An hardware-aware image polarity detector enhanced with visual attention

Edoardo Ragusa, Tommaso Apicella, Christian Gianoglio, Rodolfo Zunino and Paolo Gastaldo
Department of Electrical, Electronic, Telecommunication Engineering and Naval Architecture DITEN
University of Genoa
Genova, Italy
Email: edoardo.ragusa@edu.unige.it, s4111548@studenti.unige.it, christian.gianoglio@edu.unige.it,
rodolfo.zunino@unige.it, paolo.gastaldo@unige.it

***Abstract*—Providing user customized experience is one of the main goals for present-day electronic smart devices. Image polarity detection plays a crucial role in understanding users' preferences due to the fact that information is massively represented by means of pictures. State-of-the-art frameworks are based on deep learning networks and continue evolving adding sophisticated structures to enhance generalization performances of the inference systems. Recent works proved that image analysis can be enhanced exploiting the information about salient regions. However, better performances are obtained at the cost of a higher computational load. This paper presents a hardware-friendly deep learning framework for image polarity detectors based on salient regions of an image. Experimental results show the reliable performances of the proposed solution on real-world data.**

***Index Terms*—image polarity detection, CNN, embedded systems**

I. Introduction

Deep learning has changed the role played by Artificial intelligence (AI) in data science. The availability of computers with ever increasing computational capability and distributed computing is enabling deep learning to address very complex problems.

In this regard, sentiment analysis is one of the most challenging benchmark for AI [1]. Sentiment analysis aims to identify the emotional information expressed in a content. Image polarity detection, in particular, deals with the emotional information conveyed by an image; such topic is very interesting in that images and videos play a major role in social networks. In fact, image polarity detection is emphasizing all the challenges brought about by sentiment analysis. Image polarity suffers from the so-called *subjective perception problem* [2], i.e., different users perceive the same image in different ways.

Modern polarity detectors rely on convolutional neural networks (CNNs) to deal with the inherent complexity of the problem at-hand, as they proved very effective in extracting suitable features from images [3]. CNNs, though, require lot of human effort in the architecture design; furthermore, they are computationally hungry. Indeed, recently, several researches showed that saliency detection plays a very important role in predicting the polarity of an image [4]–[6]. The underlying rationale is that the salient parts of an image are the most informative components for sentiment analysis. Retrieving automatically the salient components of an image, however, is a non trivial task. Again, the literature proved that CNNs should be adopted to implement reliable saliency detectors. This outcome in turn emphasizes the issue of computational complexity.

This paper focuses on the design of an image polarity detector that can be deployed on resource-constrained embedded systems. In this regard, the eventual target is represented by mobile devices, which are expected to host on-line polarity detectors. Actually, the goal of the present research is to show that image polarity detection enhanced with visual attention can be supported by hardware-friendly deep networks, which allow one to properly balance generalization performances and resource occupation.

A. Contribution

This work explores the use of hardware-friendly neural networks for the development of image polarity detectors based on salient regions of an image. The outcomes of the proposed work can be summarized as follows:

- a study about saliency detector based on hardware-friendly deep networks;
- a novel algorithm for image polarity detection based on hardware-friendly architectures.

II. Related works

Currently all the state-of-the-art implementations of image polarity detection rely on deep learning. In the last years several different approaches have been presented; a complete review about this topic can be found in [2], [3], [7].

In general, polarity detection frameworks exploit the low-level features extracted by a CNN trained on object recognition; fine-tuning is adopted to adjust network parameters for the specific task. The approaches existing in the literature mainly differ in: a) the adopted CNN, b) the transfer learning technique, and c) the training data domain. Most of the approaches fine-tuned pre-trained object classifiers. In this regard, Campos et al. [8] adopted the *AlexNet* architecture for feature extraction; the paper proposed an interesting analysis on the effects of layer ablation and layer addition on the eventual generalization performances of the fine-tuned network. Ontology-based representation have been widely explored on

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

top of object recognition models [9]. Interestingly, a few custom architectures were proposed: You et al. [10] proposed an architecture in which a pair of convolutional layers and four fully connected layers were stacked.

