Image Polarity Detection on Resource-Constrained Devices

Edoardo Ragusa
Christian Gianoglio
Rodolfo Zunino
Paolo Gastaldo
University of Genoa

***Abstract*—Image polarity detection opens new vistas in the area of pervasive computing. State-of-the-art frameworks for polarity detection often prove computationally demanding, as they rely on deep learning networks. Thus, one faces major issues when targeting their implementation on resource-constrained embedded devices. This paper presents a design strategy for convolutional neural networks that can support image-polarity detection on edge devices. The outcomes of experimental sessions, involving standard benchmarks and a pair of commercial edge devices, confirm the approach suitability.**

IMAGE POLARITY DETECTION aims to identify the emotional content expressed in a picture. Such a technology is crucial in the era of pervasive computing, when millions of users interact everyday with smart devices and use social networks [1]. Modern polarity detectors rely on object recognition technologies and embed convolutional neural networks (CNNs). The increasing availability of reliable, pre-trained CNNs for object recognition is a major advantage, as it allows to use transfer learning to shift the focus on polarity detection; as a result, training a polarity detector does not require to collect a large dataset, which is a complex, time-consuming task due to the the so-called *subjective perception problem* [2].

Pre-trained CNNs, however, are most effective for extracting high-level, structured features from images, at a considerably high computational cost. Thus, their implementation on low-power embedded devices is typically quite challenging. A practical polarity-detection framework, in fact, might not require those high-end object recognition models, as it is known that polarity is not uniquely determined by the objects identified in an image [3], [4], [5]. In principle, suitable performances in polarity detection can be achieved by feeding the model with ‘simple’ features inspired directly by the psychology of vision [6]. In the presence of stringent constraints on computational resources, it seems reasonable to rely on CNN architectures that are less computationally demanding, even at the expense of a limited performance in object recognition. Research on that subject [4] recently showed that the CNN architecture (i.e., the pre-trained object recognition model) does affect the overall performance in polarity detection only when using a small dataset for fine tuning.

This paper proposes that a proper design of the CNN architecture for high-level features extraction can lead to embedded implementations of polarity detection with real-time capabilities. The underlying design criteria accomplish the goal of limiting the computational complexity of the architecture without affecting the performances of the polarity detectors significantly.

Contribution

This paper presents a hardware-friendly design model for image polarity detection. The model is supported by a CNN architecture exploiting Depthwise Separable Convolution (DSC) and weight truncation. Two commercial low-power embedded devices, namely, the Movidius Neural Computing Stick and the Nvidia Jetson TX2, provided the platforms for system deployment. Experimental results involved four real word datasets and confirmed the prediction effectiveness of the design criterion.

RELATED WORK

The literature offers a variety of interesting approaches to polarity detection. Recent survey works on this topic [2], [4], [7] concur on the fact that CNN-based models are state-of-the-art in this area. It is worth to mention that several important works addressed image polarity detection before that deep learning changed the approach to image processing [8], [9], [10], [11]. However, currently models based on CNNs represent the state-of-the-art.

According to the transfer-learning paradigm, polarity detection frameworks rely on the low-level features extracted by a CNN trained on object recognition. The implementations mainly differ in a) the adopted CNN, b) the transfer learning technique, and c) the training data domain. Many works have explored the idea of fine-tuning pre-trained object classifiers. In this regard, Campos et. al [12] adopted the *AlexNet* architecture for feature extraction; they carried out an interesting analysis on the effects of layer ablation and layer addition on the eventual accuracy in polarity detection. Other approaches augmented the basic framework by inserting an ontology-based representation on top of object recognition [13]. You et al. [14] proposed a custom architecture, in which a pair of convolutional layers and four fully connected layers were stacked [15].

In some variants of CNN architectures for polarity detection, both the image and a set of additional features formed the network inputs. In the work of Fan et al. [16], the *Vgg_19* architecture processed an image along with its focal channel, to model human attention. Likewise, others works studied the role played by art features [6], context information [17], salience [18], [19], attention mechanisms [20], [21], [22] and combinations of the last two [23]. Rao et al. [24] exploited a faster-RCNN to locate the distinctive parts of an image.

