Viraliency: Pooling Local Virality

Xavier Alameda-Pineda, Andrea Pilzer, Dan Xu, Nicu Sebe, Elisa Ricci

Viraliency: Pooling Local Virality

Xavier Alameda-Pineda[1,2], Andrea Pilzer[1], Dan Xu[1], Nicu Sebe[1], Elisa Ricci[3,4]
[1] University of Trento, [2] Perception Team, INRIA [3] University of Perugia, [4] Fondazione Bruno Kessler
xavier.alameda-pineda@inria.fr, {andrea.pilzer,dan.xu,niculae.sebe}@unitn.it, eliricci@fbk.eu

Abstract

In our overly-connected world, the automatic recognition of virality – the quality of an image or video to be rapidly and widely spread in social networks – is of crucial importance, and has recently awaken the interest of the computer vision community. Concurrently, recent progress in deep learning architectures showed that global pooling strategies allow the extraction of activation maps, which highlight the parts of the image most likely to contain instances of a certain class. We extend this concept by introducing a pooling layer that learns the size of the support area to be averaged: the learned top-N average (LENA) pooling. We hypothesize that the latent concepts (feature maps) describing virality may require such a rich pooling strategy. We assess the effectiveness of the LENA layer by appending it on top of a convolutional siamese architecture and evaluate its performance on the task of predicting and localizing virality. We report experiments on two publicly available datasets annotated for virality and show that our method outperforms state-of-the-art approaches.

1. Introduction

Beyond the automatic understanding of objective properties of images, such as the presence of an object and its position in the scene, the computer vision community also invested efforts in analyzing subjective attributes of visual data. Memorability [4, 15], popularity [18], virality [7] and emotional content [1, 21] are few examples of such attributes. Further analysis was conducted to understand which parts of the image were responsible for the recognition of such properties. For instance, Doersch *et al*. identified specific mid-level visual patterns when recognizing city-based architectural styles [9]. De Nadai *et al*. studied the perception of safety in urban scenes [6], detecting which areas in an image are responsible for this perception. Naturally, many researchers also wondered how to transform an image so as to enhance or diminish its subjective properties, or even how to generate images with such properties. In this regard, Koshla *et al*. [17] investigated how to transform a face image so as to make it more memorable. Given a natural image, Gatys *et al*. [12] showed how to generate a stylized image from a natural image and an artistic painting using deep neural architectures.

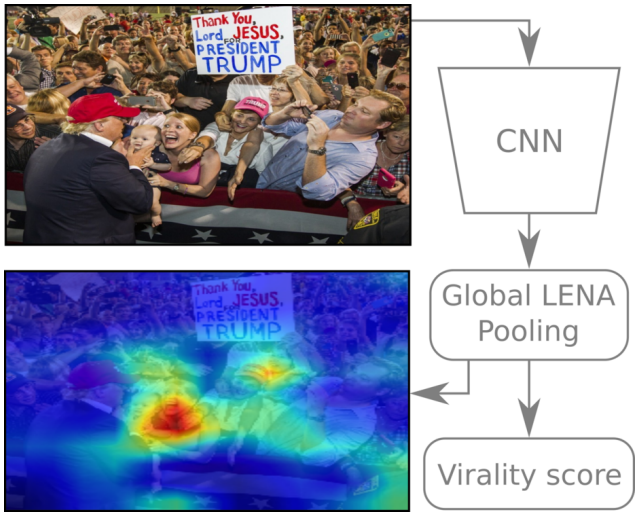

Figure 1. An image related to the U.S. presidential elections that went viral in different social networks, and its estimated viraliency map. The proposed pipeline: (i) A convolutional deep architecture is trained to generate virality-sensitive feature maps of the image; (ii) These features are passed through a LENA global pooling layer; (iii) The global pooling provides activations to estimate the virality score as well as a rough localization of the virally salient parts of the image, hence the title *viraliency*.

The particular case of virality – the quality of an image or video to be rapidly and widely spread on social networks– is of crucial importance in our overly-connected world, and it is the focus of this study. We hypothesize that, within an image, there are few different virally salient regions, *i.e*. areas responsible for making the image viral. Inspired from previous research studies [27, 22], we introduce a novel global pooling layer, the learned top-N average (LENA) pooling layer, specifically designed to automatically detect the visual patterns correlated with virality, *i.e*. the *viraliency map* (Figure 1). We further show that, by embedding our LENA pooling into a convolutional deep architecture, we can successfully predict the virality score of an image and simultaneously uncover its viraliency map. We test the LENA layer within different network architectures and perform an extensive experimental evaluation on two recent and publicly available datasets used for visual virality analysis [14, 7], demonstrating that our method outperforms state-of-the-art approaches on virality prediction and localization.

Up to the authors' knowledge, this is the first study addressing the complex task of recognizing image virality with an end-to-end trainable deep network. The proposed architecture is specifically designed to simultaneously predict image virality and to automatically identify the parts of the image responsible for it (without using any information on virality localization). Secondly, we introduce the LENA pooling layer and demonstrate its effectiveness in virality prediction and in enhancing the identification of virally salient zones (viraliency maps) in two publicly available datasets. Interestingly, we also show that including objectness maps derived from pretrained deep models is advantageous for the task of interest. Finally, we complement the existing datasets with virality localization annotations and provide visualization results for the intuitive understanding of the advantage of the LENA pooling layer.

2. Related work

Our work is closely related to two recent trends in the computer vision community: (i) the understanding and recognition of subjective properties of visual data and (ii) the use of global pooling layers in deep neural networks for weakly-supervised localization.