In the last few years the focus has moved to augmenting the basic solutions based on a fully convolutional architecture. 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. [11], 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 [12], context information [13], salience [4]–[6], attention mechanisms [14]–[16] and combinations of the last two [17]. Rao et al. [18] exploited a faster-RCNN to locate the distinctive parts of an image.

Finally, previous works proved that multimodal approaches can enhance the overall performances of sentiment classifiers [19]. However, these models critically rely on the performances of the feature extractors for different information sources and require additional computations for the fusion of the extracted features.

Little attention, though, has been paid to the deployment of image polarity classifiers on resources constrained devices. This topic has been recently addressed in [20], where the authors analyzed solutions employing hardware-friendly neural networks and weights truncation. The next section extends the the study proposed in [20] by including automatic saliency detection to the purpose of enhancing polarity detection performances.

III. Hardware-friendly deep networks

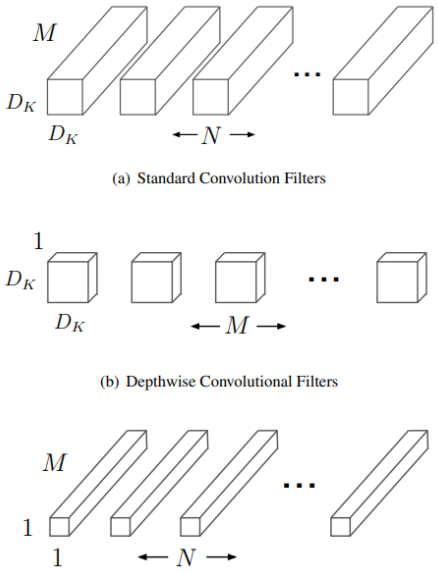
Deep networks embrace different architectures, which bring about several design issues when targeting hardware implementations. Actually, one needs to properly balance accuracy, memory consumption and computational cost.

Recently, Ragusa et. al [20] empirically proved that it is possible to implement polarity detection with hardware friendly CNNs without significantly affecting the eventual generalization performances. Such goal can be reached when a large dataset is available for fine-tuning. Notably, effective results were achieved with *MobileNetV1*, i.e., an architecture specifically designed for embedded systems.

Such outcome is interesting as state-of-the art saliency detectors derive from object detection meta-architectures [21], [22]. Indeed, according to [23], the framework combining MobilenetV1 and Single Shot Detector (SSD) represents the best option when one wants to implement object detection on resource-constrained devices. In fact, in [23] it has been empirically proved that such setup lies on the pareto-optimally surface in a latency versus accuracy analysis of more than 100 different configurations.

Thus, MobilenetV1 proved able to effectively support hardware deployment of both the core tasks investigated in this paper, i.e., image polarity detection and saliency detection (via object detection). The Depthwise Separable Convolution (DSC) is one of key features of MobilenetV1 [24]–[27]. 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 computing linear combinations of the input channels.

Figure 1 [25] schematizes the overall concept. Figure 1(a) sketches the representation of a standard kernel. Instead, Figure 1(b) shows the two components of the depth-wise kernel, i.e. the M kernels that convolute the input data and the N 1-D kernel used to merge the information retrieved from the single channel kernels.



(a) Standard Convolution Filters

(b) Depthwise Convolutional Filters

Fig. 1. Pictorial representation of depth-wise convolution [25]

This decomposition remarkably reduces computational costs: given an input of size $H \times W \times M$ and a convolutional layer characterized by N kernels of size $D_k \times D_k$, the computational cost C_{sc} of standard convolution is

$$C_{sc} = H \times W \times M \times N \times D_k \times D_k. \quad (1)$$

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

$$C_{DSC} = H \times W \times M \times (D_k^2 + N). \quad (2)$$

which is significantly smaller than (1).