POLARITY DETECTION ON EDGE DEVICES

The deployment of inference models derived from trained networks on edge devices brings about several design issues, as one needs to properly balance prediction accuracy, power consumption, and computational speed. When considering that CNNs are an unavoidable requirement due to their effectiveness, that trade-off is even more demanding because of the inherent complexity of the networks. A reasonable approach goal seems to get a reliable image polarity detector even starting from a relatively weak object detector, i.e., starting from a CNN with moderate complexity.

Interestingly, a recent work [4] empirically proved that such target is achievable. The experiments showed that, when applying fine-tuning strategies and in the presence of large datasets, accuracy of the final polarity detector did not vary significantly with the basic, pre-trained CNN. A possible explanation of this phenomenon is that large datasets are built on top of automatic labeling processes, and therefore are presumably affected by induced noise. Therefore, it seems convenient to design the CNN architecture according to a set of strategies that can reduce the computational cost of a deployed model [25] such as weight factorization, parameter pruning and sharing, low-rank factorization, transferred/compact convolutional filters, and knowledge distillation.

The CNN architectures adopted in the research presented in this paper feature a pair of specific design strategies. First, the proposed method involves Depthwise Separable Convolution (DSC), which mostly proves as the key element of the most successful implementation of CNNs on resource constrained devices [26], [27], [28], [29]. In DSC, a standard convolutional operator is replaced by two separate layers, which represent a factorized version of the original convolution operation. The first layer supports a depth-wise convolution and involves a single convolutional filter per input channel. The second layer is a 1×1 convolution, called point-wise convolution; it extracts a new set of features by working out linear combinations of the input channels. This decomposition remarkably reduces computational costs: given an input of size $h \times w \times d$ and a convolutional layer characterized by q kernels of size $k \times k$, the computational cost C_{sc} of standard convolution is

$$C_{sc} = h \times w \times d \times q \times k \times k. \qquad (1)$$

By contrast, when using the factorized version, the cost C_{DSC} becomes

$$C_{DSC} = h \times w \times d \times (k^2 + q). \qquad (2)$$

which is significantly smaller than (1). Empirical evidence proves that, in spite of this reduction in cost, CNN architectures exploiting DSC can yield object-detection accuracy values that are comparable with those attained by state-of-the-art architectures [26], [27], [28], [29].

Weight truncation is the second design strategy used in the proposed approach, and involves half-precision arithmetic. The eventual outcomes are appealing: lower latency, memory saving, and a decrease in power consumption. On the other hand, quantization errors may seriously affect arithmetic computations. As modern CNNs involves hundreds of layers, error propagation is a major problem that ends up in a lower accuracy of the inference model. Thus, the effects of weight truncation on the deployed model should be carefully evaluated.

In general, both DSC and half-precision arithmetic act as low pass filter, reducing the capability of the model of learning details. This issue, however, might not be an actual disadvantage in the present application, since DSC and half-precision arithmetic might help reduce the effects of overfitting phenomena with the noisy datasets at hand.

EXPERIMENTS

The experimental sessions were designed 1) to assess the accuracy scored by the different image polarity models built upon the different CNN architectures, ad 2) to analyze the memory usage, the power consumption, and the latency of the models when deployed on embedded devices.

This work compares three CNN architectures involving DSC: MobileNet_v1, MobileNet_v2, and MobileNet_v2(1.4). These comparisons were selected because pre-trained models on *Imagenet* were available. This made a fair comparison feasible with state-of-the-art architectures for object recognition such as Res_101, Res_152, Vgg_16 and Vgg_19, which had already proved to be most effective for image polarity detection [4].

Four popular datasets provided the experimental benchmarks: ANP40 [11], MANP40 [15], T4SA [30], and “Twitter” [14]. ANP40 and MANP40 hold collections of images crawled from *Flicker*® repository. Both these datasets are characterized by adjective-noun pairs that have been generated by an automatic system. Although each original dataset held more than 1 million images, the experiments involved pruned versions of the samples [4]: only the 20 most positive and 20 most negative adjective noun-pairs were selected.