Understanding and recognizing subjective properties of images is challenging because, unless some related information can be automatically extracted from auxiliary data sources (*i.e*. metadata), collecting and annotating datasets is a tremendous effort. Indeed, given that the perception of subjective properties inherently depends on the perceiver in a strong manner, each image requires to be annotated by a relatively large number of people. Such strategy could be an option if a web-based platform already exists and it provides annotations, as for instance for aesthetics [8]. Usually, it is easier to give relative annotations between a pair of images than absolute scores. This scheme has been successfully employed in the past for urban perception [10] and emotion recognition from abstract paintings [23], but typically requires some post-processing to handle noisy annotations.

This problem is aggravated by the data-hungry deep neural architectures. It is therefore unsurprising that the computer vision community payed special attention to those subjective properties for which semi-automatic annotation schemes can be devised. Memorability [16, 4, 17, 15, 24] is the example *par excellence*, since the memory game sets a very user-friendly and enjoyable environment for memorability annotation. Popularity and virality fall also into this category, thanks to the computational proxies provided by social networks. In particular, statistics of *upvotes*, *likes*, *shares* and *resubmissions* can provide almost-clean labeled datasets. The difference between virality and popularity is that viral images have been upvoted/liked and have been shared/resubmitted several times, while popular images do not satisfy the latter, as reported in [7]. Khosla *et al*. [18] analyzed the popularity of images in Flickr. The study from Deza and Parikh [7] was the first attempt to understand virality from visual content by focusing on the mid-level attributes of images. In this paper we explore an orthogonal research direction to [7] and propose a deep architecture including a novel pooling layer specifically designed to understand which parts of an image contribute to virality. To our knowledge, this is the first work focusing on this aspect.

Our proposal is inspired from recent research on deep networks analyzing the role of global pooling layers for weakly-supervised object localization [20, 27]. As for the case of subjective properties, collecting datasets with annotations of the objects' bounding box is very tedious. Therefore, researchers in computer vision found alternative ways to tackle the problem of detection using only image-level annotations, *i.e*. simply indicating the object presence/absence in the image [5, 3]. A recent line of research explored weakly-supervised object localization through the use of global pooling layers on convolutional neural networks. For instance, Oquab *et al*. [20] analyzed the ability of global max pooling to predict locations of objects inside a deep network trained for object classification. Similarly, Zhou *et al*. [27] addressed weakly-supervised object localization using global average pooling and extended their analysis to abstract concepts. Porzi *et al*. [22] introduced the top-N average pooling to study subjective judgments from urban scenes and automatically extract image regions responsible for these judgments. In this paper we follow this research direction and analyze if global pooling layers are also effective when used for classifying and localizing patterns associated to virality. In addition, we step beyond previous research studies by introducing the *learned* top-N average pooling, able to learn the size of the support area to be averaged. LENA is designed to find the best compromise between average and max pooling, and it is described in the next section, together with the proposed architecture.

3. Viraliency through global LENA pooling

3.1. Predicting and localizing virality

We use an end-to-end trainable siamese deep neural network consisting on three main blocks: a fully convolutional front-end, a global LENA pooling layer and a final inner-product layer used to predict the virality score. These three blocks can be observed in the scheme of Figure 2. We remark that the network is fully siamese: all the parameters of the convolutional, global pooling and inner product layers are shared between the two branches. Importantly, the front-end (base architecture) can be arbitrarily chosen as long as it is fully convolutional. In the experimental section we show results with three different base architectures.

We chose to use a siamese network in order to be as close as possible to the philosophy of previous studies on visual-based virality prediction [7]. More formally, we as-

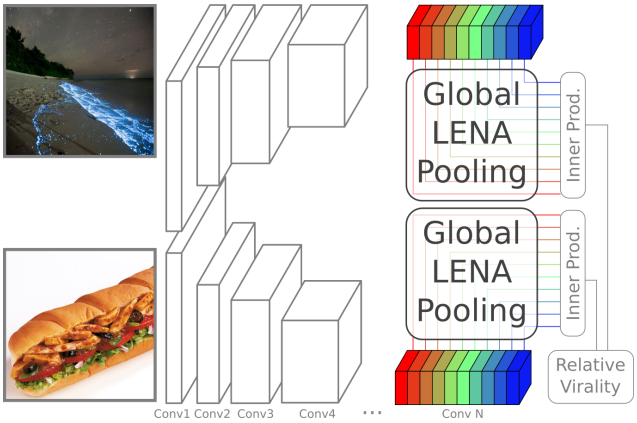

Figure 2. The proposed end-to-end trainable siamese architecture consisting on: (i) a fully convolutional front-end, (ii) a global learned top-N average (LENA) pooling (used on top of the convolutional structure to extract the activation from each feature map) and (iii) an inner product layer to estimate the relative virality between two images from the extracted activations.

sume the existence of a training set $\mathcal{T}$ consisting of M pairs of images and the annotated relative virality: $\mathcal{T} = \{(I_m, J_m, v_m)\}_{m=1}^{M}$, where $v_m = 1$ if I_m is more viral than J_m and $v_m = -1$ otherwise. In order to train the parameters of the siamese network, we optimize the sigmoid cross-entropy loss over the training set using stochastic gradient descent:

$$\mathcal{L} = \sum_{m=1}^{M} -v_m \log \hat{v}_m - (1 - v_m) \log(1 - \hat{v}_m)), \quad (1)$$

where $\hat{v}_m = s(I_m; \theta) - s(J_m; \theta)$ is the subtraction of the output of the two branches of the siamese network, and θ denotes the set of shared parameters.

By designing the network as in Figure 2, the convolutional front-end will extract a set of feature maps, also known as latent concept detectors [27, 22]. While in previous studies these concepts were associated to the presence/absence of objects in the image or to the safety of urban scenes, in our case the latent detectors will be associated to virality. In this paper, we adopt global pooling so as to exploit these latent detectors for virality prediction and weakly-supervised virality localization.

