MULTI-LAYER ORTHOGONAL VISUAL CODEBOOK FOR IMAGE CLASSIFICATION

Xia Li[1], Yan Song[2], Yijuan Lu[3], and Qi Tian[1]

[1]Department of Computer Science, University of Texas at San Antonio, TX, 78249
[2]Department of EEIS, University of Science and Technology of China, Hefei, 230027, China
[3]Department of Computer Science, Texas State University, San Marcos, TX, 78666

ABSTRACT

Recently, Bag of Visual Words (BoW) model has shown its success in image classification and retrieval. The key idea behind the BoW model is to quantize the continuous high-dimensional space of image features (*e.g.* SIFT [1]) to a manageable visual codebook. The quality of the visual codebook has an important impact on BoW-based methods. Different from the existing techniques, such as Kernel codebook [4] and Sparse Coding [5], we propose a novel multi-layer orthogonal codebook (MOC) generation approach. It aims at explicitly reducing quantization error by generating codebook from the feature space that is orthogonal to the existing codebook. Furthermore, we propose two simple schemes to apply the new codebook with spatial pyramid matching (SPM) [2] on image classification. Experimental results show the efficiency of our proposed MOC generation method, and the performance on image classification is comparable to the state-of-the-art techniques.

Index Terms—Image classification, visual codebook, orthogonal space, Spatial Pyramid Matching

1. INTRODUCTION

The BoW model is an important component of recent image classification and retrieval applications. The key idea behind the BoW model is to quantize the continuous high-dimensional space of low-level features (*e.g.* SIFT [1], HOG [3]) to a manageable visual codebook. This is typically achieved by grouping the features into clusters by unsupervised learning algorithms, *e.g. K*-Means clustering. Then by treating each cluster center as the visual word in the codebook, one can map each extracted feature descriptor to its nearest cluster center, and represent each image as a histogram over visual codebook.

The quality of codebook has an important impact on BoW-based methods. In *K*-Means clustering approach, the codebook is generated by minimizing the total reconstruction error or expected distortion, and the original features are hard quantized into their nearest cluster centers or visual words. Here, reconstruction error refers to the distance between a descriptor and its assigned visual word. Besides *K*-Means clustering, there also exist various codebook generation methods, such as Kernel Codebook (KC) [4] and Sparse Coding (SC) [5]. These methods apply the so-called soft quantization scheme, which implicitly reduces the reconstruction error by representing the original feature with multiple weighted visual words. The image classification results on several benchmark datasets, such as Caltech-101 [6] and 15 Scenes [7, 8], show the superiority of soft quantization. However, the computation complexity of soft-quantization methods is expensive, especially for visual codebook with tens of thousands visual words. In [9], Jegou, *et al*. propose the Hamming Embedding for large-scale image search, which provides binary signatures to refine the match based on visual words. In [11], Zhou, *et al*. propose a non-linear feature transformation on local appearance descriptor, termed as super-vector, which exploits the residual vector information obtained from the vector quantization (VQ).

Different from existing methods, in this paper, we propose a novel multi-layer orthogonal codebook (MOC) generation algorithm. The basic idea is to generate multiple codebooks by incorporating information from the feature space that is orthogonal to the one spanned by the existing codebook. More specifically, the primary codebook can be first created using *K*-Means algorithm, and then the orthogonal information that cannot be represented by the current codebook is derived. The new codebook is further generated to explicitly compensate the reconstruction error. This process can be iterated several times to generate the multi-layer codebook. It is worth noting that other codebook generation methods can also be applied together with MOC. Compared with existing methods [9, 10, 11], our proposed scheme has the following merits: 1) Decompose the large-scale codebook into multiple orthogonal ones. 2) Explicitly minimize the reconstruction error. 3) No complex learning or clustering stage, which makes it applicable to large-scale computer vision systems. 4) A simple scheme can be combined with many state-of-the-art methods.

