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.)

(D: dim of FV, C:num. of classes)

- Too many training parameters $(D \cdot C)$
- Limited memory and storage

our findings

- **2-bit representation of classifier weights led to only slight performance loss.**

Standalone mobile obj. rec.

Client-side recognition *vs.* server-side

- anywhere without internet
- Quick response
- No problem on server scalability

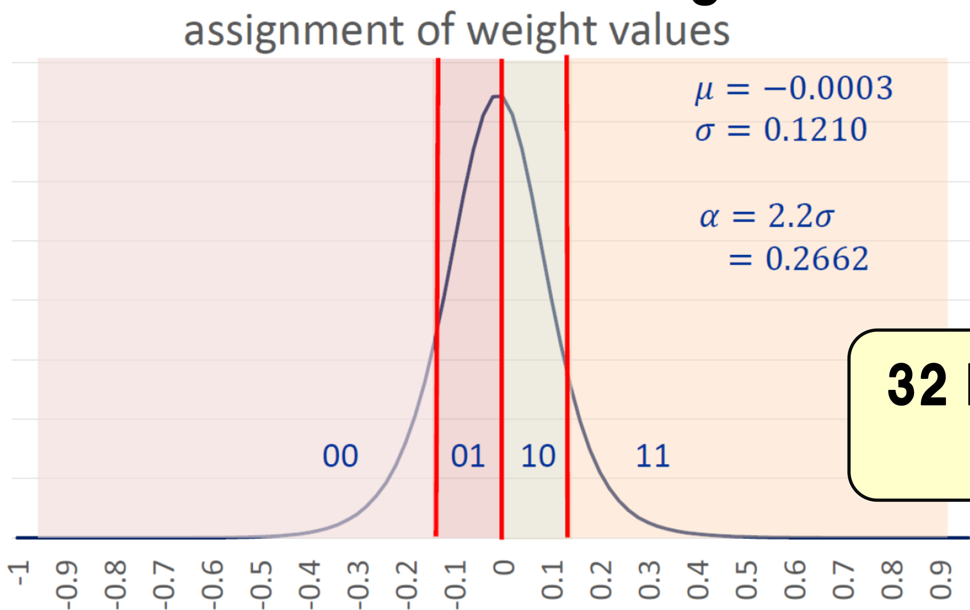
- Large-size training parameters
- Low computational power

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



Linear classifier
$y = \boldsymbol{w} \cdot \boldsymbol{x}$

32 bit ⇒ 2 bit (1/16)

$w_i' = (w_i - w_{avg})/\alpha$ $\quad (\alpha = 2.2\sigma$ (estimated by CV))

$$w_i'' = \begin{cases} 0.999999 & (w_i' \geq 1) \\ w_i' & (-1 < w_i' < 1) \\ -1 & (w_i' \leq -1) \end{cases}$$

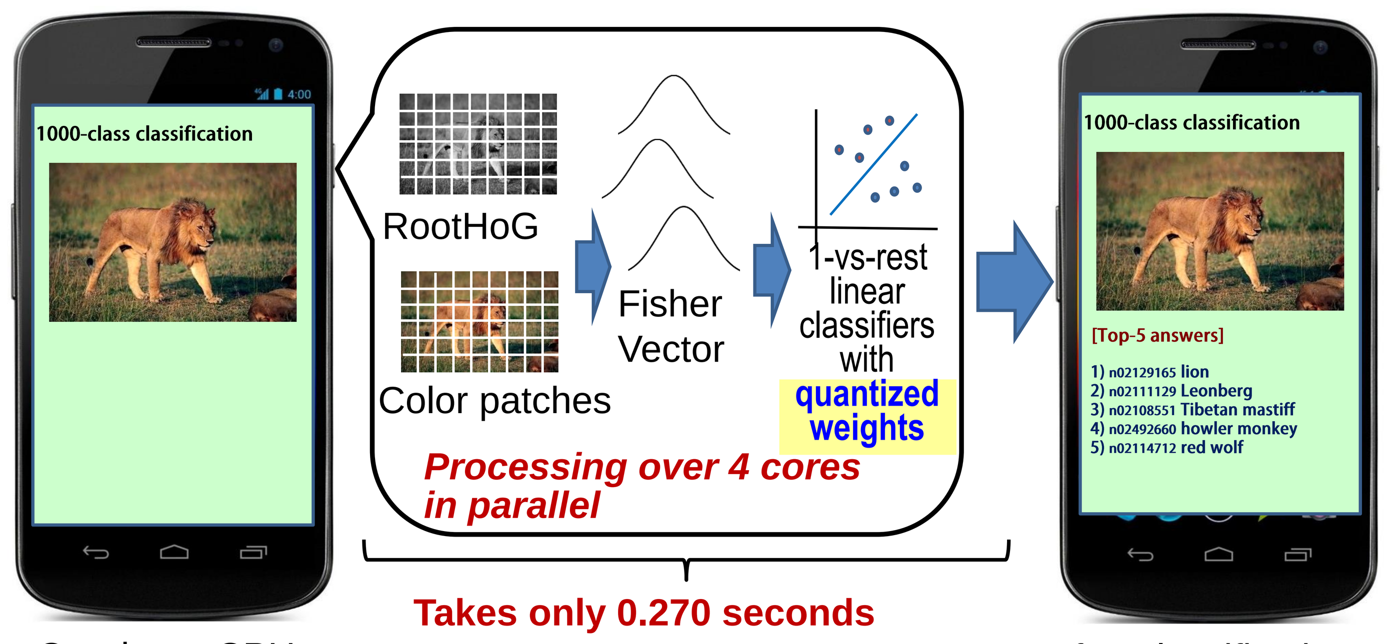
$$w_i''' = \lfloor w_i'' \times 2^{n-1} + 2^{n-1} \rfloor,$$