The corresponding images were split into two subsets (positive and negative samples). T4SA is a recent dataset of images crawled from *Twitter*® for sentiment analysis; it contains 470,586 images. In this research, images were categorized into three classes according to textual information: positive, negative and neutral. Finally, “Twitter” is a small dataset of images crawled from *Twitter*® and manually labeled by 5 human annotators. In this work, following the experimental setup employed in previous research papers, the five-on-five agreement version of the dataset was taken into account.

Table 1. Datasets.

	Training	Devel.	Test	**Total**
ANP40	9.916	200	400	10.516
MANP40	25.258	3100	3100	31.258
T4SA	368.586	51.000	51.000	470.586
Twitter	-	-	882	882

The checkpoints of the 7 networks trained on the 3 dataset are freely available on the author's website[1].Table 1 summarizes the main details of the used datasets; the columns give the size of the training set, the validation set for model selection (Development), and the test set. It is worth noting that Twitter dataset was only used to test models trained on a different dataset.

Single domain

The first group of experiments evaluated the performances of the image polarity models when trained and tested on data belonging to the same dataset. All architectures were trained using ADAM optimizer with forgetting factor equal to 0.9, second moment of the gradient equal to 0.999, and batch size 32. Early stop mechanism was implemented with validation patience of 5.

Table 2 reports on the result of the experiments. The second column gives, for each single CNN architecture, the average accuracy obtained by the image polarity model on the test set in the bi-class experiments involving ANP40 with 32-bit floating point representation. The value between brackets refer to the difference between such accuracy and the accuracy obtained when adopting 16-bit floating point representation. A positive value indicates that the half-precision model scored a lower accuracy. The third column refers to the bi-class experiments involving MVSO40, while the fourth column refers to the three-class experiments involving T4SA.

Figure 1 depicts the accuracy attained by different classifiers when implemented using half-precision floating point arithmetic. It is worth to note that figure 1 conveys the same information of Table 2 in a different format. In the figure, accuracies are grouped in 3 sets correspond to the 3 datasets. Bars refer to the score of different predictors.

[1]http://sealab.diten.unige.it

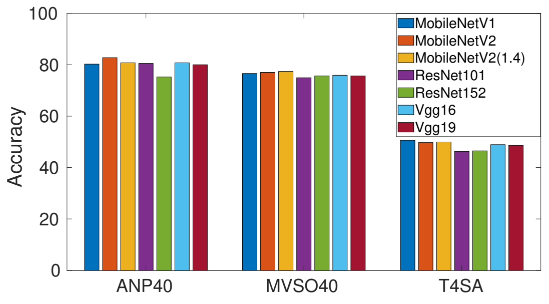

Figure 1. Error rate of the proposed architectures

This experimental session yielded quite interesting results. For each dataset, the accuracy values always lied within a narrow interval. Secondly, ResidualNets seemed to suffer the switch from full precision to half precision more than other comparisons. This issue can be explained by considering that ResidualNets architectures were far deeper than the others, thus making error propagation non negligible. Third, Mobile networks could compete with ResidualNets and VggNets; moreover, their performances were not affected by half-precision representation. In summary, empirical evidence confirmed that DSC is a valuable strategy for the design of the CNN architecture in polarity detectors, under the hypothesis that a large dataset is available for fine tuning.

Transfer domain

The second experimental session considered a realistic case, in which a model was trained on a large dataset and was tested on a different sample. This configuration emulated the practical run-time implementation. In this session, ANP40, MANP40, and T4SA provided the training set, while "Twitter" formed the test set. Unfortunately, the latter dataset was rather unbalanced, as 581 images were labeled as 'positive' and 301 as 'negative'. Thus, any naïve classifier (always predicting 'positive') would obtain 65% accuracy.

To tackle this issue, true positive rate and true negative rate were proposed: for each class, true positive rate and true negative rate give the number of patterns correctly classified as belonging to the corresponding class over the total number of patterns of the class. The aforementioned naïve classifier would obtain true positive rate 1.0 and 0.0 true negative rate.