Furthermore, to utilize other components in image classification framework, *i.e*. MAX-pooling and linear SVM [12], we evaluate the performance of MOC by the following proposed methods: 1) *Kernel Fusion* (KF), linear combination of the kernels derived from each layer of codebook. 2) *Soft Weighting* (SW), similar as KC, it makes use of the soft quantization scheme. However, the efficiency

is greatly improved by codebook decomposition. Experiments on various benchmark datasets demonstrate that our proposed method achieves consistent performance improvements over the traditional *K*-Means codebook, and is comparable to the state-of-the-art methods, such as KC [4] and ScSPM [5].

In the following sections, we will introduce the multi-layer orthogonal codebook in Section 2. In Section 3, the kernel fusion and soft weighting methods are discussed in the image classification framework. The experimental results are shown in Section 4. Finally, discussion and future work are concluded in Section 5.

2. MULTI-LAYER ORTHOGONAL CODEBOOK

In this section, we introduce our proposed multi-layer orthogonal codebook (MOC) generation method in two steps. First, the standard *K*-Means algorithm is introduced, and the reconstruction error of hard quantization is derived. Then the multi-layer orthogonal codebook approach is proposed to explicitly reduce the reconstruction error.

2.1. *K*-Means Clustering

We denote the training set for codebook generation as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N] \in \mathbf{R}^{D\times N}$, which contains N randomly sampled D-dimensional local descriptors. The codebook is denoted as $\mathbf{B} \in \mathbf{R}^{D\times M}$, where M is the codebook size. , and the corresponding code for a given local descriptor is denoted as $\boldsymbol{\alpha}$, $\boldsymbol{\alpha} \in \mathbf{R}^{M}$. As shown in [15], standard *K*-Means clustering method solves the following least square problem:

$$\begin{aligned} &\min_{\mathbf{B},\boldsymbol{\alpha}} \sum_{i=1}^{N} \left\| \mathbf{x}_i - \mathbf{B}\boldsymbol{\alpha}_i \right\|^2 \\ &\text{s.t.} \quad \left\| \boldsymbol{\alpha}_i \right\|^0 = 1, \left\| \boldsymbol{\alpha}_i \right\|^1 = 1, \boldsymbol{\alpha}_i \geq 0, \forall i \end{aligned} \tag{1}$$

where $\|\cdot\|^2$ denotes the L_2-norm of vectors. The constraints in equation (1) mean that the code $\boldsymbol{\alpha}_i$ is an indication vector with only one non-zero entry corresponding to the closest cluster center $\mathbf{x}_i$ belongs to.

From equation (1), we can see that the codebook with fixed size is generated by minimizing the overall reconstruction error on training set $\mathbf{X}$, and a local descriptor $\mathbf{x}_i$ is hard quantized by finding its nearest center $\mathbf{b}_i$ ($\mathbf{b}_i = \mathbf{B}\boldsymbol{\alpha}_i$) with reconstruction error $\|\mathbf{x}_i - \mathbf{b}_i\|^2$, as illustrated in Figure 1(a). Since the optimal codebook size is unknown, the reconstruction error of each local descriptor $\mathbf{x}_i$ can hardly be reduced by simply increasing the codebook size. It motivates our multi-layer orthogonal codebook generation method.

2.2. Multi-layer Orthogonal Codebook Generation

Let us construct a L-layer codebook $\{\mathbf{B}^{(l)}\}$, l=1,2,…,L, where $\mathbf{B}^{(l)} = \left[\mathbf{b}_1^{(l)}, \mathbf{b}_2^{(l)}, \ldots, \mathbf{b}_{M^{(l)}}^{(l)}\right] \in \mathbf{R}^{D\times M^{(l)}}$ is the l-th layer codebook containing $M^{(l)}$ visual words. Each local descriptor $\mathbf{x}$ in a given image is represented by a series of codes $\boldsymbol{\alpha}^{(1)}$,…, $\boldsymbol{\alpha}^{(L)}$, where $\boldsymbol{\alpha}^{(l)}$ is the $M^{(l)}$-dimensional code corresponding to the l-th layer codebook. The training set for generating each layer codebook is denoted as $\mathbf{X}^{(l)}$, l=1,…,L, where $\mathbf{X}^{(l)} = [\mathbf{x}_1^{(l)}, \ldots, \mathbf{x}_N^{(l)}] \in \mathbf{R}^{D\times N}$.