3.2. Global pooling

We assume the existence of L latent concept detectors and attempt to learn their relationship with virality. Each of these concept detectors is one output channel of size $W \times H$ of the convolutional front-end, $f_l \in \mathbb{R}^{W \times H}$ (each of the colored slices of `Conv N` in Figure 2). Generally speaking, global pooling extracts activations from each latent detector and feeds them to a fully connected layer responsible for classification. We remark the existence of three global pooling strategies in the literature.

Global average pooling In [27], the features maps are channel-wise averaged and fed to the fully connected layer. The classification score for each class $k \in \{1, \ldots, K\}$ (before the soft-max operation) is given by:

$$q_k^{\text{GAP}} = \sum_{l=1}^{L} w_{kl}^{\text{GAP}} \frac{1}{WH} \sum_{w,h=1}^{W,H} f_l(w, h), \quad (2)$$

were w_{kl}^{GAP} are the weights of the classification layer. This strategy is referred to as global average pooling (GAP) since all the pixels of channel l are averaged before being fed to the fully connected layer. One prominent advantage of global pooling is that we can easily construct a *class activation map* for each class k, which in case of GAP writes:

$$a_k^{\text{GAP}}(w, h) = \sum_{l=1}^{L} w_{kl}^{\text{GAP}} f_l(w, h). \quad (3)$$

Global max pooling In case of global max pooling (GMP, see [20] for details), the classification score writes:

$$q_k^{\text{GMP}} = \sum_{l=1}^{L} w_{kl}^{\text{GMP}} \max_{w,h} f_l(w, h). \quad (4)$$

And the class activation map associated to GMP is:

$$a_k^{\text{GMP}}(w, h) = \sum_{l=1}^{L} w_{kl}^{\text{GMP}} f_l^0(w, h), \quad (5)$$

where:[1]

$$f_l^0(w, h) = \begin{cases} f(w, h) & \text{if } (w, h) = \arg\max_{w',h'} f_l(w', h'), \\ 0 & \text{otherwise.} \end{cases}$$

Global top-N average pooling Intuitively, while average pooling takes all the pixels into account, max pooling takes only one value into account. In between, global top-N average pooling [22] (GNAP) takes the average of the N largest values in the feature map. Thus, both average and max pooling can be seen as particular cases of top-N average pooling with $N = WH$ and $N = 1$ respectively. More formally, if $\eta \in [0, 1]$ defines the proportion of pixels in the feature map to be averaged, we set $N_\eta = 1 + \lceil \eta (WH - 1) \rceil$, so that $N_0 = 1$ and $N_1 = WH$. With this notation, the top-N_η average pooling writes:

$$q_k^{\text{GNAP}} = \sum_{l=1}^{L} w_{kl}^{\text{GNAP}} \frac{1}{N_{\eta_l}} \sum_{(w,h) \in \mathcal{N}_l^{\eta_l}} f_l(w, h), \quad (6)$$

where $\mathcal{N}_l^{\eta_l}$ is the set of indices corresponding to the largest N_{η_l} values of f_l. The associated class activation maps are:

$$a_k^{\text{GNAP}}(w, h) = \sum_{l=1}^{L} w_{kl}^{\text{GNAP}} f_l^{\eta_l}(w, h), \quad (7)$$

[1] The choice of the notation f_l^0 will become clear later on.

with:

$$f_l^{\eta_l}(w,h) = \begin{cases} f(w,h) & \text{if } (w,h) \in \mathcal{N}_l^{\eta_l}, \\ 0 & \text{otherwise.} \end{cases}$$

We now remark that the notation f_l^0 is justified since GNAP with $\eta = 0$ corresponds to GMP. In addition, we note that GAP can be expressed as GNAP with $\eta = 1$, and thus we can write: $a_k^{\text{GAP}}(w,h) = \sum_{l=1}^{L} w_{kl}^{\text{GAP}} f_l^1(w,h)$.

Even if the top-N average pooling may seem a good idea that generalizes the concept of average and max pooling, we are left with the tedious task of setting η_l. In order to avoid heuristics or the unaffordable process of cross-validation, we present an efficient way to estimate the gradient of the loss with respect to η_l, so that learning η_l is included within the stochastic gradient descent optimization.

3.3. Global learned top-N average pooling

Providing a formulation of the gradient with respect to η_l requires understanding the behavior of the top-N_η average with respect to η, since the dependence of q_k^{GNAP} with η_l is not differentiable. In this section we propose a very efficient and intuitive way to approximate this gradient. Very importantly, the definition of the top-N average pooling given above, and thus the formalization in this section, are general and independent of the applicative scenario.

We assume the back-propagation algorithm is able to compute the derivatives of the loss $\mathcal{L}$ with respect to q_k^{GNAP}, so that we can use the chain rule to compute the derivative with respect to η_l using:

$$\frac{\partial \mathcal{L}}{\partial \eta_l} = \sum_{k=1}^{K} \frac{\partial \mathcal{L}}{\partial q_k^{\text{GNAP}}} \frac{\partial q_k^{\text{GNAP}}}{\partial \eta_l}. \tag{8}$$

We only require now to give an expression for $\frac{\partial q_k^{\text{GNAP}}}{\partial \eta_l}$. In order to do that, we first observe that, from (6) we have:

$$\frac{\partial q_k^{\text{GNAP}}}{\partial \eta_l} = \sum_{l=1}^{L} w_{kl}^{\text{GNAP}} \frac{\partial g_l(\eta_l)}{\partial \eta_l}, \tag{9}$$

where we defined $g_l(\eta_l) = \frac{1}{N_{\eta_l}} \sum_{(w,h)\in\mathcal{N}_l^{\eta_l}} f_l(w,h)$.