Many CNN architectures exploiting DSC can yield on object detection the same performances of state-of-the-art architectures in terms of accuracy [24]–[28]. In the meanwhile, DSC reduces memory consumption. Standard deep learning algorithms need devices with GBs of dedicated memory, as an example Movidius Neural Computing Stick offers 4GBs of memory dedicated to model parameters, meanwhile, Jetson TX2 hosts 8 GBs of Ram [20]. Conversely, most of the commercial devices such as smartphones or smart devices dedicate

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

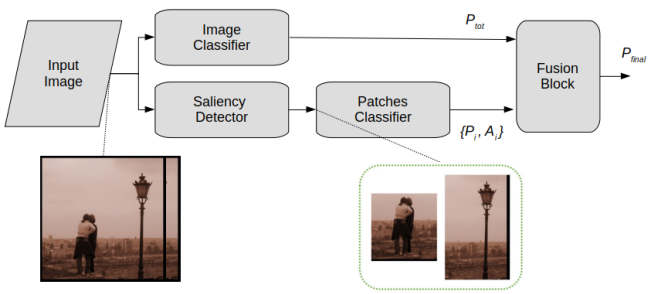


Fig. 2. Complete schema showing the classification flow for the whole system.

only few MBs for model parameters. As an example, microcontroller Cortex M7, largely employed in IoT applications, devotes only 2 MBs for parameters' storing. In these cases, DSC based solutions are the only option to cope with memory constraints.

IV. Proposal

The present paper approaches the problem of enhancing image polarity detection by splitting the overall problem into sub-tasks, thus exploiting a bottom-up approach. Such setup follows a common setup adopted in the literature [6]. However, in this work, each module of the proposed architecture has been designed by taking into account hard constraint in terms of memory consumption, power consumption and number of floating point operations.

Figure 2 summarizes the overall processing flow. The system is organized into four logical blocks and two branches. The upper branch coincides with the standard classification scheme exploited in polarity detection literature: the input image feeds an image classifier that outputs the polarity (P_{tot}) computed on the overall image. The lower branch firstly elaborates the input images using the Saliency Detector block. This module produces a set of patches, extracted from the original image, containing the salient part of the pictures. The figure shows the output of this block by using a sample image as input; in this example, the patches of the two salient objects are generated. The patches feed the Patches Classifier, which assigns a polarity to each patch. Finally, a Fusion Block analyzes the output of the Image Classifier and the Patches Classifier; this block is entitled to produce the label to be assigned to the input image.

A detailed description of the four blocks is given in the following. It is assumed that the output of each polarity classifier is a scalar $\in [-1, 1]$, where -1 means totally negative and $+1$ means totally positive.

A. Image Classifier

The Image Classifier approaches the image polarity classification problem by using a single CNN; it receives as input the whole image without additional information. Thus, a pre-trained CNN architecture provides the starting point. A fine tuning procedure is applied to address image polarity detection.

This work adopted a MobileNetV1 as CNN. The architecture is characterized by 4.2M parameters; an inference phase requires 569M multiplication and addition operations. The most commonly adopted architecture for image polarity, VGG_16, involves 138M parameters; it requires 15300M multiplication and addition operations for completing an inference [25]. In other words, MobilenetV1 is about 32 times smaller in terms of memory consumption and 27 times more efficient in terms of floating point operations. Nonetheless, MobileNetV1 has proved able to reach almost the same accuracy of VGG_16 in polarity detection [20].

B. Saliency Detector

Saliency detection is a sub-task of computer vision that aims to individuate automatically the salient part of an image, i.e., the parts that are more interesting for a human. Recent works [6] approached this specific task adopting some strategies from object detection [23]. In principle, the inference strategy employed by deep networks for object detection can be divided into two logical parts: 1) a meta-architecture that produces box proposal, and 2) the backbone CNN that performs feature extraction on the boxes. Examples of meta-architectures are external proposal generators, like R-CNN and Fast R-CNN, or part of an architecture, like SSD or YOLO. Examples of CNNs usually exploited for feature extraction are VGG16 and ResNet50. Again, the starting point is a pre-trained network targeted on object detection. A fine-tuning procedure is adopted to shift the domain to saliency detection.

