Image features classification using Bayesian model

Vojtěch Dorňák[1,2], Martin Čermák[1,2,a)], Marek Pecha[2] and Lukáš Pospíšil[1]

[1]*Department of Mathematics, Faculty of Civil Engineering, VSB-TU Ostrava, Czech Republic*
[2]*Department of Applied Mathematics, FEECS, VSB-TU Ostrava, Czech Republic*

[a)]Corresponding author: martin.cermak@vsb.cz

Abstract. This paper compares two conventional classification methods, namely Support Vector Machine and the Bayesian model. The data used for the classification is projected onto a space of significant features using the Scale Invariant Feature Transform technique.

Introduction

In this paper, we introduce the Bayesian model and the Support Vector Machine classification techniques. We experiment with these techniques on classifying photographs of cats and dogs from the Oxford-IIIT Pet Dataset [1], transformed into a space of significant features using the Scale Invariant Feature Transform technique [2].

It has been shown in [3], that classifying the transformed data using the Support Vector Machine is viable option. In [4], these classification techniques have been explored on multiple different datasets. In this paper, we compare the results of the two classification techniques.

Scale Invariant Feature Transform

Scale Invariant Feature Transform (SIFT) was proposed by David Lowe in 2004 [2]. It is a local feature extractor which describes an area around a few selected keypoints by the means of a vector called a descriptor. Such descriptors are invariant to scale, rotation, illumination and to affine transformations.

Keypoint detection

Let the scale space representation of an image $L(x, y, \sigma)$ be the convolution of the input image with the Gaussian kernel $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$. The scale-invariant key-points are selected as extrema of the scale normalized Laplacian of Gaussian (LoG):

$$\triangle_{norm} L(x, y, \sigma) = \sigma \left(\frac{\partial^2 L(x, y, \sigma)}{\partial x^2} + \frac{\partial^2 L(x, y, \sigma)}{\partial y^2} \right). \tag{1}$$

As determination of LoG would be time-consuming, it is approximated by Difference of Gaussian (DoG). The DoG is computed by subtracting two adjacent scale space representations of the image.

This is done across different scales, where the scales are constructed by progressively convolving the original image with the Gaussian kernel.

Each candidate pixel is compared with the 8 surrounding pixels as well as with the 9 pixels in the scale above and the 9 pixels in the scale below. If the candidate pixel intensity is lower or higher than the the 26 surrounding pixels, it is selected as a key-point.

To detect sub-pixel location of extrema, the DoG is interpolated using a quadratic Taylor expansion at each candidate. The keypoint is discarded, if the offset from the keypoint is greater than 0.5 in any dimension, as this indicates that the extremum is close to a better key-point.

Keypoints located along edges are also considered poorly located, and therefore are discarded.

Keypoint description

The keypoint descriptors for the object must be almost identical across different scales, rotations, illuminations and other transformations.

To ensure descriptor invariance to rotation, the keypoint orientation needs to be determined. It is selected as a dominant gradient magnitude, from an orientation histogram.

The descriptor is constructed from a window of the size 16×16 pixels around the keypoint. This window is rotated according to the previously selected orientation of the keypoint. The window is divided into 16 (4 × 4) sub-windows. In each sub-window, an 8 bin histogramm weighted by the gradient magnitudes, is created. These histohgrams form a descriptor vector.

The SIFT extractor provides us with varied number of descriptors for each image. As we require each image to be represented by a single vector, we apply the Bag of Words technique.

Bag of Words represents each sample in a dataset as a multiset of its words. In our case, the words are a categorization of descriptors from all images. The categorization is done using k-means clustering.

The Support Vector Machines

The Support Vector Machine is a supervised learning model designed for binary data classification.

Let $T := \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), ..., (\boldsymbol{x_n}, y_n)\}$, be the training dataset, where n is the number of the samples, $\boldsymbol{x_i} \in \mathbb{R}^m$, $i \in \{1, 2, \dots, n\}$, is the sample and $y_i \in \{-1, 1\}$ is the label related to the sample $\boldsymbol{x_i}$. The classification model is represented by the means of the hyperplane

$$H : \boldsymbol{\omega}^T \boldsymbol{x} - \widetilde{b} = 0, \tag{2}$$

where $\boldsymbol{\omega} \in \mathbb{R}^m$ is the normalized normal vector of the hyperplane H, and $\widetilde{b} = \frac{b}{\|\omega\|} \in \mathbb{R}$ is the bias from the origin.