We need to compute the derivative of $g_l(\eta_l)$ with respect to η_l. Unfortunately, the function $g_l(\eta_l)$ is not differentiable with respect to η_l everywhere. Moreover, at the points where the derivative is well-defined, it does not describe the trend of g_l. Indeed, the derivative is undefined at integer multiples of $\delta = (WH-1)^{-1}$ and zero elsewhere. Figure 3 shows an example of $g_l(\eta_l)$ as a function of η_l. In all, we opt to ignore the exact derivative (when available) and try to understand the trend of the function g_l instead.

We adopt a very intuitive strategy that leads to an efficient implementation: approximate $g_l(\eta_l)$ by a second degree polynomial (parabola) and use the derivative of the latter as a proxy for the trend of the original function. One

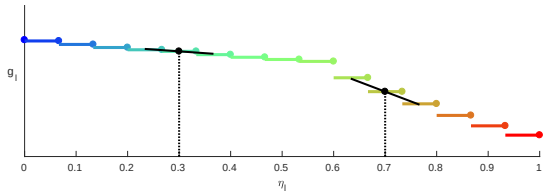

Figure 3. Example of the dependency of g_l with η_l. Even if the original function is clearly non-differentiable, we can approximate the trend of the function very efficiently (black lines).

could think that fitting L parabolas (one per channel) at every backward pass of the network is computationally intensive, since the coefficients of these parabolas need to be computed. However, if we carefully choose the fitting points of the parabola, the estimate of the derivative comes almost for free. Indeed, if η_l^0 denotes the current value of η_l, we fit the parabola at abscissae $\eta_l^0-\delta, \eta_l^0, \eta_l^0+\delta$, because the corresponding values $g_l(\eta_l^0-\delta), g_l(\eta_l^0), g_l(\eta_l^0+\delta)$ are the top-$N_{\eta_l^0}-1$, $N_{\eta_l^0}$, $N_{\eta_l^0}+1$ averages respectively. Moreover, the derivative of such fit parabola at η_l^0 writes:

$$\left.\frac{\partial g_l(\eta_l)}{\partial \eta_l}\right|_{\eta_l^0} = \frac{g_l(\eta_l^0+\delta) - g_l(\eta_l^0-\delta)}{2\delta}. \tag{10}$$

Very importantly this strategy comes at almost no computational cost when compared to performing only the forward pass. This is because the most computationally intense operation is sorting the pixels of the feature map (this is required by the forward pass anyway).[2] Once this is done, we need to compute the top-$N_{\eta_l^0}-1$ average, and update it to obtain the top-$N_{\eta_l^0}$ average and the top-$N_{\eta_l^0}+1$ average, but the overall computational cost is an average of $N_{\eta_l^0}+1$ real numbers. While the top-$N_{\eta_l^0}$ is used for the forward pass, the other two averages are used to estimate the trend of $g(\eta_l)$ using (10), to further update the value of η_l.

When back-propagating down to the layer below, the memory requirements of the LENA layer are slightly higher than max pooling or average pooling. This is because this layer requires to store the $N_{\eta_l^0}$ pixel positions of the feature map that contributed to the forward pass, so that the layer propagates the error only to these pixels. Formally:

$$\frac{\partial g_l(\eta_l)}{\partial f_l(w,h)} = \begin{cases} (N_l^{\eta_l^0})^{-1} & \text{if } (w,h) \in \mathcal{N}_l^{\eta_l^0}, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

We expect the LENA layer to be able to learn from the data which channels need to go through average pooling, which ones through max pooling and which ones require an intermediate option. Before describing the experimental protocol and showing the results, we briefly discuss how do we include objectness maps in our viraliency framework.

[2] Our CPU implementation in the worst case (when fine-tunning only LENA) increases the iteration time by 1.5% compared to GAP/GMP.

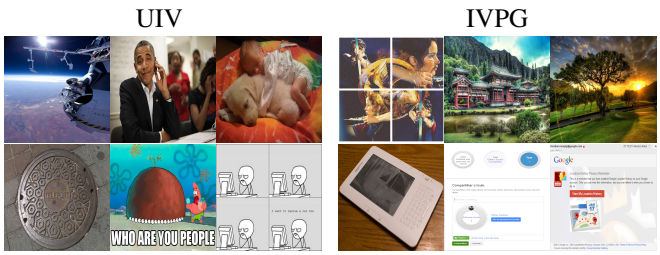

Figure 4. Sample most (top) and least (bottom) viral images from the UIV (left) and the IVGP (right) datasets.

3.4. Incorporating objectness

Intuitively, virality is related to the presence of objects in the images and in order to ascertain the veracity of this statement, we also devise a straightforward strategy to include objectness information in our formulation. We choose to use objectness maps that in our case correspond to the class activation maps of [27] and are computed in the following way. First, we classify all the training images with AlexNet pretrained on ImageNet to extract the 30 most activated classes in the datasets we use. We then generate the class activations maps of these classes for each image of the test and training sets. The objectness maps are then concatenated[3] to the feature maps of the siamese network (right before the global pooling, hence to `Conv N` in Figure 2). An extra convolutional layer is used to fuse the objectness maps with the latent concepts, and produce the same number of feature maps, that now include objectness information.

4. Experimental validation

4.1. Datasets and experimental protocol

In order to assess the effectiveness of the proposed approach for virality prediction and localization, we perform experiments on two recently published datasets: the understanding image virality (UIV) dataset [7] and the image virality on GooglePlus (IVGP) dataset [14]. We also evaluate LENA in the PASCAL VOC dataset [11] for object localization. In the following, we describe the experimental protocol, including datasets, network architectures and baselines.