The training set for the first layer codebook generation consists of randomly sampled local descriptors. We use equation (1) to generate the first layer codebook $\mathbf{B}^{(1)}$. The training set for l-th layer codebook generation, where $l \geq 2$, is composed of the orthogonal residues derived from the upper layer as follows,

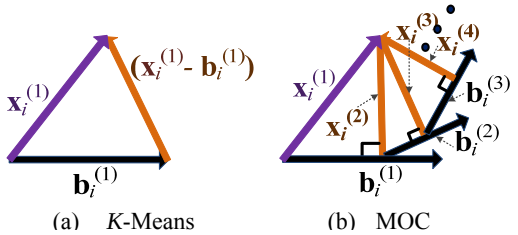
$$\begin{aligned} &\mathbf{x}_i^{(l)} = \min_{\gamma_i^{(l)}} (\mathbf{x}_i^{l-1} - (\mathbf{B}^{(l-1)}\boldsymbol{\alpha}_i^{(l-1)})\gamma_i^{(l)}) \\ &\text{s.t.} \quad 0 \leq \gamma_i^{(l)} \leq 1 \end{aligned} \tag{2}$$

where l=2,…,L. $\mathbf{x}_i^{(l)}$ is orthogonal to (l-1)-the layer word $\mathbf{b}_i^{(l-1)}$ for $\mathbf{x}_i^{(l-1)}$, and it cannot be represented by $\mathbf{b}_i^{(l-1)}$, as the example shown in Figure 1(b).

Given the training set $\mathbf{X}^{(l)}$, where $l \geq 2$, we can generate the l-th layer codebook by the same *K*-Means criteria:

$$\begin{aligned} &\min_{\mathbf{B}^{(l)},\boldsymbol{\alpha}^{(l)}} \sum_{i=1}^{N} \left\| \mathbf{x}_i^{(l)} - \mathbf{B}^{(l)}\boldsymbol{\alpha}_i^{(l)} \right\|^2 \\ &\text{s.t.} \quad \left\| \boldsymbol{\alpha}_i^{(l)} \right\|^0 = 1, \left\| \boldsymbol{\alpha}_i^{(l)} \right\|^1 = 1, \boldsymbol{\alpha}_i^{(l)} \geq 0, \forall i \end{aligned} \tag{3}$$

where $\boldsymbol{\alpha}_i^{(l)}$ is the code for $\mathbf{x}_i^{(l)}$ on the l-th layer codebook $\mathbf{B}^{(l)}$. It subjects to the same constraints as $\boldsymbol{\alpha}_i$ in equation (1). $\mathbf{B}^{(l)}$ is able to improve existing codebook capacity as it explicitly minimizes individual descriptor reconstruction error.



(a) *K*-Means (b) MOC

Figure 1. Illustration of MOC generation. $\mathbf{x}_i^{(1)}$ is a local descriptor in the training set of $\mathbf{B}^{(1)}$. It is mapped to its nearest neighbor words $\mathbf{b}_i^{(1)}$, $\mathbf{b}_i^{(2)}$, … respectively on $\mathbf{B}^{(1)}$, $\mathbf{B}^{(2)}$, … In (a) its reconstruction error is $\|\mathbf{x}_i - \mathbf{b}_i^{(1)}\|^2$. In (b), $\mathbf{x}_i^{(2)}$, $\mathbf{x}_i^{(3)}$, … are its orthogonal residues on $\mathbf{B}^{(2)}$, $\mathbf{B}^{(3)}$, … of MOC.

We can iteratively construct multi-layer codebook until the reconstruction error is small enough to be ignored. The lower layer explicitly reduces the reconstruction error from the upper layer and hence descriptors can be represented with much less information loss.

