Kernel Methods for Machine Learning

Kaggle Challenge

Atallah Rayen, Schoonaert Antoine
MVA Master

April 2024

1 Introduction

The aim of this project is to create a model that can effectively classify color images into ten classes using **kernel methods implemented from scratch**. These images come from the CIFAR10 database, on which pre-processing has been carried out. The images represent 10 different classes, such as airplanes or cars. In total, there are $5,000$ train images and 2000 test images of size $32 \times 32 \times 3$ in this database. A sample of the images with their associated classes is available in the appendix to figure 1.

2 Feature extractors

2.1 Histogram of Oriented Gradients

Histogram of oriented gradients (HOG) were introduced by Dalal and Triggs [1]. This method provides features based on image gradients. The technique involves counting occurrences of gradient orientation in localized parts of an image. In practice, the image is divided into cells and the contrast of each cell is normalized. Each cell then has a histogram of the gradients present, and the combination of these creates the final features of the image. This method focuses on gradient intensity and orientation rather than location.

2.2 Scale Invariant Feature Transform

This method, called Scale Invariant Feature Transform (SIFT) and introduced by Lowe [2], can be used to obtain image features. This method was created to match or mismatch two images with different viewpoints. Features are invariant to rotation and scale. To achieve this, SIFT determines the keypoints precisely so that they can be compared with other images. These keypoints can also be used to classify images.

2.3 Kernel Descriptors

We followed the methodology outlined in the paper [3] to compute both gradient and color kernels. By incorporating these kernels into our feature extraction pipeline, we were able to generate rich low-level descriptors that capture both gradient and color information from images. These descriptors serve as a foundational step for the subsequent stage of processing, which is classification

2.4 Fisher Vector

The Fisher Vector (FV) [4] is a powerful representation method, particularly for image classification tasks. It combines generative modeling with discriminative learning, offering higher accuracy and efficiency compared to traditional Bag of Visual Words (BoV) representations. It is derived from a generative model, typically a Gaussian Mixture Model, which captures the distribution of local image features (HOG or SIFT in our case). Mathematically, the FV is computed as the gradient of the log-likelihood of the data on the model's GMM parameters λ:

$$G_X^\lambda = \nabla_\lambda \log u_\lambda(X)$$

Based on the FIM, a similarity measurement between two samples X and Y using the Fisher Kernel is defined by:

$$K_{FK}(X,Y) = G_X^T F_\lambda^{-1} G_Y^\lambda$$

where $F_\lambda = E_{x\sim u_\lambda}[G_X^\lambda (G_X^\lambda)^T]$ is the Fisher Information Matrix. The Normalized Fisher Vector ϕ_X^λ of sample X is obtained by normalizing the gradient vector:

$$\phi_X^\lambda = L_\lambda G_X^\lambda$$

where L_λ is the Cholesky decomposition of F_λ^{-1}, such that $F_\lambda^{-1} = L_\lambda^T L_\lambda$.
In summary, the Fisher Vector encodes statistical information about the feature distribution and captures deviations of the sample from the generative model.

3 Experiments

3.1 Data Augmentation

Given the limited amount of data available, we undertook data augmentation to increase the size of our dataset while maintaining balance. We achieved this by vertically flipping every training image and rotating each image by a random angle between -30 and 30 degrees, resulting in a total of 15,000 augmented training images. This approach has demonstrated improvements in our results on the leaderboard.

3.2 Classifiers

3.2.1 SVM with Sequential Minimal Optimization

The first classification method we used was SVM. Given that the database consists of 5,000 images of size $32 \times 32 \times 3$, implementing an SVM with a "classical" algorithm is very consuming in terms of computational resources. To overcome this problem, we implemented an algorithm called Sequential Minimal Optimization (SMO), presented by Platt [5]. SMO breaks down this large QP problem into several smaller QP problems. This means that the size of the memory required is linear with the size of the training database. To move on from this binary classification algorithm to a multiclass classification algorithm, we created as many binary classifiers as there were classes and then used the one vs all method.

3.2.2 Kernel Ridge Classifier

The second method is kernel ridge regression. We set up as many regressions as we had classes, and set up one vs. all to obtain the final predicted class, by taking the corresponding regression giving the biggest predicted value.

3.3 Kernels

Here is the list of kernels used in features extraction (Kernel Descriptors) and classification :

- Linear kernel: $K(x, y) = x^T y$
- Gaussian kernel: $K(x, y) = \exp(-\|x - y\|^2)$
- Laplacian RBF kernel:
  $K(x, y) = \exp\left(-\frac{\sum_i |x_i - y_i|}{\sigma^2}\right)$
- Sublinear RBF kernel:
  $K(x, y) = \exp\left(-\frac{\sum_i |x_i - y_i|^{0.5}}{\sigma^2}\right)$
- Hellinger kernel: $K(x, y) = \sum \sqrt{xy}$