Datasets. The **UIV** dataset [7] consists on 10K+ Reddit images with the associated virality score (a detailed explanation of the dataset construction can be found in the original paper). In [7] the data are organized in pairs, such as to predict relative virality scores, and a training and a test set of 4,550 and 489 images pairs are created. Since the insufficiency of data can easily lead to overfitting problems when training deep architectures, we created a much larger dataset for our experiments. Inspired by how the training and test sets are generated in [7], we randomly created a test set of 2,965 image pairs, taking one sample from the 250 most viral images and one from the 250 least viral images. The training set consists on 18,182 randomly generated image pairs, containing one image with above-median virality and one image with below-media virality. Very importantly, we ensured that the training and test sets are disjoint, so that the test pairs are not used during training.

The **IVGP** dataset presented in [14] consists on GooglePlus images of the top followed profiles in this social network. Images were gathered from the most followed profiles to avoid "friendship dynamics" when reposting content and to ensure enough visibility to each image (see [14] for more details). Intuitively this guarantees that all images go through a minimum number of impartial views, and therefore the measures of virality are significant. The original dataset has 150K+ images, but only 90K are currently available.[4] After assessing their virality with the formulation in [7], we generated 11,704 and 2,926 image pairs for training and test respectively. Each image pair consists of one of the 15K most viral images and one of the 15 K least viral images. The training and test sets are disjoint.

Sample images of both datasets are shown in Figure 4.

Network architectures. We used three different base architectures for the proposed siamese network. Specifically, we consider the five convolutional layers of AlexNet [19] (Alex5), similarly to [27] we append two convolutional layers with 512 units, `3x3` kernel and stride 1 to Alex5 leading to 7 convolutional layers (Alex7)[5] and finally the 13 convolutional layers of the VGG16 network [25] (VGG13). All the networks have been fine-tuned for 10K iterations and the learning rate policy was fixed for all experiments using the same base architecture. The weights of these networks were initialized from pretrained ImageNet models. The training protocol details can be found in the supplementary material. The LENA code is publicly available at: https://github.com/xavirema/lena_pooling.

Baselines. We compare the proposed method with several baselines. Deza & Parikh [7] is the only previous approach addressing virality prediction and considers an SVM with features extracted from the sixth layer of the AlexNet network. Importantly, we could not use visual attributes because they are available only for a small subset of the dataset in [7] (and not for [14]) and in addition it is not straightforward to extract them in an automatic manner. In order to evaluate the effectiveness of the proposed LENA layer, we compare it with global max pooling (GMP) [20], global average pooling (GAP) [27] and with top-N average pooling (GNAP) [22]. Regarding the LENA layer, we try different initializations for η_l, namely 0, $1/2$ and 1, and denote them by GLENAP-0, $-$½ and -1.

[3] A bilinear filter implemented as a deconvolutional layer was used to resize the objectness maps into the size of the feature maps, if needed.

[4] Image are available only through the users' public profile.

[5] Both the architecture and the trained model used to initialize it are publicly available in https://github.com/metalbubble/CAM.

Table 1. Accuracy results on predicting virality with Alex7 on the UIV and IVGP datasets. The two last columns, →UIV and →IVPG correspond to cross-dataset results, training in IVPG and testing in UIV and viceversa, respectively.

Method	UIV	IVGP	→UIV	→IVPG
Deza & Parikh [7]	59.5	65.4	51.4	48.5
GAP [27]	61.4	68.0	54.0	52.0
GMP [20]	62.2	71.0	56.2	52.3
GNAP [22]	62.3	71.2	**57.3**	**52.7**
GLENAP−0	**62.7**	71.3	55.9	52.0
GLENAP−½	61.5	71.6	57.1	**52.7**
GLENAP−1	62.6	**72.7**	55.9	52.3

4.2. Predicting virality

We first evaluate the performance on virality prediction. Table 1 shows the accuracy of the different methods on the UIV and IVPG datasets. The first two columns correspond to the standard training and testing, while the third and fourth columns to cross-dataset experiments. For instance, →UIV means training on IVPG and testing on UIV.

We first observe that all the end-to-end trainable models systematically outperform the SVM-based method in [7],[6] which is in agreement to the findings of the community in a wide variety of vision applications. Also, in agreement with previous studies [27], we found that embedding a global pooling layer into a specific architecture (*e.g.* AlexNet) is outperformed by considering a corresponding fully connected layer within the same network (by 1.5 and by 1.7 points on UIV and IVGP, respectively, numbers not reported in the table). We remark that this slight increase of performance comes at the cost of completely losing the ability to localize the viral parts of the image (as also discussed in [27] for weakly-supervised object detection). Thirdly, for the "within dataset" experiments (training and test belong to the same dataset), we remark that at least one of the initializations of the proposed LENA pooling is systematically outperforming all the baseline methods. Regarding the cross-dataset experiments, we highlight the inability of all the methods to maintain the same virality recognition performance. Finally, when comparing the performance dataset-wise we realize that: (i) the accuracy on the within dataset experiments for IVPG are higher than for UIV and (ii) more importantly, the performance on →UIV are systematically better than for the →IVPG experiments. This would suggest that the IVPG contains data allowing better generalization than UIV. For the rest of the present study, we intensively exploit the IVPG dataset.

4.3. The use of objectness maps

Naturally, one may wonder if prior knowledge of which objects are in the image (and where are them) could help predicting virality. In order to analyse this aspect, we performed experiments that take the objectness of the images into account. Table 2 reports the accuracy results on virality prediction with the three base architectures (Alex5, Alex7 and VGG13) on the IVGP dataset with (w/) and without (w/o) objectness (the third column of Table 2 corresponds to the second column of Table 1).