Table 2. Average accuracy on the test set for single domain experiments.

Architecture	ANP40	MVSO40	T4SA
MobileNet_v1	80.50 (0.25)	76.58 (0.00)	50.68 (0.07)
MobileNet_v2	82.75 (0.00)	77.10 (0.07)	49.83 (0.12)
MobileNet_v2(1.4)	80.50 (-0.25)	77.55 (0.16)	49.94 (-0.01)
Res_101	82.25 (1.75)	78.19 (3.25)	48.27 (1.98)
Res_152	81.00 (5.75)	78.94 (3.29)	48.61 (2.12)
Vgg_16	80.75 (0.00)	75.90 (0.00)	48.89 (-0.01)
Vgg_19	80.00 (0.00)	75.71 (0.06)	48.69 (0.04)

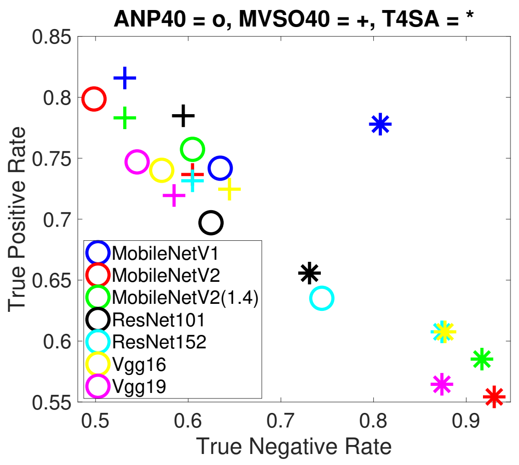

Figure 2. Transfer Domain results

Figure 2 reports on the results of this experimental session. Each mark is associated with the result of one experiment, involving a training set and a CNN architecture. In the graph, the x-axis gives the true negative rate, and the y-axis gives the true positive rate. Different markers identify the training datasets: circles refer to ANP40, crosses refer to MANP40, and stars refer to T4SA. Likewise, colors denote the various architectures (as per legend). The experiments were all performed by adopting the half-precision representation.

Empirical results confirmed that, unsurprisingly, the performances of the inference models strictly depended on the adopted training set. Both ANP40 and MANP40 led to high true positive rate values, and lower true negative rate values. T4SA instead lead to the opposite behavior. In term of architectures, MobileNet_v1 trained on T4SA scored the best performance; the polarity detector scored about 80% values for both the descriptors.

Implementation on Edge Devices

The final test session focused on the implementation of the polarity detector on low-power embedded devices. Experiments involved two commercial devices, namely Movidius Neural Computing Stick[2] and Nvidia Jetson TX2[3].

The Movidius Neural Computing Stick (MNCS) is an hardware accelerator designed to be plugged to a computing unit through a USB3.0 port. It embeds a VPU unit, namely Myriad 2, and 4 GB of LPDDR3 DRAM. MNCS implements the inference phase of a CNN after receiving the input data from the computing unit, which deals with data acquisition, pre-processing and operations scheduling. Inference operations are performed on half-precision floating point arithmetic.

Jetson TX2 is a stand alone device embedding a 256-core Pascal 1300MHz GPU, an ARM Cortex-A57 (quad-core) 2GHz, a NVIDIA Denver2 (dual-core) 2GHz and 8GB LPDDR4 RAM. In the experimental setup, a 32 SDD memory has been employed for operating system installation. During the experimental session a swap partition of 12 GB was adopted. The device was set on high performance through Nvidia configuration file.

In each experiment, the deployed model was obtained by fine tuning the involved CNN on ANP40. Table 3 summarizes the results of the experimental session: each row corresponds to an architecture (as per first column). The subsequent columns are divided in two groups: the second, third and fourth columns refer to Movidius and give the average latency in ms, the average memory consumption, and the average power consumption for completing an inference, respectively. The classification results were all estimated by testing on the “Twitter” dataset.