In the present work, the Saliency Detector is handled with the SSD-MobileNetV1 architecture. SSD is a meta-architecture that merges box proposal and classification steps, thus reducing consistently the overall computational cost. In fact, this model uses 1.2 billions multiplication and addition operations and stores 6.8 M parameters. The equivalent process using Fast R-CNN as meta-architecture and VGG_16 as feature extractor involves 64.3 billions operations and 138.5M parameters.

C. Patches Classifier

The Patches Classifier is entitled to assign a polarity to each patch forwarded by the Saliency Detector. Basically, it has the same structure of the Image Classifier. However, the fine-tuning procedure is performed over patches extracted from image polarity datasets. The details of this procedure will be discussed in Section V-B.

D. Fusion Block

The Fusion Block processes the outcomes of the Image Classifier and the Patches Classifier to assign a polarity to the input image. In the proposed system, this block adopts an hard coded algorithm; hence, it is not subject to training.

The procedure is summarized in the following points:

- Salient patches are divided in 5 cluster $C_j, j = 1, ..., 5$ on the basis of the ratio between the area of the i$-th$ patch A_i and the area of the overall image A_{tot}. The five ranges are: $(0.0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1)$. In

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

practice, small patches are in the set C_1 with range $0.0 < A_i/A_{tot} < 0.2$ and big patches are in the set C_5 with range $0.8 \leq A_i/A_{tot} < 1$.

- the polarity of each cluster is obtained adding together the polarity of all the patches inside a cluster: $P_{C_i} = \sum_{j \in C_i} P_j$. For example, if a cluster contains only two patches having polarity -0.4 and $+0.1$, respectively, the overall polarity of the cluster is -0.3.
- The final prediction is computed considering the element with maximum absolute value $h = [P_{C_1}, P_{C_2}, P_{C_3}, P_{C_4}, P_{C_5}, P_{tot}]$. Thus, $\arg\max_j(|h_j|)$ is the magnitude of the eventual prediction. Then, one assigns positive or negative polarity according to the sign of h_j. For example if an images has $P_{C_1} = -0.1$, $P_{C_2} = -0.7$, $P_{C_3} = -0.3$, $P_{C_4} = +0.5$, $P_{C_5} = -0.2$ and $P_{tot} = 0.3$, h_j is -0.7 thus the eventual polarity is negative.

V. Experimental Results

Experiments aimed at evaluating the accuracy of the proposed method. The experimental setup has been divided in three sessions. The first session focused on the performance of the Saliency Detector. The second session assessed the performance of the Image Classifier and the Patches Classifier. Finally, the third session evaluated the performance of the whole framework.

A. Saliency Detector

The training of the Saliency Detector has been implemented by following the setup proposed in [6]. The training involved a SSD_MobileNet_COCO model[1], i.e., the SSD_MobileNet architecture trained for the object detection task using COCO dataset [29]. All the training parameters have been inherited from the configuration file of the original model. Only the number of iterations has been changed to 50,000.

Two datasets for saliency detection have been utilized in this experimental campaign. Both of them are collections of images paired with information about salient contents. The fine-tuning procedure has been divided in two phases. First, the original model has been fine-tuned by using the ILSVRC-2014 dataset, which contains 127030 images. Then, a second fine-tuning process has been completed by using the SOS dataset [2], which contains 3951 images. Standard hold-out procedure has been applied to evaluate generalization performances: 90% of the data made the training set while the remaining 10% of the data were included in the test set. Accordingly, the test set was never employed for tuning any parameter or hyper-parameter.

Table I summarizes the results of the experimental session. The first column lists the two models resulting from the two fine-tuning processes. SSD_MobileNet_ILSVRC is the model trained by using only the ILSVRC dataset; SSD_MobileNet_SOS is the model that underwent a second fine-tuned process by adopting the SOS dataset. The second column gives the number of true boxes contained in the test set. The third column gives the number of boxes proposed by the fine-tuned models. Finally, the last column provides to IoU75 measure. The IoU75 mAP indicator [29] usually assesses the performance of object detection models. The IoU takes into account the set A of pixels associated to a box proposed by the detector and the ground-truth set of pixels B. It can be formalized as:

$$IoU(A, B) = \frac{A \cap B}{A \cup B} \tag{3}$$

IoU75 means that a prediction is classified as correct if $IoU(A, B) > 0.75$. Informally speaking, this quantity considers as correct only predictions that are sufficiently overlapped with respect to the real position of the salient region.