Table 2. Virality prediction accuracy for the three base architectures (Alex5, Alex7 and VGG13) with (w/) and without (w/o) objectness on the IVGP dataset.

Method	Alex5		Alex7		VGG13	
	w/o	w/	w/o	w/	w/o	w/
GAP [27]	68.0	71.2	68.0	71.1	71.1	74.1
GMP [20]	68.4	70.7	71.0	71.6	74.1	73.2
GNAP [22]	69.3	**71.3**	71.2	71.6	72.4	74.9
GLENAP−0	**69.7**	70.2	71.3	**72.6**	73.4	**75.6**
GLENAP−½	67.9	69.8	71.6	72.2	73.4	74.0
GLENAP−1	66.9	69.1	**72.7**	**72.6**	**75.7**	74.7

We first observe that the use of objectness is advantageous: in most of the cases the accuracy raises when adding objectness. Second, we notice that VGG13 results are systematically higher than Alex7 ones, independently of the objectness. In other words, for a given method the minimum over the fifth and sixth columns is always higher than the maximum over the third and fourth columns. A similar trend is found when comparing the performance of the Alex7 and Alex5 networks. Finally, we highlight that, as in the case of Table 1, the best accuracy across the initializations of LENA is consistently superior to the three baselines, independently of the base architecture and of the use of objectness maps (except for Alex5 with objectness). This reinforces the idea that learning η within a global pooling scheme at the top of a convolutional network helps the virality prediction task.

4.4. Viraliency maps

One of the prominent features of the global pooling layers is their capacity to implicitly localize the objects and concepts through the analysis of the class activation maps [20, 27]. More importantly, this is achieved with weak supervision: no localization information is used during training. In the precise case of virality prediction these maps correspond to the virally salient parts of the image, *i.e.* the viraliency maps. Figure 5 plots the viraliency maps superposed to three of the top viral images of the IVGP dataset for GAP (3), GMP (5) and GLENAP−1 (7) (denoted by GLENAP from now on), without obectness in the first three columns, and including objectness in the last three columns. When no objectness is used, the viraliency maps associated to the three pooling layers have clear distinct behaviors. Indeed, GAP seems to be able to point to a fairly compact zone of the image responsible for virality, while GMP highlights several small zones, thus leading to a viraliency map that is spread over the image. The proposed LENA pooling

[6] We attribute the small improvement of the baseline over what was reported in [7] to our larger dataset.

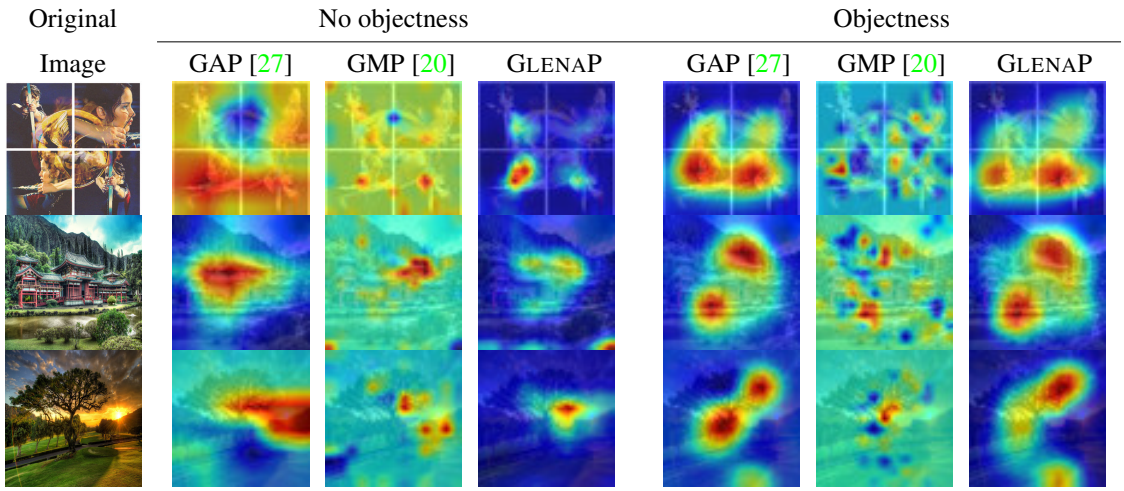

Figure 5. Viraliency (class activation) maps for three images of the IVGP dataset using the Alex7 base network. The four columns correspond to (left to right) the original image, viraliency for GAP, for GMP and for GLENAP−1, without and with objectness.

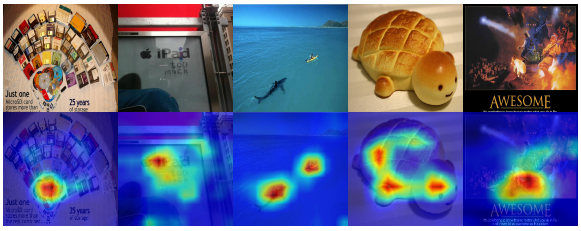

Figure 6. Viraliency maps from GLENAP of five images of IVGP.

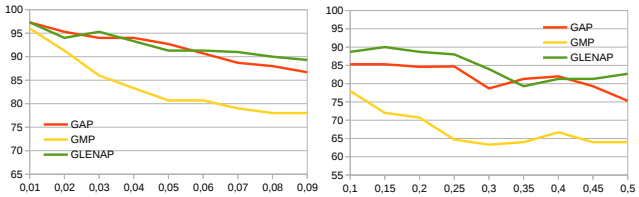

Figure 7. Viraliency maps evaluation by comparing the original image to the ρ hottest pixels of the map.

is able to spot a few (2 to 4) zones in the image responsible for virality. The use of objectness seems to bring the three global pooling layers towards a more similar behavior, as expected. Indeed, we can see that the viraliency maps of GAP and GLENAP are now close to each other. Regarding GMP, even if the spread behavior observed when no objectness was used is still dominant, the more viral zones are aligned with the big bulbs in GAP and GLENAP when objectness is used. Interestingly, we can observe in the examples that the use of objectness is a two-edged sword. For instance, the viraliency map of the second image with GLENAP without objectness contains a very hot spot on the bottom right corner of the image, which disappears when using objectnes. At the same time, a bulb in the lower part of the third viraliency map for GLENAP appears when objectness is included, but the main spot is widen to include the tree. These results confirm our initial hypothesis that richer global pooling strategies are adequate when recognizing subjective/abstract properties of images. More pictures of viraliency maps for all pooling strategies are shown in the supplementary material.

