ISSN 0002-306X. Proc. of the RA NAS and NPUA Ser. of tech. sc. 2021. V. LXXIV, N3

UDC 004.852

COMPUTER SCIENCE AND INFORMATICS

DOI 10.53297/0002306X-2021.3.v74-308

A.V. BATUTIN, K.P. LYDA, L.A. KHOJOYAN, J.P. FEDERICO, T.S. KARAMYAN, D.S. KARAMYAN

EVALUATION OF INFERENCE OPTIMIZED IMAGE CLASSIFICATION MODELS ON IOS PLATFORM

In this paper, we evaluate MobileNetV2 and EfficientNet network architectures performance on the iOS platform. Even though both networks are well researched, there is a certain lack of empirical data when it comes to the iOS platform. In this paper, we provide experimental data for top-1 accuracy and inference time, as well as RAM footprint collected on iPhone X and iPhone 12 Pro. The evaluation was performed using CPU, GPU, and Neural Engine (iPhone 12 only). Also, a full int8 quantization evaluation was done for MobileNetV2. Challenges around EfficientNet quantization are also described. The scoring metrics and dataset used to train models were provided as part of Mobile AI 2021 Real-Time Camera Scene Detection Challenge: https://competitions.codalab.org/competitions/28113 .

Keywords: Camera Scene Detection challenge, MobileNetV2, EfficientNet, image classification.

Introduction. In recent years iOS and Android platforms have received a substantial boost in computational resources making them fit for on-device CNN model inference [1]. At the same time, most of the research and evaluation metrics for CNN architectures are centered around top-1 accuracy with little regard for inference time and model size. Mobile model deployment requires both accuracy and inference time optimized architectures. For the evaluation we have used the dataset published at Mobile AI 2021 Real-Time Camera Scene Detection Challenge. The goal of the challenge is to train image classification model which is both accurate and fast.

Model Selection. MobileNet [2] was chosen as a first model to evaluate. This model is based on an inverted residual structure where the shortcut connections are between the thin bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. It is important to remove non-linearities in the narrow layers in order to maintain representational power. This kind of actions improve performance and provide an intuition that led to MobileNet design. The basic building block is a bottleneck depth-separable convolution with residuals. The detailed structure of

this block is shown in [2]. The architecture of MobileNetV2 contains the initial fully convolution layer with 32 filters followed by 19 residual bottleneck layers. ReLU6 activations are used as the non-linearity because of its robustness when used with low-precision computation. The kernel size is always 3 × 3 in this architecture as is standard for modern networks, and utilize dropout and batch normalization during training. With the exception of the first layer, the constant expansion rate is used throughout the network.

With ImageNet top-1 accuracy of 74.7%, it was obvious that getting model accuracy above 96% without significant overfitting would be difficult.

The second was EfficientNet [3], since its architecture is a result of neural architect search for the best model accuracy with minimum FLOPS required. Tan, et al [4], developed a baseline network by leveraging a multi-objective neural architecture search that optimizes both accuracy and FLOPS. Specifically, they use the same search space as (Tan et al., 2019), and use $ACC(m) \times [FLOPS(m)/T]^{w}$ as the optimization goal, where $ACC(m)$ and $FLOPS(m)$ denote the accuracy and FLOPS of model m, T is the target FLOPS and $w = -0.07$ is a hyperparameter for controlling the trade-off between the accuracy and the FLOPS. Since they use the same search space as described in [4], the architecture is similar to Mnas-Net, except our EfficientNet-B0 is slightly bigger due to the larger FLOPS target. Initial experiments on the challenge data set showed little difference between the B2 and B3 variants in validation accuracy. So B2 was chosen as a faster variant. With 80.1% ImageNet top-1 accuracy and FLOPS optimized architecture, it was our prime candidate.

ResNet50 with 76.0% top-1 accuracy is similar to MobileNetv2 but slower. It was chosen to test how much BigTransfer can improve the model accuracy.

The appropriate evaluation of the Models is shown in Table 1.

Table 1

Evaluated Models

Architecture	ImageNet Top-1	FLOPS
MobileNetV2	74.7%	88 M
EfficientNet B2	80.1%	1000 M
ResNet50	76.0%	4100M

Dataset. Camera Scene Detection dataset consisting of images belonging to 30 different classes was used for the model training. This dataset was used for all our experiments. The dataset was divided into:

• Train data: 9897 images of resolution 576 x 384 px from the above 30 classes that were used for training the model;

• Validation data: 600 images of resolution 576 x 384 px. The top-1 accuracy was measured on this subset.