By augmenting the vector $\boldsymbol{\omega}$ and each sample $\boldsymbol{x_i}$ with an additional dimension, so that $\widehat{\boldsymbol{\omega}} \leftarrow \begin{bmatrix} \boldsymbol{\omega} \\ B \end{bmatrix}$, $\widehat{\boldsymbol{x_i}} \leftarrow \begin{bmatrix} \boldsymbol{x}_i \\ \beta \end{bmatrix}$, where $B \in \mathbb{R}$, and $\beta \in \mathbb{R}^+$ is a user defined variable, the bias $\widetilde{b}$ is included in the problem. This is called the no-bias classification [5].

The hyperplane $H : \widehat{\boldsymbol{\omega}}^T \widehat{\boldsymbol{x}} = 0$ can be found using a following constrained optimization problem in the primal formulation:

$$\arg\min_{\widehat{\omega}, \xi_i} \frac{1}{2}\|\widehat{\omega}\| + \frac{C}{p}\sum_{i=1}^{n} \xi_i^p \quad \text{s.t.} \quad \begin{cases} y_i(\widehat{\boldsymbol{\omega}}^T \widehat{\boldsymbol{x_i}}) \geq 1 - \xi_i, \\ \xi_i \geq 0 \text{ if } p = 1, i = 1, ..., n, \end{cases} \tag{3}$$

where $\xi_i = \max(0, 1 - y_i(\widehat{\boldsymbol{\omega}}^T \widehat{\boldsymbol{x_i}}))$ is the hinge loss function, $C \in \mathbb{R}^+$ is a penalty, that penalizes misclassification error. The penalty c can be found using grid-search combined with cross-validation.

Exploiting the Lagrange duality and evaluating the Karush-Kuhn-Tucker conditions for (3), we obtain the $l1$-loss $l2$-regularized SVM formulation

$$\arg\min_{\lambda} \frac{1}{2}\boldsymbol{\lambda}^T \boldsymbol{H} \boldsymbol{\lambda} - \boldsymbol{\lambda}^T \boldsymbol{e} \quad \text{s.t.} \quad \boldsymbol{o} \leq \boldsymbol{\lambda} \leq C\boldsymbol{e}, \tag{4}$$

for $p = 1$, and the $l2$-loss $l2$-regularized SVM formulation

$$\arg\min_{\lambda} \frac{1}{2}\boldsymbol{\lambda}^T \left(\boldsymbol{H} + C^{-1}\boldsymbol{I}\right) \boldsymbol{\lambda} - \boldsymbol{\lambda}^T \boldsymbol{e} \quad \text{s.t.} \quad \boldsymbol{o} \leq \lambda, \tag{5}$$

for $p = 2$.

$\boldsymbol{H} = \boldsymbol{Y}^T \boldsymbol{G} \boldsymbol{Y}$, $\boldsymbol{G} = \boldsymbol{X}^T \boldsymbol{X}$ denotes the Gramian matrix, $\boldsymbol{X} = \begin{bmatrix} \widehat{\boldsymbol{x}}_1 & \dots & \widehat{\boldsymbol{x}}_n \end{bmatrix}$, $\boldsymbol{Y} = diag(\boldsymbol{y})$, $\boldsymbol{y} = [y_1,\ y_2,\ \dots, y_n]^T$, $\boldsymbol{e} = [1, 1, \dots, 1]^T \in \mathbb{R}^n$, $\boldsymbol{o} = [0,\ 0,\ \dots,\ 0]^T \in \mathbb{R}^n$.

In the $l2$-loss $l2$-regularized SVM formulation, the Hessian matrix $\boldsymbol{H}$, regularized by the matrix $c^{-1}\boldsymbol{I}$ is symetric positive definite, therefore, this optimization problem should be more stable.

Bayesian Model

This classifier is suitable for classifying data represented by a stochastic vector. The BoVW data can be easily transformed into such vector. Instead of each component of the BoVW vector representing the number of key-points in the respective category, the component in our new vector represents the probability of key-points belonging to the respective category.