3. APPLICATION IN IMAGE CLASSIFICATION

We apply MOC in image classification framework with spatial pyramid matching (SPM) [1]. SPM is a spatial extension of BoW model to incorporate the location and shape information of objects. Assume an image is partitioned into R sub-regions in different scales with N_r denoting the set of descriptor indices within sub-region r. By some coding method, code $\boldsymbol{\alpha}_i$ is obtained for a given descriptor $\mathbf{x}_i$. Let $f(\cdot)$ denotes a pooling strategy *i.e.*, average or max pooling [12]:

$$\mathbf{h}_r = f(\{\boldsymbol{\alpha}_i\}_{i\in N_r}), r = 1, \ldots, R \tag{4}$$

where $\mathbf{h}_r$ is the histogram within sub-region r, and then all the histograms are concatenated together to form the final vector representation of the whole image:

$$\mathbf{H}^T = \left[\mathbf{h}_1^T \cdots \mathbf{h}_R^T\right] \quad (5)$$

At last, a specific kernel, *i.e.*, non-linear intersection or linear kernel [12], is computed as:

$$\mathbf{k}_{\mathbf{I}_1,\mathbf{I}_2} = g(\mathbf{H}_{\mathbf{I}_1}, \mathbf{H}_{\mathbf{I}_2}) \quad (6)$$

where $g()$ is a kernel function. $\mathbf{k}_{\mathbf{I}_1,\mathbf{I}_2}$ is the kernel value for any two images $\mathbf{I}_1$ and $\mathbf{I}_2$. And we use LIBSVM [15] package to train the classifiers with one-versus-all strategy.

3.1. Kernel Fusion

To make use of MOC, an intuitive method is kernel fusion (KF), which treats the multiple codebooks separately. Specifically, a descriptor $\mathbf{x}_i$ is quantized by its nearest neighbor visual word on each layer codebook, and a series of codes $\boldsymbol{\alpha}_i^{(1)}, \ldots, \boldsymbol{\alpha}_i^{(L)}$ are obtained and concatenated: $\boldsymbol{\alpha}_i^{\mathrm{T}} = [(\boldsymbol{\alpha}_i^{(1)})^{\mathrm{T}}, \ldots, (\boldsymbol{\alpha}_i^{(L)})^{\mathrm{T}}]$. Then following equation (4) - (6), we can obtain a series of kernel values for image $\mathbf{I}_1$ and $\mathbf{I}_2$: $\mathbf{k}_{\mathbf{I}_1,\mathbf{I}_2}^{(1)}, \ldots, \mathbf{k}_{\mathbf{I}_1,\mathbf{I}_2}^{(L)}$ respectively with each layer codebook. Then we compute the final kernel value for $\mathbf{I}_1$ and $\mathbf{I}_2$ as follows:

$$\mathbf{K}_{\mathbf{I}_1,\mathbf{I}_2} = \frac{1}{L}\sum_{l=1}^{L} \mathbf{k}_{\mathbf{I}_1,\mathbf{I}_2}^{(l)} \quad (7)$$

3.2. Soft Weighting

Actually, the multiple layers codebooks are equally weighted in KF scheme. However, each layer of codebook might play a different role, and the case is not identical for all descriptors. Hence, we propose a soft weighting (SW) scheme to adjust weights of codes from different layers of codebook. In SW, the descriptor $\mathbf{x}$ is first hard quantized into code $\boldsymbol{\alpha}^{(l)}$ on codebook $\mathbf{B}^{(l)}$. Then the weight for each selected word, $\mathbf{b}_i^{(1)}, \ldots, \mathbf{b}_i^{(L)}$ is estimated by minimizing the following reconstruction error:

$$\begin{aligned} &\min_{\mathbf{w}_i} \left\| \mathbf{x}_i - \left[\mathbf{B}^{(1)}\boldsymbol{\alpha}_i^{(1)}, \mathbf{B}^{(2)}\boldsymbol{\alpha}_i^{(2)}, \ldots, \mathbf{B}^{(L)}\boldsymbol{\alpha}_i^{(L)}\right]\mathbf{w}_i \right\|^2 \\ &\text{s.t.} \quad \mathbf{w}_i^T\mathbf{1} = 1, \forall i \end{aligned} \quad (8)$$

where $\mathbf{w}_i \in \mathrm{R}^L$, it consists of weight for each selected word form L-layer codebook. Equation (8) is a least square optimization problem, which can be solved much more efficiently compared with the sparse coding algorithm used in ScSPM [5]. In practice, the best performance is achieved when L equals to 3.