Evaluation Metrics. In this challenge, each submission was validated based on the following two metrics:

• The accuracy of the predictions;

• The runtime of the model on the actual target mobile platform.

The exact scoring formula used in this challenge:

$$Score = \frac{2^{(top1-96.0)+(top3-99.0)}}{C * runtime}, \quad (1)$$

where C is a constant normalization factor that does not depend on the submission. Table 2, Table 3 and Table 4 represent the final, iPhone 12 Pro and iPhone X evaluation results. In those tables 'float' is the default TFLite conversion format and 'int' is the full int8 conversion.

Table 2

Final Results

TFLight Model	Size (MB)	Top-1 (%)	iPhone 12 Pro CPU (s)	Score
MobileNet V2 float	8.6	92.5	0.083	0.0941
MobileNet V2 int	2.7	91.3	0.058	0.0255
EfficientNet float	29.5	96	0.27	3.7037
EfficientNet int	9.1	95.3	0.28	1.3533

Table 3

iPhone 12 Pro Results. Inference time in seconds, RAM and file size in MB

Model	Top-1 (%)	iOS12 CPU (s)	iOS12 GPU (s)	iOS 12 NPU (MB)	iOS 12 CPU RAM (MB)	iOS 12 GPU RAM (MB)
MobileNet V2 float	92.5	0.083	0.016	0.014	200	166
MobileNet V2 int	91.3	0.058	0.093	0.092	160	160
EfficientNet float	96	0.272	-	-	220	-
EfficientNet int	95.3	0.283	-	-	160	-

Table 4

iPhone X Results. Inference time in seconds, RAM and file size in MB

Model	Top-1 (%)	iOS X CPU (s)	iOS X GPU (s)	iOS X CPU RAM (MB)	iOS X GPU RAM (MB)
MobileNet V2 float	92.5	0.095	0.067	170	200
MobileNet V2 int	91.3	0.132	0.133	136	136
EfficientNet float	96	0.432	-	195	-
EfficientNet int	95.3	0.398	-	142	-

We have set C to 1 for our experiments. As the formula shows all the models with top-1 accuracy less than 96% are heavily penalized.

Evaluation environment.

- Tensorflow and Keras were the main frameworks used for model implementation, training, and exporting.
- Both MobileNetV2 and EfficientNet implementations were taken from official Tensorflow repository.
- ImageNet pre-trained weights were used to initialize models.
- TensorFlow Lite image classification iOS example application was used to evaluate models.

Model Optimization and TFLite Conversion. Models were converted to TFlite in 2 formats:

- Float - default TFlite conversion format;
- Int - full int8 conversion.

The evaluation of TFlite models was done on iPhone 12 Pro and iPhone X devices. The evaluation was done on CPU, GPU, and NPU (iPhone 12 only) Memory footprint was measured for the entire iOS app.

EfficientNet TFLite Conversion Issues. During conversion to TFLite format 2 things turned out:

- The Normalization Layer used by the EfficientNet network is not implemented in the quantized version in TFLite.
- The ResizeBilinear Layer, which we used for rescaling input images is not implemented in TFLite.

Therefore, we decided to remove problematic layers and fine-tune such a model. Model without those layers was converting properly to TFLite format.

Results. As expected, EfficientNet B2 has shown the best result during the development phase. It achieved 96% accuracy on a challenge dataset. Its runtime was 10x slower than MobileNet. But MobileNet accuracy on the challenge dataset was only 93%. Thus, EfficientNet scored much higher on the final evaluation.

CoreML does not support EfficientNet execution on GPU or Neural Engine since it uses dynamic-sized tensors. So we could not evaluate EfficientNet running on GPU and NPU.

Final model evaluation was done for float and int8 tflite models running iPhone 12 Pro CPU.

Conclusion and future work.

• GPU seems to be the go-to accelerator for CNN on the iOS platform.

• On-device CNN inference performance is suitable for low-latency real-time applications.

• EfficientNet has a big potential but a workaround for GPU acceleration is required.

• Full int8 quantization yields significantly smaller models with minimal accuracy loss.