Let $x_t \in X := \{\mathsf{x}_1, \dots \mathsf{x}_{K_x}\}, t = 1, \dots, T$ be the input variables and let $y_t \in \mathsf{Y} := \{\mathsf{y}_1, \dots, \mathsf{y}_{K_Y}\}$ be output categorical variables. Let us denote the stochastic data vector $\Pi_{xt} \in \mathbb{R}^{K_x}, t = 1, \dots, T$, where T is the number of samples, K_x is the size of BoVW. Let the vector $\Pi_{yt} \in \mathbb{R}^{K_y}$ be a vector of probabilities, with which Π_{xt} belongs to each category, and K_y the number of categories.

Given a stochastic data vector Π_x, we can describe the transformation $\mathbb{R}^{K_x} \to \mathbb{R}^{K_y}$ using a matrix $\Delta \in \mathbb{R}^{K_y, K_x}$:

$$\Delta_{ij} = P(y_t = \mathsf{y}_i | x_t = \mathsf{x}_j), i = 0, \dots, K_y, j = 0, \dots, K_x, \tag{6}$$

where Π_x^n is the n-th element of Π_x, similar to Π_y^n, and the matrix Δ is a left stochastic matrix.

The determination of the optimal Δ^* can be written as

$$\Delta^* = \arg\min_{\Delta \in \Omega_\Delta} \sum_{t=1}^{T} \mathrm{dist}(\Pi_{yt}, \Delta\Pi_{xt}), \tag{7}$$

where Ω_Δ is a set of left stochastic matrices. The $\mathrm{dist}(\Pi_{yt}, \Delta\Pi_{xt})$ is calculated as Kullback-Leiber divergence [6]:

$$\mathrm{dist}(\Pi_{yt}, \Delta\Pi_{xt}) = -\sum_{i=1}^{K_y} \Pi_{yt}^i \ln \frac{[\Delta\Pi_{xt}]_i}{[\Pi_{yt}]_i} = -\sum_{i=1}^{K_y} [\Pi_{yt}]_i (\ln[\Delta\Pi_{xt}]_i - \ln[\Pi_{yt}]_i). \tag{8}$$

For the optimization, the term $\ln[\Pi_{yt}]_i$ is constant, therefore it can be ignored:

$$\mathrm{dist}(\Pi_{yt}, \Delta\Pi_{xt}) \propto -\sum_{i=1}^{K_y} [\Pi_{yt}]_i \ln[\Delta\Pi_{xt}]_i. \tag{9}$$

The analytical solution for this problem does not exist. However, $-\ln()$ is a convex function and $\Delta\Pi_{xt}$ is a convex combination, thus Jensen's inequality can be used:

$$-\sum_{i=1}^{K_y} [\Pi_{yt}]_i \ln[\Delta\Pi_{xt}]_i \leq -\sum_{i=1}^{K_y} [\Pi_{yt}]_i (\sum_{j=1}^{K_x} [\Pi_{xt}]_j \ln(\Delta_{ij})) = -\sum_{i=1}^{K_y} \sum_{j=1}^{K_x} [\Pi_{yt}]_i [\Pi_{xt}]_j \ln \Delta_{ij}. \tag{10}$$

From this estimation, we get an approximated optimization problem

$$\Delta^* = \arg\min_{\Delta \in \Omega_\Delta} -\sum_{t=1}^{T} \sum_{i=1}^{K_y} \sum_{j=1}^{K_x} [\Pi_{yt}]_i [\Pi_{xt}]_j \ln \Delta_{ij}, \tag{11}$$

where $\Omega_\Delta = \{\Delta \in [0,1]^{K_y, K_x}, \forall j \in \{1, 2, \dots, K_x\} : \sum_{i=1}^{K_y} \Delta_{ij} = 1\}$ is a feasible set of left stochastic matrices. The problem can be solved analytically, which gives us the optimal Δ^* with components

$$\Delta^*_{\hat{i}\hat{j}} = \frac{\sum_{t=1}^{T} [\Pi_{yt}]_{\hat{i}} [\Pi_{xt}]_{\hat{j}}}{\sum_{i=1}^{K_y} \sum_{t=1}^{T} [\Pi_{yt}]_i [\Pi_{xt}]_{\hat{j}},}. \tag{12}$$

Another approach to finding the optimal Δ^* is optimizing (7) without using the Jensen inequality. Since the feasible set Ω_Δ is a closed convex set, the problem can be solved numerically using the Spectral Projected Gradient method [7].

We compare both approaches (the analytical solution using Jensen inequality and the numerical solution) in our experiments.