TABLE I
Saliency detector performances

Saliency detector	True boxes	Predicted	IoU $> 75\%$
SSD_MobileNet_ILSVRC	13982	13118	10207
SSD_MobileNet_SOS	13982	14723	9272

The outcomes of the experimental campaign confirm that the number of predicted boxes for the test set is close to the ground-truth. Moreover, the IoU measure asserts that the predicted boxes are sufficiently overlapped with the real positions. IoU is greater than 75% in the 77.8% of the cases for SSD_MobileNet_ILSVRC and in the 63.0% of cases for SSD_MobileNet_SOS. Such results are very interesting when considering that an hardware-friendly architecture has been utilized.

B. Image Classifier and Patches Classifier

The second session aimed at characterizing the performance of two classification modules involved in the proposed framework, namely Image Classifier and Patches Classifier. Both these modules are based on the MobileNetV1 architecture.

The CNN training for image polarity detection requires a large size dataset. In this work a pruned version of the adjective-noun pairs (ANP) dataset [30] has been utilized as training set. The original dataset holds more than 1 million images crawled from Flickr by an automatic system. This paper relied on ANP40 [3], which focuses only on the 20 most positive and the 20 most negative adjective noun-pairs. The corresponding images have been split in the corresponding subsets (positive and negative samples). Eventually ANP40 contains 10,516 images.

The Image Classifier and the Patches Classifier share the same architecture: the MobileNetV1 model trained on the ILSVRC-2012-CLS dataset[2]. Indeed, the two classifiers have been fine-tuned separately. Hence, the Image Classifier has been trained on the entire ANP40 dataset. The Patches Classifier has been trained only on the patches extracted from the ANP40 dataset. In the following, PC_ILSVRC will refer to the Patches Classifier fine tuned on the patches extracted

[1] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md

[2] https://github.com/tensorflow/models/tree/master/research/slim

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

via SSD_MobileNet_ILSVRC, while PC_SOS will refer to the Patches Classifier fine tuned on the patches extracted via SSD_MobileNet_SOS (as per Sec. V-A).

The classifier have been evaluated on two well known datasets: Twitter (Tw) [10] and Multi-view (Mv_eq and Mv_maj) [31]. Notably, these datasets have been utilized only as test sets. Neither parameters tuning nor model selection were performed by using these data. The two datasets have been selected because the images were manually annotated from a pool of human users. For Tw the all-agreed version has been considered. Two versions of MVSA have been considered: MVSA_eq, which is the all-agreed version of MVSA; MVSA_maj, where the eventual label is assigned by majority voting of the human annotators.

The performances of the classifiers -given a test set- have been assessed by applying the following procedure. First, for each image in the test set the patches have been extracted. Second, the patches have been split into five categories according to their relative size with respect to the source image (as per Sec. IV-D). Third, for each category two groups have been generated: the group of patches belonging to that category and the group of images from which that patches were extracted.

Table II reports the outcomes of the tests involving the Tw dataset; here, the Patches Classifier was supported by PC_ILSVRC. The table compares the performance of the Image Classifier and the Patches Classifier on the five categories separately, based on the area covered by the patches: C_1 contains patches that cover from 0% to 20% of the area of the entire image meanwhile C_5 includes patches that cover from 80% to 100% of the original image. Accordingly, the second column indicates the input provided to the classifier, given a category: the set of patches or the set of images. Columns from 3 to 7 refer, respectively, to the five categories. Thus, given a classifier and an input set, column 3 gives the average accuracy -expressed in the range [0, 1]- over the patches/images belonging to the category C_1, i.e., the category that refers to the patches that cover less than the 20% of the source image. The same format applies to the remaining columns.