Then we replace those non-zero elements in $\boldsymbol{\alpha}_i^{(1)}, \ldots, \boldsymbol{\alpha}_i^{(L)}$ respectively with the corresponding word weight in $\mathbf{w}_i$. For the descriptor $\mathbf{x}_i$, we obtain the final code $\boldsymbol{\alpha}_i$ by concatenating $\boldsymbol{\alpha}_i^{(1)}, \ldots, \boldsymbol{\alpha}_i^{(L)}$ together.

We can also assign multiple words to a descriptor as LLC [13] on each layer codebook to explore the correlations between words on multiple layers. The proposed SW scheme can still be applied. The best performance is achieved by using 2 nearest neighbor words on each layer (SW+2NN). At last, we follow equation (4) - (6) with max pooling and linear kernel SVM classifier.

4. EXPERIMENTS

We show our experimental results on two benchmark datasets including object dataset: Caltech-101 [6] with 9144 images in 101 classes, and scene dataset: 15 Scenes [7, 8] with 4485 images falling into 15 categories. Our implementation uses only a single descriptor type: SIFT as in [1, 4, 5]. The SIFT descriptors are extracted from 16×16 pixel patches densely sampled over a grid with spacing of 6 pixels. All images are preprocessed into gray scale and resized to less than 300×300 pixels with aspect ratio kept. In SPM framework, 21=16+4+1 sub-regions at three resolution levels are used.

Following the experiment setting of [1,4,5], we report our final classification accuracy as the mean and standard deviation of average accuracies from ten runs experiments with different randomly selected training and test images.

4.1. Reconstruction Error Comparisons

In this experiment, we evaluate the quality of codebook in terms of reconstruction error. Specifically, a subset of 100k descriptors are randomly sampled from Caltech101 for generating codebooks. We measure the percent of descriptors which have reduced reconstruction error when codebook size changes from M_1 to M_2, and the decrease percent of total reconstruction error of all descriptors. The experiment results are shown in Table 1. From the first two columns of Table 1, we can see that simply increasing codebook size from 128 to 256 and 512, the total reconstruction error decreases 8% and 15% respectively, and only 70-80% descriptors have decreased individual reconstruction error. However, our 2-layer 128+128 codebook reduces the total error by 24% and 90% descriptors have decreased individual reconstruction error. It demonstrates that MOC is more effective in reducing reconstruction error.

Table 1. Reconstruction error. We measure the percent of descriptors which have reduced reconstruction error when codebook size changes from M_1 to M_2, and the decreased percent of reconstruction error of all descriptors. 128+128 refers to a 2-layer codebook with size 128 on each layer.

codebook size: M1 vs. M2	128 vs. 256	128 vs. 512	128 vs. 128+128
# of descriptors (%)	72.06%	84.18%	**91.22%**
total error decrease (%)	8.42%	15.56%	**24.55%**

4.2. Image Classification

Experiment 1: *K*-Means vs. MOC generation

In this experiment, we show the effectiveness of our MOC generation method compared with standard *K*-Means for image classification task. The classification accuracies on Caltech 101 dataset with different codebooks are shown in

table 2. It can be seen that the SW method with 2-layer codebook outperforms the *K*-Means method. In our implementation, a 2-layer codebook with size 256 consists of two codebooks each with size 128. Hence, it means decomposing a single codebook into a 2-layer codebook with half of the size on each layer improves the classification performance. For other datasets, similar conclusion can be drawn. Since there are many possible combinations on MOC, we will further study the effect of codebook size on each layer in the future.

Table 2. Classification accuracy (%) comparision of *K*-Means and MOC generation on Caltech101 dataset with 30 training images per class.

codebook size	256	512	1024	2048
single layer	67.68	68.22	68.36	67.39
2-layer SW	67.94	70.11	71.08	71.45

Experiment 2: Comparisons with the state-of-the -art

In this experiment, we compare our proposed method with several state-of-the-art works. Their performance with single feature type is reported in [1, 4, 5, 13].