REFERENCES

1. AI Benchmark: All About Deep Learning on Smartphones in 2019 / **A. Ignatov, et al** // 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW).– Seoul, Korea (South).– 2019.-P. 3617-3635, doi: 10.1109/ICCVW.2019.00447
2. MobileNetV2: Inverted Residuals and Linear Bottlenecks / **M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen** // 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition.– Salt Lake City, UT, USA, 2018.– P. 4510-4520, doi: 10.1109/CVPR.2018.00474.
3. **Tan M, Le Q.V.** EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks[C] // International Conference On Machine Learning.– 2019.– P. 6105-6114.
4. MnasNet: Platform-aware neural architecture search for mobile / **M. Tan, B. Chen, R. Pang, V. Vasudevan, et al.-** CVPR, 2019.

DataArt (Kiev, Poland, Argentina, Armenia), Yerevan State University. The material is received on 18.06.2021.

Ա.Վ. ԲԱՏՈՒՏԻՆ,Կ.Պ. ԼԻԴԱ, Լ.Ա. ԽՈՋՈՅԱՆ, Խ.Պ. ՖԵԴԵՐԻԿՈ,
Տ.Ս. ՔԱՐԱՄՅԱՆ, Դ.Ս. ՔԱՐԱՄՅԱՆ

ՕՊՏԻՄԱՑՎԱԾ ՊԱՏԿԵՐՆԵՐԻ ԴԱՍԱԿԱՐԳՄԱՆ ՄՈԴԵԼՆԵՐԻ ԱՐԴՅՈՒՔՆԵՐԻ ԳՆԱՀԱՏՈՒՄԸ IOS ՊԼԱՏՖՈՐՄԻ ՎՐԱ

Գնահատվել է MobileNetV2 և EfficNet ցանցերի ճարտարապետությունների արտադրողականությունը iOS պլատֆորմի վրա: Չնայած երկու ցանցերն էլ լավ ուսումնասիրված են, այդուհանդերձ, էմպիրիկ տվյալների որոշակի պակաս կա, երբ խոսքը վերաբերում է iOS պլատֆորմին: Ներկայացվել են փորձարարական տվյալներ մոդելների ճշգրտության և արագագործության վերաբերյալ, ինչպես նաև RAM- ի վերաբերյալ տեղեկություն, որը հավաքվել է iPhone X- ի և iPhone 12 Pro- ի վրա: Գնահատումն իրականացվել է CPU-ի, GPU- ի և նյարդային շարժիչի միջոցով (միայն iPhone 12-ով): MobileNetV2- ի համար ևս կատարվել է int8 քվանտացման ամբողջական գնահատում: Նկարագրվել են նաև EfficNet քվանտացման շուրջ մարտահրավերները: Մոդելները պատրաստելու համար օգտագործված գնահատման չափորոշիչները և տվյալների հավաքածուն տրամադրվել են Mobile AI 2021 իրական ժամանակում տեսախցիկի տեսարանի հայտնաբերման (Mobile AI 2021 Real-Time Camera Scene Detection) մարտահրավերի շրջանակներում. https://competitions.codalab.org/-competitions/28113

Առանցքային բառեր. Camera Scene Detection մարտահրավեր, MobileNetV2, EfficientNet, պատկերների դասակարգում:

**А.В. БАТУТИН, К.П. ЛИДА, Л.А. ХОДЖОЯН, Х.П. ФЕДЕРИКО,
Т.С. КАРАМЯН, Д.С. КАРАМЯН**

ОЦЕНКА ВЫВОДОВ ОПТИМИЗИРОВАННЫХ МОДЕЛЕЙ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ НА ПЛАТФОРМЕ IOS

Дается оценка производительности сетевых архитектур MobileNetV2 и EfficientNet на платформе iOS. Несмотря на то, что обе сети хорошо изучены, существует определенный недостаток эмпирических данных, когда речь идет о платформе iOS. Приводятся экспериментальные данные по точности и времени вывода, а также объем оперативной памяти, собранной на iPhone X и iPhone 12 Pro. Оценка проводилась с использованием центрального процессора, графического процессора и Neural Engine (только для iPhone 12). Кроме того, для MobileNetV2 проведена полная оценка квантования int8. Описаны проблемы, связанные с квантованием EfficientNet. Показатели оценки и набор данных, используемые для обучения моделей, были предоставлены в рамках конкурса Mobile AI 2021 Real-Time Camera Scene Detection Challenge: https://competitions.codalab.org/competitions/28113

Ключевые слова: вызов Camera Scene Detection, MobileNetV2, EfficientNet, классификация изображений.