4 Results

- **Retained features:** We experimented with various local features, including HOG, SIFT, gradient and color kernel descriptors, and Fisher vector applied on them to form a global feature representation. However, the final model only uses HOG features and a combination of HOG with Fisher vector.
- **Computational complexity:** Due to the high dimensionality of our features, we applied kernel PCA with a linear kernel before feeding them to the classifiers. Additionally, for our final submissions, we used our own implementation of SMO. However, during our local testing, we used another implementation of the same algorithm obtained from https://github.com/itsikad/svm-smo due to its faster execution. While we prioritized improving accuracy over optimizing our SVM implementation's speed, we acknowledge the need for future optimizations.
- **Hyperparameter tuning:** We conducted hyperparameter tuning using 5-fold cross-validation. The tuned parameters include the choice of kernels for Kernel Ridge Classifier (KRC) and Support Vector Machine (SVM), kernel's gammas, number of components for both used features in kernel PCA, and regularization parameters.
- **Ensemble learning:** Our final model is constructed through voting among three predictions: HOG + SVM with Gaussian kernel (γ = 0.23), HOG + KRC with Gaussian kernel (γ = 1.2), and HOG + Fisher Vector + SVM with Gaussian kernel (γ = 1.5). In case of ties, the decision is taken from the first model, which appears to be less prone to overfitting compared to the others, even though their performances on both train and test are very comparable. This ensemble approach achieved a final public accuracy of 0.605 on the test set.

References

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

[2] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94

[3] L. Bo, X. Ren, and D. Fox, “Kernel descriptors for visual recognition,” 01 2010, pp. 244–252.

[4] J. Sánchez, T. Mensink, and J. Verbeek, “Image classification with the fisher vector: Theory and practice,” *International Journal of Computer Vision*, vol. 105, 12 2013.

[5] J. Platt, “Sequential minimal optimization : A fast algorithm for training support vector machines,” *Microsoft Research Technical Report*, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:577580

5 Appendix

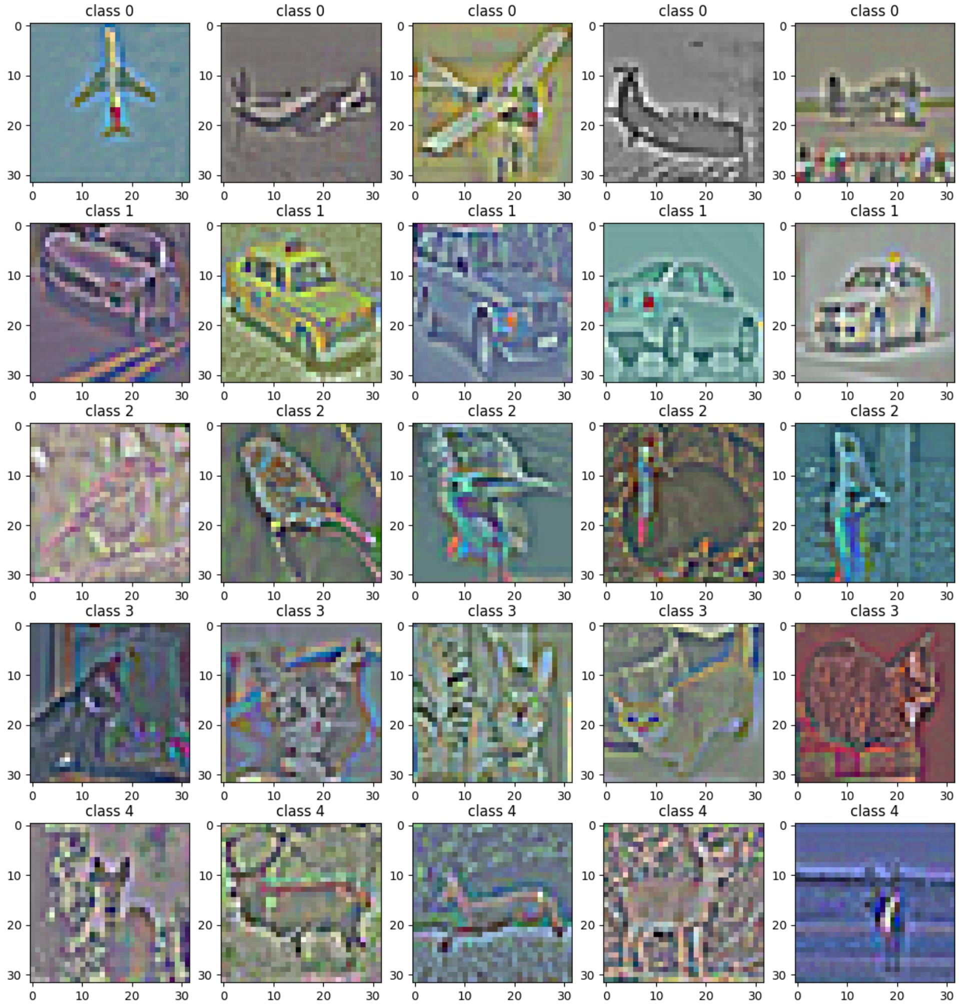

Figure 1: Samples from some classes