TABLE II
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET TWITTER WITH PATCH EXTRACTOR TRAINED ON ILSVRC

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.628	0.718	0.516	0.768	0.740
	Image	0.718	0.732	0.645	0.750	0.719
PC_ILSVRC	Patches	0.679	0.761	0.645	0.696	0.719
	Images	0.756	0.723	0.726	0.732	0.688

Results enlighten few interesting outcomes. When the input are patches, PC_ILSVRC outperforms Image Classifier, considering the small and medium patches size (i.e., C_1, C_2 and C_3 clusters). The same trend holds for C_1 and C_3 when the entire images are considered. On the contrary, Image Classifier has a better accuracy than PC_ILSVRC for both patches and full images for C_4 and C_5. This is not surprising because bigger patches are similar to the full images.

Table III reports the result for Tw dataset, when patches are extracted from SD_SOS.

TABLE III
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET TWITTER WITH PATCH EXTRACTOR TRAINED ON SOS

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.584	0.748	0.748	0.685	0.761
	Image	0.734	0.67	0.696	0.71	0.746
PC_SOS	Patches	0.659	0.709	0.738	0.71	0.72
	Images	0.676	0.688	0.734	0.71	0.756

With the patches as input, PC_SOS has a better accuracy in C_1 and C_4

Consistently with Table II, for the smallest patches (i.e., C_1) the Patch Classifier is the best option. However in all the other cases, except for cluster C_4, the standard Image Classifier seems to be more convenient.

Table IV reports the result for dataset MVSA_eq when PC_ILSVRC is employed.

TABLE IV
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET MVSA_EQ WITH PATCH DETECTOR TRAINED ON ILSVRC

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.519	0.578	0.665	0.649	0.631
	Image	0.722	0.689	0.636	0.649	0.614
PC_ILSVRC	Patches	0.839	0.852	0.799	0.814	0.817
	Images	0.829	0.852	0.804	0.818	0.805

Experimental results are significantly different with respect to the ones obtained for Twitter dataset. The major outcome of this experiments is that the classifier trained on patches is always significantly better, independently from the input provided (images or patches) and from the size of the patches. This behaviour is probably due to the fact that this dataset is made of many heterogeneous images, i.e., one image could contain text, schemes, draws, etc. As a consequence, the patches classifier, trained on saliency region, only proves more suitable to cope with this kind of data.

Table V completes the analysis of MVSA_eq using PC_SOS.

TABLE V
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET MVSA_EQ WITH PATCH EXTRACTOR TRAINED ON SOS

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.624	0.635	0.669	0.656	0.605
	Image	0.701	0.71	0.689	0.625	0.611
PC_SOS	Patches	0.903	0.863	0.834	0.831	0.744
	Images	0.891	0.859	0.836	0.82	0.748

The experimental outcome confirms the trend shown with PC_ILSVRC.

Finally, Tables VI and VII complete the analysis.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

TABLE VI
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET MVSA_MAJ WITH PATCH DETECTOR TRAINED ON ILSVRC

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.524	0.542	0.627	0.62	0.621
	Image	0.691	0.627	0.614	0.628	0.599
PC_ILSVRC	Patches	0.811	0.791	0.785	0.774	0.778
	Images	0.812	0.803	0.781	0.785	0.786

TABLE VII
COMPARISON OF CLASSIFICATION ACCURACY OF IMAGE CLASSIFIER AND PATCHES CLASSIFIER FOR DATASET MVSA_MAJ WITH PATCH DETECTOR TRAINED ON SOS

Classifier	Input	C_1	C_2	C_3	C_4	C_5
Image Classifier	Patches	0.605	0.631	0.617	0.633	0.562
	Image	0.68	0.674	0.633	0.612	0.564
PC_SOS	Patches	0.871	0.828	0.784	0.776	0.69
	Images	0.86	0.811	0.782	0.779	0.69

The results, except for small fluctuations, confirm the outcomes of previous experiments.