Data Classification

To test our extraction-classification pipeline, we use the Oxford-IIIT Pet Dataset [1]. The dataset contains over $7,000$ photograps of different cat and dog breeds. The dataset provides a trimap for each photograph, allowing us to cut out the animal from the background. This allows us to concentrate only on classifying the data relevant to the animal. We assign 90% of the data to the training subset and 10% to the testing subset.

We run the presented experiments on the Salomon supercomputer [8] at IT4Innovations. As the underlying SVM solver, we use PermonSVM [9]. The Bayesian Model using the Jensen inequality is implemented in Python, while the Bayesian Model solved using the Spectral Projected Gradient method is trained using an Octave code.

Results

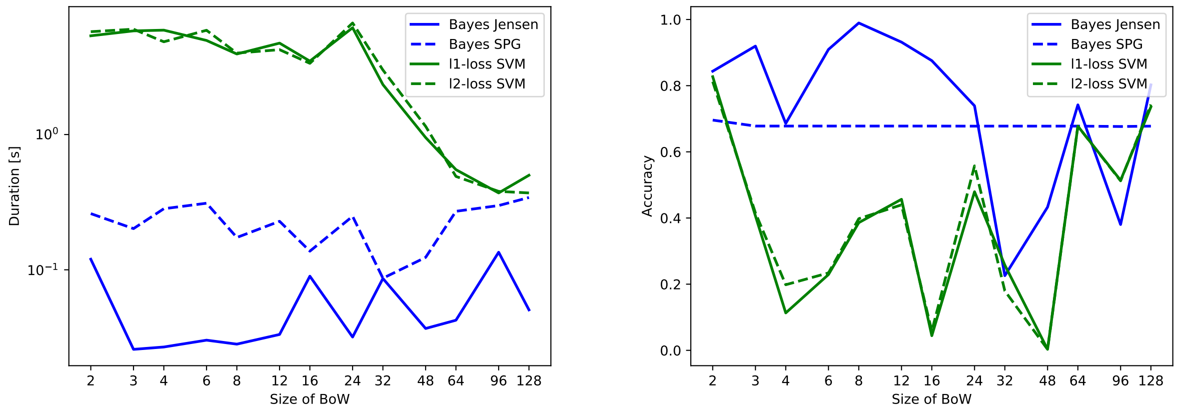

FIGURE 1. The training duration and accuracy for different sizes of BoW.

From the graphs FIGURE 1 we can observe that for the lower BoW sizes both versions of the Bayes classifier perform better in regards to accuracy and the duration of training. However, with the larger BoW sizes, both the *l*1-loss and the *l*2-loss regularized SVM performs similarly to the Bayes classifier.

Conclusion and Acknowledgement

The performance of the Bayesian model shows significant improvement, compared to the performance of the SVM. Further results will be presented at ICNAAM 2021.

This contribution has been completed thanks to the financial support provided to VSB-Technical University of Ostrava by the Czech Ministry of Education, Youth and Sports from the budget for conceptual development of science, research and innovations for the 2021 year.

REFERENCES

[1] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *IEEE Conference on Computer Vision and Pattern Recognition* (2012).

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International journal of computer vision **60**, 91–110 (2004).

[3] V. Dorňák, M. Čermák, M. Pecha, and L. Pospíšil, "SIFT feature extraction applied in SVM classification," (2020 (accepted)).

[4] V. Dorňák, "Algorithms used for classifying visual signals using methods for extracting significant features," Diploma thesis, VŠB - Technical University of Ostrava 2021, supervised by Martin Čermák.

[5] M. Pecha and D. Horák, “Analyzing l1-loss and l2-loss support vector machines implemented in PERMON toolbox,” in *International Conference on Advanced Engineering Theory and Applications* (Springer, 2018), pp. 13–23.

[6] S. Kullback and R. A. Leibler, “On information and sufficiency,” The annals of mathematical statistics **22**, 79–86 (1951).

[7] E. G. Birgin, J. M. Martinez, and M. Raydan, “Nonmonotone spectral projected gradient methods on convex sets,” SIAM Journal on Optimization **10**, 1196–1211 (2000).

[8] IT4Innovations: National Supercomputing Center, VSB-Technical University of Ostrava, Salomon cluster documentation – hardware overview, 2017.

[9] V. Hapla, D. Horák, and M. Pecha, PermonSVM, `http://permon.vsb.cz/permonsvm.htm` (2017).