining codebooks of the same feature descriptor, but under different configurations (for example, Histogram of Oriented Gradients with a different segmentation grid, or different bin distributions). Other grounds for further research could focus on the necessary sizes of the codebooks in regard to feature vector dimensionality, as it would be ideal if one were able to improve performance by incorporating a mere 100-centroid small extra codebook, which might be based on a local feature descriptor with a computational complexity or intensity too high to consider for larger codebooks.

Alternative considerations might be to use a deep codebook [16] in a dual- or N-codebook framework, and attempt to include deeper features.

In regard to the use of the L2 primal support vector machine as classifier, it proved to be quite more efficient to train than the conventional support vector machine implementation. Though a drawback still remains in an undeniable necessity for parameter optimization. Concerning computational intensity, one might consider the learning rate used (0.0000001) in combination with the number of iterations (2000).

We hope to develop an open framework[¶] which combines not only easy modularity and flexibility of combining a number of codebooks, but also remains open to recycling of codebooks, exporting and importing centroids derived from previously trained codebooks, to allow the user to avoid the need to re-train the entire codebook.

# References


[1] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International Journal of Remote Sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[2] G. Csurka, C. Bray, C. Dance, and L. Fan, "Visual categorization with bags of keypoints," *Workshop on Statistical Learning in Computer Vision, ECCV*, pp. 1–22, 2004.

[3] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng, "Text detection and character recognition in scene images with unsupervised feature learning," in *Proceedings of the 2011 International Conference on Document Analysis and Recognition*, pp. 440–445, 2011.

[4] A. Coates, H. Lee, and A. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (G. Gordon, D. Dunson, and M. Dudk, eds.), vol. 15 of *JMLR Workshop and Conference Proceedings*, pp. 215–223, JMLR W&CP, 2011.

[5] C. LeCun, Y. Cortes, "The mnist database of handwritten digits," 1998.

[6] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.


[¶]The framework is currently available online at https://github.com/JonathanMaas/nCodebooks . Special thanks to M.Groefsema and A.Wanningen for cooperative team effort, and development of this project.

[7] O. Surinta, M. F. Karaaba, T. K. Mishra, L. R. Schomaker, and M. A. Wiering, "Recognizing handwritten characters with local descriptors and bags of visual words," in *Proceeding of the 16th International Conference, Engineering Applications of Neural Networks (EANN), Rhode, Greece*, pp. 255–264, Springer International Publishing, 2015.

[8] O. Surinta, L. Schomaker, and M. Wiering, "A comparison of feature and pixel-based methods for recognizing handwritten Bangla digits," in *12th International Conference on Document Analysis and Recognition*, pp. 165–169, 2013.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, 2005.

[10] K. Takahashi, S. Takahashi, Y. Cui, and M. Hashimoto, *Remarks on Computational Facial Expression Recognition from HOG Features Using Quaternion Multi-layer Neural Network*, pp. 15–24. Cham: Springer International Publishing, 2014.

[11] J. Arrspide, L. Salgado, and M. Camplani, "Image-based on-road vehicle detection using cost-effective histograms of oriented gradients," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 1182 – 1190, 2013.

[12] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, (Berkeley, Calif.), pp. 281–297, University of California Press, 1967.

[13] H. Gao, W. Chen, and L. Dou, "Image classification based on support vector machine and the fusion of complementary features," *CoRR*, vol. abs/1511.01706, 2015.

[14] A. Wanningen, "A primal support vector machine for handwritten character recognition using a bag of visual words," *Bachelor's thesis, University of Groningen*, 2016.

[15] B. Graham, "Fractional max-pooling," *CoRR*, vol. abs/1412.6071, 2014.

[16] M. Groefsema, "Deep architectures using the bag of words model for object and handwritten character recognition," *Bachelor's thesis, University of Groningen*, 2016.