Offline 1000-Class Classification on a Smartphone

Yoshiyuki Kawano and Keiji Yanai
The University of Electro-Communications, Tokyo
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182-8585 Japan
{kawano-y,yanai}@mm.inf.uec.ac.jp

Abstract

In this demo, we propose an offline large-scale image classification system on a smartphone. The proposed system can classify 1000-class objects in the ILSVRC2012 dataset in 0.270 seconds. To implement a 1000-class object classification system, we compress the weight vectors of linear classifiers, which leads only slight performance loss.

1. Introduction

Due to recent rapid progress of smartphones such as iPhone and Android phones, they have obtained enough computational power for real-time image recognition. Currently, a quad-core CPU is common as a smartphone's CPU, which is almost equivalent to a PC's CPU released several years ago in terms of performance. Taking advantage of high computational power of latest smartphones, standalone real-time object recognition systems on a smartphone has become possible [4, 5], which can classify 100-class foods. However, due to memory resource limitation on a smartphone, the number of classes to be classified was 100 at most.

In this demo, we propose an offline large-scale image classification system on a smartphone, which can classify 1000-class objects in the ILSVRC2012 dataset. It carried out all the processing steps on a smartphones without any communication to external servers. To implement a 1000-class object classification system, we compress the weight vectors of linear classifiers, which leads only slight performance loss. The system implemented on a latest quad-core smartphone takes only 0.270 seconds for a 1000-class recognition.

2. Proposed System

The basic processing flow is the same as the state-of-the-art large-scale image recognition [8] and real-time mobile image recognition [4]. Firstly, we extract local features such as HOG patches and color patches, secondly code them into Improved Fisher Vectors (IFV) [7] with Spatial Pyramid Matching (SPM) [6] after applying PCA to each of the local features, and thirdly classify coded vectors with linear classifiers in a one-vs-rest manner.

The problem to implement a 1000-class classification system on a smartphone is a large amount of weight vectors of trained linear classifiers. When using FV, SPM and multiple features, the dimension of feature vectors tends to be high. In case of this work, the total dimension of feature vectors is 17,920. To represent weight vectors of linear classifiers for 1000 classes in the "float" representation which requires four bytes for one real value, the total amount of memory for them is 71.7MB, which is too much for a smartphone. In this work, to reduce it, we compress the weight vectors. Note that in our work, recognition on 1000 categories is carried out on a smartphone, while training on 1000 categories is performed on PCs.

As local features in the demo system, we use densely sampled RootHOG patches and Color patches. RootHOG is an element-wise square root of the L1 normalized HOG, which is inspired by "RootSIFT" [1]. Color patches are the same as [4]. After extracting local features, according to Improved Fisher Vectors (IFV) [7], we code them into IFV.

For training, we use AROW [2] as an online learning method. Although training is carried out on cluster PCs, the amount of training data is very large, and then, we compress training samples with Product Quantization (PQ) [3] following to [8]. Only the sample is uncompressed, when it is needed to update the parameters.

For classification on a smartphone, the problem is that 1000-class classification with one-vs-rest linear classifiers with high dimensional Fisher Vector requires a large amount of classifier weights. To solve it, we adopt a scalar-based quantization method for weight vectors of linear classifiers. In the demo system, we compress each element of

the weight vectors into 4 bits, which enables us to reduce the required memory to 1/8. To classify images in a one-vs-rest manner, only relative descending order of the output values of the classifiers is important. Since we adopt scalar-based quantization, it is enough for classification to compute a dot product between a compressed weight vector and a FV-coded feature vector.

3. Implementation

Before FV coding, we reduce the dimension RootHOG and Color patches into 32-d and 24-d, respectively. Then, we code them into FV with the GMM with 64 Gaussians and Spatial Pyramid level 1 (1x1+2x2). As results, the total dimensions of FV are 10240 for RootHOG-FV and 7680 for Color-FV, respectively. For feature fusion, we simply add two output values of linear classifiers on each class.

We implement a system as a multi-threaded system for a quad-core smartphone. Refer to Kawano et al. [4], we parallelize feature extraction for RootHOG Patch and Color Patch features. Extraction of local descriptors, applying of PCA with them, encoding them into Fisher Vector with power normalization and L2 normalization, and evaluation of 1000 classifiers are carried out over 2 cores in parallel for each feature. Moreover, in the same way as [4], we compute all the constant values which can be calculated in advance. Regarding the weights of the classifiers, we store two 4-bit compressed vectors into one-byte memory, and separate them on demand in classification time.

4. Experiments

In the experiment, as a large-scale image dataset, we use the ILSVRC2012 dataset which consists of 1000 classes. We evaluate the results of 1000-class classification with the top-5 classification rate, which represents the rate that a ground-truth label is found in the top-5 classes in the descending order of the output values of the classifiers.

With the setting mentioned in the previous section, we obtained 47.9% with original "float" weights and 48.7% with compressed "4-bits" weights for the top-5 rate for 1000-class (Table 1). The required memory for linear classifier weights were 71.7MB for "float" weights and 9.0MB for "4-bit" weights. This shows compression of 1/8 on weight vectors leads only 0.8 point loss, which have not been explored before.

In the experiment, we use Samsung Galaxy Note II (1.6GHz 4-core CPU, 4 threads, Android 4.1) for measuring recognition time for 1000 classes. With 4-core implementation, it took 0.270 seconds in one-time recognition of 1000 classes of ILSVRC.

Table 1. Comparison on the top-5 classification rate and the required memory to store weight vectors.

	float(32bit)	compressed(4bit)
Rate	48.7%	47.9%
Memory	71.7MB	9.0MB

5. Conclusions

In this demo, we proposed an offline large-scale image classification system running on a smartphone. We have proved that mobile large-scale image classification requires no communication to external servers. To realize that, we proposed a scalar-based compression method for weight vectors of linear classifiers. In the experiment, we showed that compressing the weights to 1/8 leaded to only 0.80% performance loss for 1000-class classification with the ILSVRC2012 dataset. For future work, we will examine and analyze weight vector compression more comprehensively and deeply.

References

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, pages 2911–2918, 2012. 1

[2] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. *Machine Learning*, 91(2):155–187, 2013. 1

[3] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. on PAMI*, 33(1):117–128, 2011. 1

[4] Y. Kawano and K. Yanai. Rapid mobile object recognition using fisher vector. In *Proc. of Asian Conference on Pattern Recognition*, 2013. 1, 2

[5] Y. Kawano and K. Yanai. Real-time mobile food recognition system. In *Proc. of CVPR Workshop on Mobile Vision*, 2013. 1

[6] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178, 2006. 1

[7] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010. 1

[8] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR*, pages 1665–1672, 2011. 1