value	range for 2bit
00	$-\infty < w_i < -\alpha$
01	$-\alpha \leq w_i < 0$
10	$0 \leq w_i < \alpha$
11	$\alpha \leq w_i < \infty$

The same α is used for all the weights

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

Implementation



Quadcore-CPU smartphone

after classification

Multi-threads over quadcore
(Galaxy S5 (2.6GHz, 4core, Android4.4.2))

Local descriptors

- **RootHOG** - gradient
- **Color** – moment

local patch: 16x16 and 24x24
dense sampling every 6 pixel

Feature vector (coding)

GMM: K=64/128/256
Spatial Pyramid: 1x1 + 2x2 (=5)
(K=128 with no SP for 10k classification)

- **Color-FV** (7680d/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.)

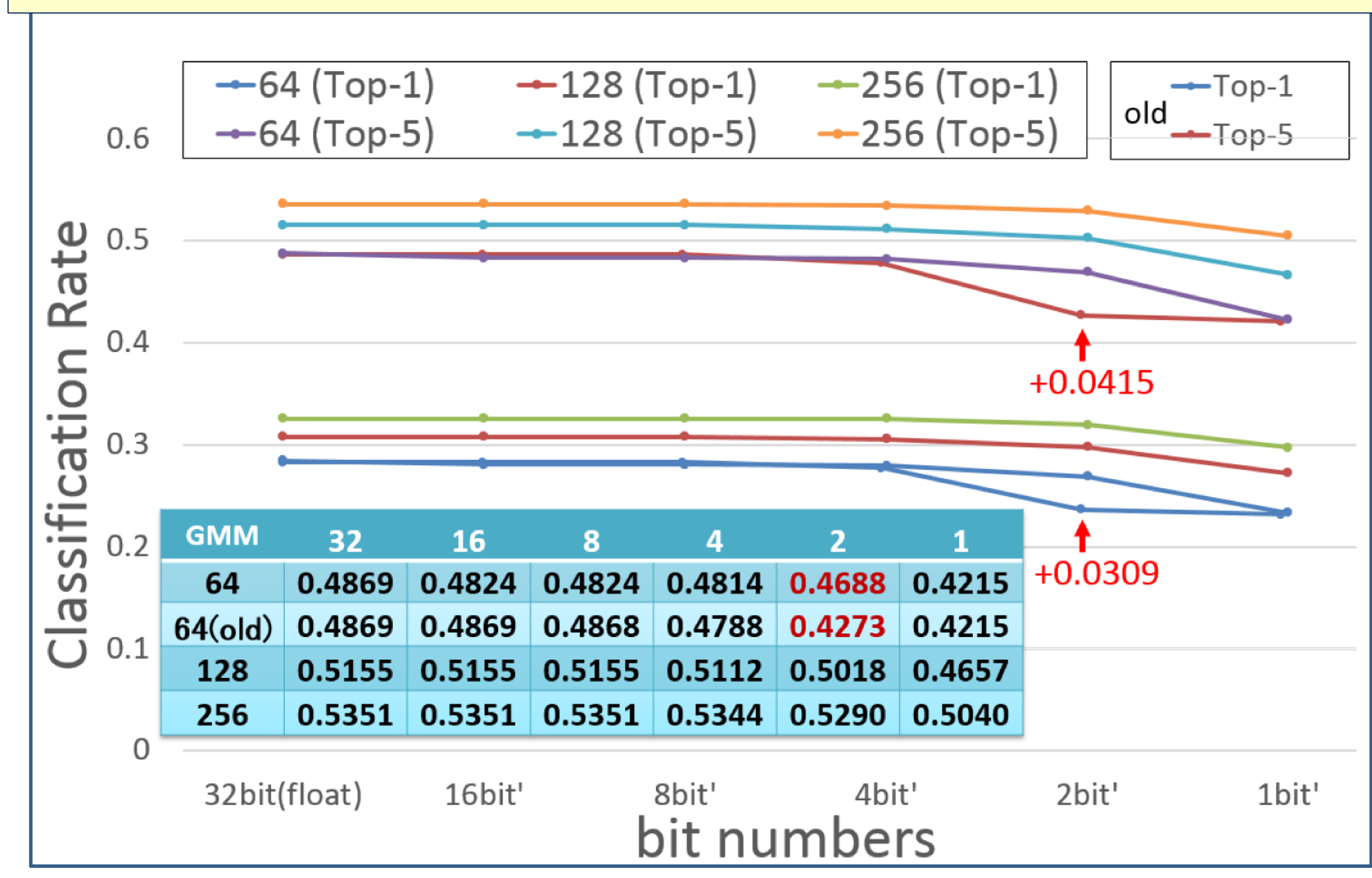
Dataset

ILSVRC2012(1000), ImageNet 10k

# class	GMM	SP level	Color-FV	HoG-FV	Mem. (32bit float)	Mem. (2bit)
1000	64	2	7680	40960	68MB	4.3MB
1000	128	2	15360	20480	136MB	8.5MB
1000	256	2	30720	40960	272MB	17.1MB
10184	128	1	3072	4096	278MB	17.4MB

Experimental results

1000-class classification rate with different bits



GMM	32	16	8	4	2	1
64	0.4869	0.4824	0.4824	0.4814	0.4688	0.4215
64(old)	0.4869	0.4869	0.4868	0.4788	0.4273	0.4215
128	0.5155	0.5155	0.5155	0.5112	0.5018	0.4657
256	0.5351	0.5351	0.5351	0.5344	0.5290	0.5040

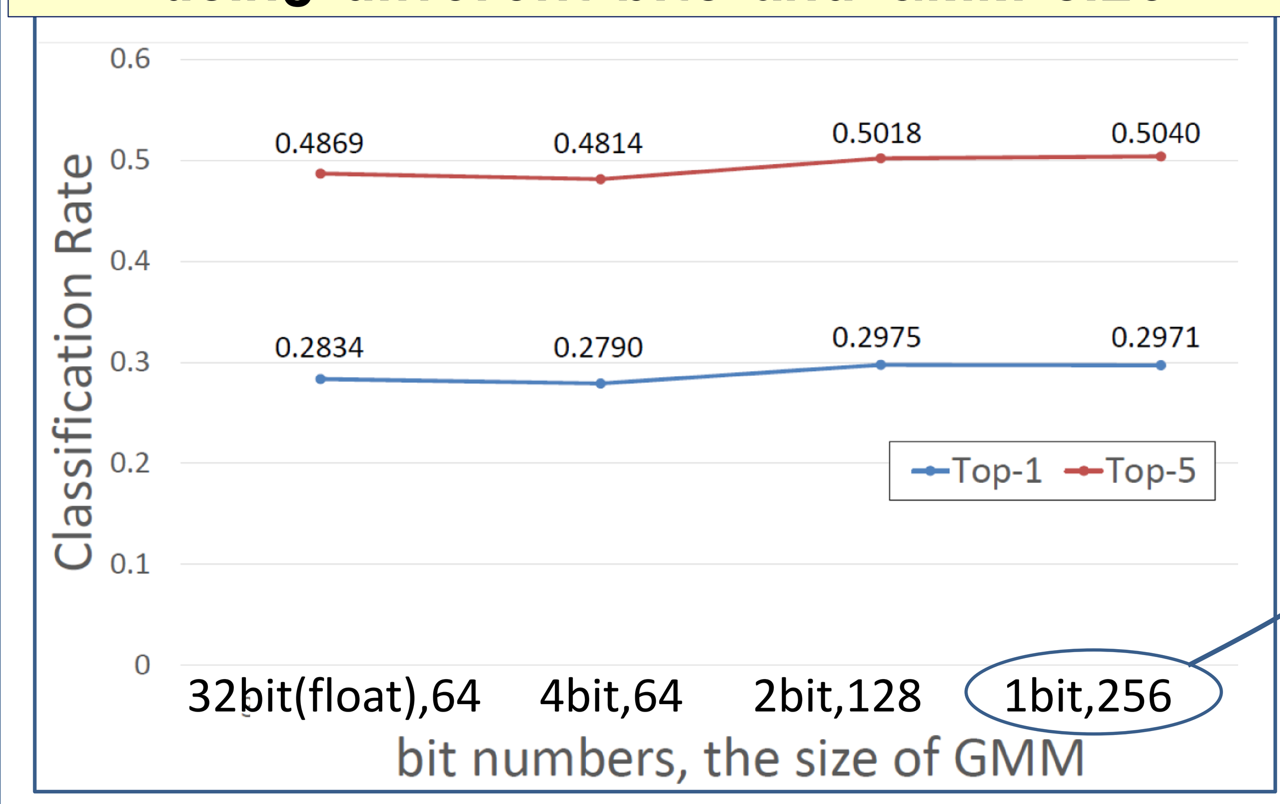
(old[2] : no subtraction of average with fixed $\alpha = 1$)

