Low-Bit Representation of Linear Classifier Weights for Mobile Large-Scale Image Classification
Yoshiyuki Kawano and Keiji Yanai
The University of Electro-Communications, Tokyo, Japan
PS2–36

Objective

• Implement standalone large-scale object recognition with 1000/10000 classes on a mobile phone.
  • Fisher vector and linear classifier
  • Multi-class classification: one-vs-rest (We are working on DCNN-based mobile object recognition, 4bit is OK currently. We will present it at the other places.)
  • Too many training parameters \((D \cdot C)\)
  • Limited memory and storage
  • 2-bit representation of classifier weights led to only slight performance loss.

Implementation

Local descriptors
• RootHoG - gradient
• Color – moment
  local patch: 16x16 and 24x24
  dense sampling every 8 pixel

Feature vector (coding)
GMM: \(K=64/128/256\)
Spatial Pyramid: \(1x1 + 2x2, (5=5)\)
\((K=128\) with no SP for 10k classification\)
• Color-FV (7860/15360d/30720d)
• HOG-FV (10240d/20480d/40960d)

Linear Classifier
AROW (online learning)
with 1-vs-rest and late fusion
A linear SVM can be used as an alternative.

Dataset
ILSVRC2012(1000), ImageNet 10k

Experimental results

1000-class classification rate with different bits

Performance with the same memory size using different bits and GMM size

10K-class classification results with 2-bit quantization and Product quantization (PQ)

Combination of PQ and 2-bit scalar quantization is more effective.

Difference to PQ: PQ needs to refer a codebook table at evaluation time, while the proposed method does not.

Standalone mobile obj. rec.

Client-side recognition vs. server-side
• anywhere without internet
  • Quick response
  • No problem on server scalability
• Large-size training parameters
• Low computational power

We have resolve it!

Low-bit representation of linear classifier weights

Represent each element of all the weight vectors of all the linear classifiers in low bits by scalar-quantization

Distribution of weights
assignment of weight values

Linear classifier
\[ y = w \cdot x \]

\[ 32 \text{ bit} \Rightarrow 2 \text{ bit} \]

(value range for 2-bit)

The same \(\alpha\) is used for all the weights

For one-vs-rest classification, reconstruction of original weights is not needed.

Hardware selection

Processing time

0.591 sec.
0.330 sec.
0.159 sec.

0.309 sec.
0.0415
0.0309

0.2824
0.2790
0.2975
0.2971

0.4689
0.4824
0.4864
0.4814
0.4788
0.4723
0.4695

128bit, no performance loss.
With 2-bit, slight loss.
With 1-bit, great loss.

Modification of the way of quantization helps improve performance with 2-bit.

Very-high-dimensional weight vectors still exhibit discriminative power after low-bit quantization.
(DCNN has the same characteristics in general.)

\[ \alpha \]