[2]http://movidius.com
[3]http://developer.nvidia.com/embedded/jetson-tx2

Table 3. Hardware analysis of the deployed models

	Movidius			Jetson		
Architecture	Latency (ms)	Memory (GB)	Power (Watt)	Latency (ms)	Memory (GB)	Power (Watt)
MobileNet_v1	43.98	0.18	1.17	12.0	0.93	1.65/5.76
MobileNet_v2	43.40	0.17	0.87	16.3	2.56	2.04/6.13
MobileNet_v2(1.4)	61.59	0.20	1.20	24.2	3.69	2.12/6.37
Res_101	405.49	0.93	1.42	22.2	4.22	1.56/7.55
Res_152	607.26	1.23	1.56	31.7	8.42	1.55/7.71
Vgg_16	864.84	2.68	-	43.7	11.60	2.41/7.94
Vgg_19	1046.84	2.79	-	58.0	11.59	2.38/7.59

Memory consumption was assessed via MNCS Python APIs, power consumption by using a USB power meter connected between Movidius and the host PC. The fifth, sixth, and seventh columns give the corresponding results for the test on Jetson TX2. In this case, memory consumption was measured by exploiting TegraStats. Accordingly, two different values are provided. The first value is the power consumption associated to the actual computing device (CPU, GPU, SOC, DDR); the second value is the power consumption associated to the whole development platform.

The table shows that, as expected, Jetson TX2 largely outperformed Movidius in terms of latency. Nonetheless, the experiments involving Movidius revealed the remarkable features of MobileNets, which were targeted indeed to resource-constrained devices. In terms of memory consumption, there was a significant gap between Movidius and Jetson. The latter platform, in fact, adds an overhead to the memory required to store the CNN's parameters. Such overhead stems from the optimization processes adopted by Jetson to reduce latency.

CONCLUSION

This paper has followed an empirical approach to show that image polarity detection can be deployed on edge devices, provided suitable design strategies are applied. The crucial aspect is the design of the pre-trained CNN that supports object classification. The hypothesis proposed in this work is that -when fine tuning can rely on a large dataset- a reliable polarity detector can be obtained even from a weak object classifier. Such a weak platform is in fact a CNN that is set to fit the constraints imposed by an edge device.

In practice, the research has proved that that hypothesis held true for the most prominent CNN architecture designed by exploiting DSCs, namely MobileNet. Polarity detectors based on MobileNet reached the same classification accuracy performances of detectors based on VggNet and ResNet. Nonetheless, the implementations on edge devices confirmed that MobileNet attained outstanding performances in terms of both latency and memory usage.

ACKNOWLEDGEMENTS

The authors acknowledge the support of NVIDIA Corporation with the donation of the Jetson TX2 used for this research.

REFERENCES

1. E. Cambria, H. Wang, and B. White, "Guest Editorial: Big Social Data Analysis," *Knowledge-Based Systems*, vol. 69, 2014, pp. 1–2.
2. S. Zhao et al., "Affective Image Content Analysis: A Comprehensive Survey." *IJCAI*, 2018, pp. 5534–5541.
3. I. Chaturvedi et al., "Fuzzy Commonsense Reasoning for Multimodal Sentiment Analysis," *Pattern Recognition Letters*, vol. 125, no. 264-270, 2019.
4. E. Ragusa et al., "A Survey on Deep Learning in Image Polarity Detection: Balancing Generalization Performances and Computational Costs," *Electronics*, vol. 8, no. 7, 2019, p. 783.
5. E. Cambria and A. Hussain, "Sentic Album: Content-, Concept-, and Context-Based Online Personal Photo Management System," *Cognitive Computation*, vol. 4, no. 4, 2012, pp. 477–496.
6. X. Liu, N. Li, and Y. Xia, "Affective image classification by jointly using interpretable art features and semantic annotations," *Journal of Visual Communication and Image Representation*, vol. 58, 2019, pp. 576–588.