Figure 6 shows some viraliency maps for the LENA pooling that are worth to be discussed. First of all, we observe that most of these viraliency maps consist on different virally salient areas, thanks to the ability of the proposed layer to pool information from several pixels. The capacity of the network to highlight the viral parts of an image using only visual information is remarkable: the network is partially learning the complexity associated to virality. Two immediate explanations of this phenomenon are the potential bias towards text and objects, as suggested by the second and third images respectively. However, the text in the first and fourth images does not belong to the highlighted region and the viraliency map of the first and fourth images does not match with the objects' regions. The fourth image is of special interest, because the network suggests that virality arises from the combination of "pastry" and "turtle".

In order to provide quantitative evaluation of the viraliency maps, we trained a siamese VGG19, this time including the fully connected layers. We then generated images by keeping the ρ hottest proportion of pixels (according to the heat map of GAP, GMP or GLENAP using Alex7) and masked out the rest of the image. Intuitively, ρ is a budget of pixel positions that each pooling strategy places according to its viraliency map. Each of the masked images was paired with the original image and given the label "1" (the masked image is more viral than the original). Figure 7 reports the accuracy of the three pooling strategies on the ρ ranges $1\%-10\%$ (left) and $10\%-50\%$ (right). First we notice that GAP and GLENAP clearly outperform GMP (and this is consistent with Figure 5). Regarding GLENAP and GAP, while for low values or ρ the performance is equivalent, GLENAP outperforms all other baselines when larger image content is available.

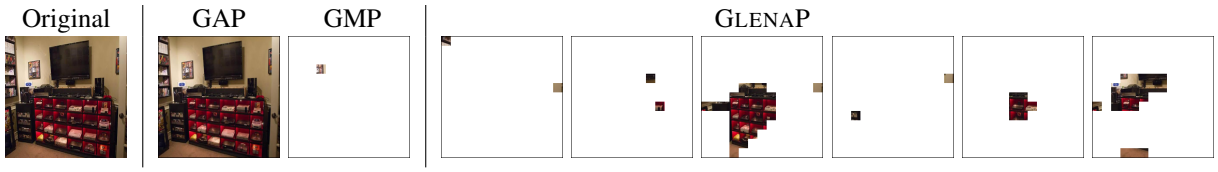

Figure 8. Example of the receptive fields of different channels of the GLENAP with Alex7 on an image from IVGP (on the left). The two most-left images correspond to the two extremes GAP (top) and GMP (bottom), where respectively all pixels and one pixel of the feature map are averaged. The rest corresponds to receptive fields of different sizes (*i.e.* η_l) learned by the LENA pooling layer.

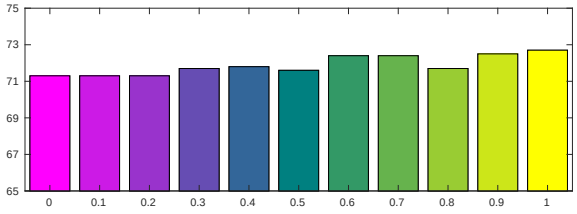

Figure 9. Sensitivity analysis of the accuracy of Alex7 on the IVGP dataset to different initialization values of η.

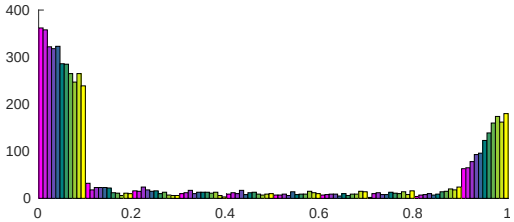

Figure 10. Histogram of the values of η after convergence of the Alex7 network on the IVGP dataset for different values of initial η. The colors correspond to each of the initializations in Figure 9.

To further push the understanding of the viraliency maps, Figure 8 plots the receptive field of different output activations of the GLENAP for an image of IVGP. The two most left correspond to channels averaging over all pixels (as GAP does) or over a single pixel (as GMP does), the rest correspond to channels with intermediate values of η_l. In terms of information propagation, the advantage of LENA is two-fold. On one hand many image regions can contribute to forward information to the classification layer, thus enlarging the forward capacity of max-pooling. On the other side the error is back-propagated only to those regions that contributed during the forward pass, leading to a more efficient strategy than average-pooling (that propagates the error everywhere). We believe that this low-level behavior explains the effectiveness exhibited by the LENA pooling.

4.5. Weakly supervised localization

We further evaluated the GLENAP layer in the task of weakly supervised localization on the PASCAL VOC dataset. The procedure to generate bounding boxes from viraliency maps is the one used in [27] (with the threshold set to 30), and the localization metrics are the ILSVRC standard metrics (as opposed to previous studies on the same dataset [20]) with the overlap threshold set to 30%. The localization accuracy for GAP, GMP and GLENAP1 is 58.18%, 15.38% and 59.74% respectively, proving the efficiency of the LENA for weakly supervised localization.