As shown in Table 3, we can see that on Caltech 101 dataset, all our proposed methods (*i.e.* KF, SW and SW+2NN) outperform SPM [1] and KC [4]. Compared with LLC [13], the experiments on Caltech101 also show that our proposed SW+2NN method outperforms LLC. Furthermore, SW+2NN method can achieve the accuracy comparable to the ScSPM [5], but our proposed MOC generation method is much more efficient than sparse coding method [14]. On 15 Scenes dataset, our methods outperform all the other ones.

Table 3. Classification accuracy (%) comparisons on Caltech-101 with 15 and 30 training images per class, and 15 Scenes with 100 training images per class. Numbers in parentheses are codebook sizes. For MOC, it shows the size on each layer. *The result is obtained by repeating LLC method with SIFT feature and *K*-Means codebook. 73.44% is reported in [13] using HoG feature.

Dataset	Caltech101 15tr	Caltech101 30tr	15 Scenes 100tr
SPM[1]	56.40 (200)	64.60 (200)	81.40±0.5 (1024)
KC[4]	-	64.14±1.18	76.67±0.39
ScSPM[5]	67.0±0.45 (1024)	73.2±0.54 (1024)	80.28±0.93 (1024)
LLC[13]	65.43 (2048)	*71.63±1.2 (2048)	-
KF	60.66±0.7 (3-layer 512)	69.28±0.8 (3-layer 512)	**83.38±0.2** (2-layer 1024)
SW	64.48±0.5 (3-layer 512)	71.60±1.1 (3-layer 512)	**82.27±0.6** (3-layer 1024)
SW+2NN	65.90±0.5 (2-layer 1024)	72.97±0.8 (2-layer 1024)	**82.54±0.5** (2-layer 1024)

5. CONCLUSIONS AND FUTRUE WORK

In this paper, we propose to generate multi-layer codebook from the orthogonal space that cannot be represented by the existing codebook. Our proposed MOC generation method is a simple yet efficient method. With multi-layer orthogonal codebook, the better representation of each local descriptor can be obtained in terms of reconstruction error. Furthermore, we propose two schemes, kernel fusion and soft weighting, to apply MOC on image classification task. Experimental results on several benchmark datasets validate the superior performance of our approach.

In future work, we will study the weighting scheme to apply MOC on image classification task. Also, we are interested in applying MOC on large-scale image retrieval.

6. ACKNOWLEGEMENT

This work is supported in part by NSF Grant IIS 1052581 and Google Faculty Research Award to Dr. Qi Tian, in part by Anhui Provincial Natural Science Foundation under grant 090412056 to Dr. Yan Song, and in part by Texas State University Research Enhancement Program Grant 2011 to Dr. Yijuan Lu respectively.

7. REFERENCES

[1] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, pp. 91-110, 2004.

[2] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *CVPR*, pp. 2169 – 2178, 2006.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *CVPR*, pp. 886-893, 2005.

[4] J. Gemert, J. Geusebroek, C. Veenman, and A. Smeulders, "Kernel codebooks for scene categorization," *ECCV*, pp. 696-709, 2008.

[5] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," *CVPR*, pp. 1794-1801, 2009.

[6] 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," *CVPR* Workshop on Generative-Model Based Vision, 2004.

[7] L. Fei-Fei and P. Perona, "A Bayesian hierarchical model for learning natural scene categories," *CVPR*, pp. 524-531, 2005.

[8] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *IJCV*, 2001.

[9] H. Jegou, M. Douze, and C. Schmid, "Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search," *ECCV*, pp. 304-317, 2008.

[10] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," *CVPR*, pp. 3304-3311, 2010.

[11] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using Super-Vector coding of local image descriptors," *ECCV*, 2010.

[12] Y. Boureau, F. Bach, Y. LeCun, J. Ponce, "Learning mid-level features for recognition," *CVPR*, pp. 2259-2566, 2010.

[13] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," *CVPR*, pp. 3360-3367, 2010.

[14] J. Yang, K. Yu, and T. Huang, "Efficient highly over-complete sparse coding using a Mixture Model," *ECCV*, pp. 113-126, 2010.

[15] C.-C. Chang and C.-J. Lin. *LIBSVM:a library for support vector machines,* 2001. Software available at http://www.csie.ntu.edu.tw/-cjlin/libsvm.