7. S. Poria et al., “A review of affective computing: From unimodal analysis to multimodal fusion,” *Information Fusion*, vol. 37, 2017, pp. 98–125.
8. W. Wei-ning, Y. Ying-lin, and J. Sheng-ming, “Image retrieval by emotional semantics: A study of emotional space and feature extraction,” *Systems, Man and Cybernetics, 2006. SMC’06. IEEE International Conference on, vol. 4*, 2006, pp. 3534–3539.
9. J. Machajdik and A. Hanbury, “Affective image classification using features inspired by psychology and art theory,” *ACM International Conference on Multimedia*, 2010, pp. 83–92.
10. V. Yanulevskaya et al., “Emotional valence categorization using holistic image features,” *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, 2008, pp. 101–104.
11. D. Borth et al., “Large-scale visual sentiment ontology and detectors using adjective noun pairs,” *ACM International Conference on Multimedia*, 2013, pp. 223–232.
12. V. Campos et al., “Diving deep into sentiment: Understanding fine-tuned cnns for visual sentiment prediction,” *International Workshop on Affect & Sentiment in Multimedia*, 2015, pp. 57–62.
13. T. Chen et al., “Deepsentibank: Visual sentiment concept classification with deep convolutional neural networks,” *arXiv preprint arXiv:1410.8586*, 2014.
14. Q. You et al., “Robust Image Sentiment Analysis Using Progressively Trained and Domain Transferred Deep Networks.” *AAAI*, 2015, pp. 381–388.
15. B. Jou et al., “Visual affect around the world: A large-scale multilingual visual sentiment ontology,” *ACM International Conference on Multimedia*, 2015, pp. 159–168.
16. S. Fan et al., “The Role of Visual Attention in Sentiment Prediction,” *ACM International Conference on Multimedia*, 2017, pp. 217–225.
17. P. Balouchian and H. Foroosh, “Context-Sensitive Single-Modality Image Emotion Analysis: A Unified Architecture from Dataset Construction to CNN Classification,” *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 1932–1936.
18. H. Zheng et al., “When saliency meets sentiment: Understanding how image content invokes emotion and sentiment,” *Image Processing (ICIP), 2017 IEEE International Conference on*, 2017, pp. 630–634.
19. L. Wu et al., “Visual Sentiment Analysis by Combining Global and Local Information,” *Neural Processing Letters*, 2019, pp. 1–13.
20. Q. You, H. Jin, and J. Luo, “Visual Sentiment Analysis by Attending on Local Image Regions.” *AAAI*, 2017, pp. 231–237.
21. J. Yang et al., “Visual sentiment prediction based on automatic discovery of affective regions,” *IEEE Transactions on Multimedia*, 2018.
22. J. Yang et al., “Weakly Supervised Coupled Networks for Visual Sentiment Analysis,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7584–7592.
23. K. Song et al., “Boosting Image Sentiment Analysis with Visual Attention,” *Neurocomputing*, 2018.
24. T. Rao et al., “Multi-level region-based Convolutional Neural Network for image emotion classification,” *Neurocomputing*, vol. 333, 2019, pp. 429–439.
25. Y. Cheng et al., “A survey of model compression and acceleration for deep neural networks,” *arXiv preprint arXiv:1710.09282*, 2017.
26. M. Sandler et al., “Mobilenetv2: Inverted residuals and linear bottlenecks,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.
27. A. G. Howard et al., “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.
28. F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” *IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.
29. X. Zhang et al., “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.
30. L. Vadicamo et al., “Cross-Media Learning for Image Sentiment Analysis in the Wild.” *ICCV Workshops*, 2017, pp. 308–317.

Edoardo Ragusa is a postdoctoral researcher at University of Genoa. His research interests include machine learning in resource constrained devices, convolutional neural networks, and sentiment analysis. Contact him at edoardo.ragusa@edu.unige.it.

Christian Gianoglio is a postdoctoral researcher at University of Genoa. His research interests include embedded system, machine learning, and advanced signal processing. Contact him at christian.gianoglio@edu.unige.it.

Rodolfo Zunino is an associate professor at University of Genoa. His research interests include embedded systems, machine learning, massive-scale text-mining and text-clustering methods, and cybersecurity. Contact him at rodolfo.zunino@unige.it.

Paolo Gastaldo is an associate professor at University of Genoa. His research interests include machine learning, embedded computing systems, and artificial tactile sensing. Contact him at paolo.gastaldo@unige.it.