The overall results of the experimental section confirm that the Patches Classifier and the Image Classifier have similar accuracy performances. The differences on trends depend on the tested dataset. In the following section, the experimental result proves that the combination of the two algorithm enhances the overall performances in almost all cases.

C. The overall framework

The last experimental session evaluated the performances of the overall framework (Figure 2). In the framework, the Saliency Detector, the Image Classifier and the Patches Classifier were trained as described above. The Tw, MVSA_eq and MVSA_maj datasets provided the test sets. Again, it is worth stressing that these datasets were never utilized for the model selection and the parameters tuning.

MobileNetV1 sets the baseline for the comparison with the proposed framework. Thus, the comparison is with an image polarity detector that does not involve any saliency detection. This model has been selected because 1) it is the most efficient solution in term of computational cost, and 2) it achieved the best classification accuracy among a set of architectures on a very similar experimental setup presented in [20].

Table VIII presents the result for the Twitter dataset. The first column gives the predictor: MobileNetV1, the proposed framework with a Patches Classifier supported by PC_ILSVRC, and the proposed framework with a Patches Classifier supported by PC_SOS. Columns from 2 to 5 report the performance on the test set measured by common metrics: accuracy, precision, recall, F1. Experimental results prove that both the implementations of the proposed framework were able to outperform the baseline MobileNetV1 in terms of accuracy, recall, and F1. Conversely, the baseline MobileNetV1 achieved a better precision.

Tables IX and X show the results for datasets MVSA_maj and MVSA_eq, respectively. The tables adopt the same format of Table VIII. Interestingly, in these two experiments the proposed implementations always outperformed the baseline MobileNetV1.

TABLE VIII
PERFORMANCES OF THE PROPOSED MODEL FOR DATASET TWITTER

Network	Accuracy	Precision	Recall	F1
MobileNetV1	70.86	80.00	74.35	77.07
Proposal$_{SOS}$	71.77	76.35	82.79	79.44
Proposal$_{ILSVRC}$	71.99	78.99	78.31	78.65

TABLE IX
PERFORMANCES OF THE PROPOSED MODEL FOR DATASET MVSA_MAJ

Network	Accuracy	Precision	Recall	F1
MobileNetV1	0.60	0.91	0.61	0.73
Proposal$_{SOS}$	0.69	0.90	0.72	0.80
Proposal$_{ILSVRC}$	0.63	0.90	0.64	0.75

TABLE X
PERFORMANCES OF THE PROPOSED MODEL FOR DATASET MVSA_EQ

Network	Accuracy	Precision	Recall	F1
MobileNetV1	0.64	0.95	0.64	0.76
Proposal$_{SOS}$	0.74	0.95	0.76	0.84
Proposal$_{ILSVRC}$	0.67	0.95	0.67	0.79

VI. CONCLUSIONS

The present paper explored the use of deep neural networks based on depth-wise separable convolutions as key elements for image polarity detection with image saliency information. The overall problem was addressed using a bottom-up approach. The eventual system uses three deep learning stages based on MobileNetV1. The overall setup remains computationally light compared to the standard solution presented in the field. Experimental result confirmed that the proposed method effectively balances generalization performances and overall compute cost.

REFERENCES

[1] E. Cambria, S. Poria, A. Gelbukh, and M. Thelwall, "Sentiment analysis is a big suitcase," *IEEE Intelligent Systems*, vol. 32, no. 6, pp. 74–80, 2017.

[2] S. Zhao, G. Ding, Q. Huang, T.-S. Chua, B. W. Schuller, and K. Keutzer, "Affective image content analysis: A comprehensive survey." in *IJCAI*, 2018, pp. 5534–5541.

[3] E. Ragusa, E. Cambria, R. Zunino, and P. Gastaldo, "A survey on deep learning in image polarity detection: Balancing generalization performances and computational costs," *Electronics*, vol. 8, no. 7, p. 783, 2019.

[4] S. Fan, M. Jiang, Z. Shen, B. L. Koenig, M. S. Kankanhalli, and Q. Zhao, "The role of visual attention in sentiment prediction," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 217–225.