Until 4-bit, no performance loss.
With 2-bit, slight loss .
With 1-bit, great loss.

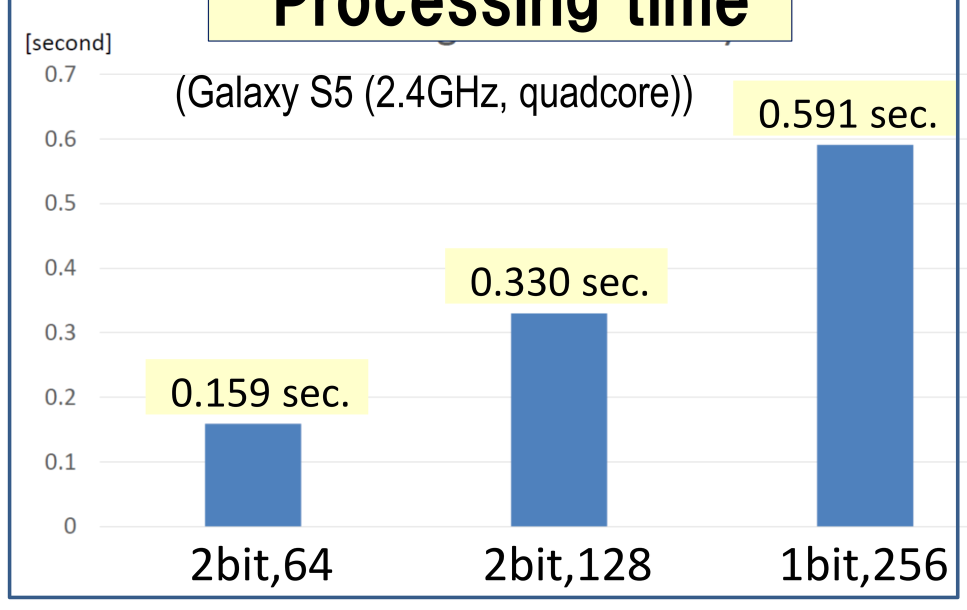
Modification of the way of quantization helps improve performance with 2-bit.

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

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



Processing time



1-bit quantization with 256-GMM achieved the best result.
However, processing time for feature extraction increased much.

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

method	top-1 (%)	top-5 (%)	memory (Mbyte)
no compression	11.83	25.25	292M
scalar	11.42	24.30	18.2M
PQ[3]	10.96	23.85	9.1M + 7.3M (PQCB)
PQ[3] + scalar	10.87	23.75	9.1M + 1.8M (8bit)

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.

[2] Y. Kawano and K. Yanai. ILSVRC on a smartphone. IPSJ Trans. CVA, 6:83–87, 2014.
[3] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128, 2011.