4.6. Sensitivity analysis

We perform an analysis to study the sensitivity of the proposed LENA layer to the initial value of η. In details, we initialize the Alex7-based siamese structure with 11 different values of η (from 0 to 1 every 0.1) and plot the accuracy of these different trainings in Figure 9. This confirms our intuition that the final accuracy does not exhibit strong dependencies to the initial value of η. To further analyze the potential behavioral differences of with respect to the initial values of η, we plot the histogram of converged values of η in Figure 10. The colors on this figure correspond to the colors on Figure 9. Very importantly, we observe that independently of the initialization roughly one-third of the channels converge to average pooling and two-thirds to max pooling. This provides an explanation of why the GLENAP strategy outperforms other global pooling strategies such as global average or max-pooling.

5. Conclusions

In this paper we addressed the task of simultaneous prediction and localization of virality using only visual information and image-level annotations. To this aim, we propose to use an end-to-end trainable siamese deep architecture with three main blocks: a convolutional front-end, a global pooling layer and a classification layer. Within this context, we introduced the LENA pooling layer, that estimates the optimal η per each convolutional feature map. We performed an extensive experimental evaluation that shows the effectiveness of the proposed architecture, and of the LENA layer, for the simultaneous prediction and localization of virality. In the future we will assess the usefulness of such architectures for other subjective properties of visual data, as well as develop methods able to exploit other metadata related to virality, such as the comments in the social network associated to the image. Additionally, we plan to identify different temporal patterns of virality and design methods to recognize them in an automatic manner. Method-wise, we would like to delve in how to robustify the learning scheme with automatic data weighting [13] or low-rank constraints [2, 26].

References

[1] X. Alameda-Pineda, E. Ricci, Y. Yan, and N. Sebe. Recognizing emotions from abstract paintings using non-linear matrix completion. In *IEEE CVPR*, 2016. 1

[2] X. Alameda-Pineda, Y. Yan, E. Ricci, O. Lanz, and N. Sebe. Analyzing free-standing conversational groups: A multimodal approach. In *ACM Multimedia*, 2015. 8

[3] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with convex clustering. In *IEEE CVPR*, 2015. 2

[4] B. Celikkale, A. Erdem, and E. Erdem. Visual attention-driven spatial pooling for image memorability. In *IEEE CVPR Workshops*, 2013. 1, 2

[5] R. G. Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *IEEE CVPR*, 2014. 2

[6] M. De Nadai, R. L. Vieriu, G. Zen, S. Dragicevic, N. Naik, M. Caraviello, C. A. Hidalgo, N. Sebe, and B. Lepri. Are safer looking neighborhoods more lively?: A multimodal investigation into urban life. In *ACM Multimedia*, 2016. 1

[7] A. Deza and D. Parikh. Understanding image virality. In *IEEE CVPR*, 2015. 1, 2, 5, 6

[8] S. Dhar, V. Ordonez, and T. L. Berg. High level describable attributes for predicting aesthetics and interestingness. In *IEEE CVPR*, 2011. 2

[9] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros. What makes paris look like paris? *ACM Transactions on Graphics*, 31(4), 2012. 1

[10] A. Dubey, N. Naik, D. Parikh, R. Raskar, and C. A. Hidalgo. Deep learning the city: Quantifying urban perception at a global scale. In *ECCV*, 2016. 2

[11] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015. 5

[12] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. arXiv preprint 1508.06576, 2015. 1

[13] I.-D. Gebru, X. Alameda-Pineda, F. Forbes, and R. Horaud. EM algorithms for weighted-data clustering with application to audio-visual scene analysis. *IEEE TPAMI*, 38(12):2402–2415, 2016. 8

[14] M. Guerini, J. Staiano, and D. Albanese. Exploring image virality in GooglePlus. In *Int. Conf. on Social Comp.*, 2013. 1, 5

[15] P. Isola, J. Xiao, D. Parikh, A. Torralba, and A. Oliva. What makes a photograph memorable? *IEEE TPAMI*, 36(7):1469–1482, 2014. 1, 2

[16] P. Isola, J. Xiao, A. Torralba, and A. Oliva. What makes an image memorable? In *IEEE CVPR*, 2011. 2

[17] A. Khosla, W. A. Bainbridge, A. Torralba, and A. Oliva. Modifying the memorability of face photographs. In *IEEE CVPR*, 2013. 1, 2

[18] A. Khosla, A. Das Sarma, and R. Hamid. What makes an image popular? In *WWW*, 2014. 1, 2

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 5

[20] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *IEEE CVPR*, 2015. 2, 3, 5, 6, 7, 8

[21] K.-C. Peng, T. Chen, A. Sadovnik, and A. C. Gallagher. A mixed bag of emotions: Model, predict, and transfer emotion distributions. In *IEEE CVPR*, 2015. 1

[22] L. Porzi, S. Rota Bulò, B. Lepri, and E. Ricci. Predicting and understanding urban perception with convolutional neural networks. In *ACM Multimedia*, 2015. 1, 2, 3, 5, 6

[23] A. Sartori, D. Culibrk, Y. Yan, and N. Sebe. Who's afraid of itten: Using the art theory of color combination to analyze emotions in abstract paintings. In *ACM Multimedia*, 2015. 2

[24] A. Siarohin, G. Zen, C. Majtanovic, X. Alameda-Pineda, E. Ricci, and N. Sebe. How to make an image more memorable? a deep style transfer approach. In *ACM ICMR*, 2017. 2

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[26] S. Tulyakov, X. Alameda-Pineda, E. Ricci, L. Yin, J. F. Cohn, and N. Sebe. Self-adaptive matrix completion for heart rate estimation from face videos under realistic conditions. In *IEEE CVPR*, 2016. 8

[27] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *IEEE CVPR*, 2016. 1, 2, 3, 5, 6, 7, 8