[5] H. Zheng, T. Chen, Q. You, and J. Luo, "When saliency meets sentiment: Understanding how image content invokes emotion and sentiment," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 630–634.

[6] L. Wu, M. Qi, M. Jian, and H. Zhang, "Visual sentiment analysis by combining global and local information," *Neural Processing Letters*, pp. 1–13, 2019.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

[7] S. Poria, E. Cambria, R. Bajpai, and A. Hussain, "A review of affective computing: From unimodal analysis to multimodal fusion," *Information Fusion*, vol. 37, pp. 98–125, 2017.

[8] V. Campos, A. Salvador, X. Giro-i Nieto, and B. Jou, "Diving deep into sentiment: Understanding fine-tuned cnns for visual sentiment prediction," in *Proceedings of the 1st International Workshop on Affect & Sentiment in Multimedia*. ACM, 2015, pp. 57–62.

[9] T. Chen, D. Borth, T. Darrell, and S.-F. Chang, "Deepsentibank: Visual sentiment concept classification with deep convolutional neural networks," *arXiv preprint arXiv:1410.8586*, 2014.

[10] Q. You, J. Luo, H. Jin, and J. Yang, "Robust image sentiment analysis using progressively trained and domain transferred deep networks." in *AAAI*, 2015, pp. 381–388.

[11] S. Fan, M. Jiang, Z. Shen, B. L. Koenig, M. S. Kankanhalli, and Q. Zhao, "The role of visual attention in sentiment prediction," in *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 2017, pp. 217–225.

[12] 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, pp. 576–588, 2019.

[13] P. Balouchian and H. Foroosh, "Context-sensitive single-modality image emotion analysis: A unified architecture from dataset construction to cnn classification," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1932–1936.

[14] Q. You, H. Jin, and J. Luo, "Visual sentiment analysis by attending on local image regions." in *AAAI*, 2017, pp. 231–237.

[15] J. Yang, D. She, M. Sun, M.-M. Cheng, P. Rosin, and L. Wang, "Visual sentiment prediction based on automatic discovery of affective regions," *IEEE Transactions on Multimedia*, 2018.

[16] J. Yang, D. She, Y.-K. Lai, P. L. Rosin, and M.-H. Yang, "Weakly supervised coupled networks for visual sentiment analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7584–7592.

[17] K. Song, T. Yao, Q. Ling, and T. Mei, "Boosting image sentiment analysis with visual attention," *Neurocomputing*, 2018.

[18] T. Rao, X. Li, H. Zhang, and M. Xu, "Multi-level region-based convolutional neural network for image emotion classification," *Neurocomputing*, vol. 333, pp. 429–439, 2019.

[19] S. Poria, I. Chaturvedi, E. Cambria, and A. Hussain, "Convolutional mkl based multimodal emotion recognition and sentiment analysis," in *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE, 2016, pp. 439–448.

[20] E. Ragusa, C. Gianoglio, R. Zunino, and P. Gastaldo, "Image polarity detection on resource-constrained devices," *IEEE Intelligent Systems*, vol. 35, no. 6, pp. –, 2020.

[21] J. Zhang, S. Ma, M. Sameki, S. Sclaroff, M. Betke, Z. Lin, X. Shen, B. Price, and R. Mech, "Salient object subitizing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4045–4054.

[22] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech, "Unconstrained salient object detection via proposal subset optimization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5733–5742.

[23] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7310–7311.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[26] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[27] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.

[28] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1314–1324.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[30] D. Borth, R. Ji, T. Chen, T. Breuel, and S.-F. Chang, "Large-scale visual sentiment ontology and detectors using adjective noun pairs," in *Proceedings of the 21st ACM international conference on Multimedia*, 2013, pp. 223–232.

[31] T. Niu, S. Zhu, L. Pang, and A. El Saddik, "Sentiment analysis on multiview social data," in *International Conference on Multimedia Modeling*. Springer, 2016, pp. 15–27.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE