Efficient Visual Saliency detection with Deep Learning

1st Erik Perillo
Institute of Computing
University of Campinas
Campinas, Brazil
erik.perillo@gmail.com

2nd Esther Colombini
Institute of Computing
University of Campinas
Campinas, Brazil
esther@ic.unicamp.br

***Abstract*—Visual saliency is an important component of attention. It helps animals survive and can also be used by computer vision applications to filter out irrelevant information from high volumes of data. In this work, we present a new convolutional neural network designed for detecting visual saliency, with architecture and data pre-processing methods specific for the task of visual saliency detection. Experiments carried out with the MIT300 benchmark presented state-of-the-art performance and a parameter reduction of 3/4 compared to similar models.**

***Index Terms*—visual saliency, deep learning, computer vision**

I. Introduction

One of the most challenging unsolved problems in Artificial Intelligence is vision. However, it is fundamental for the conception of systems that interact in the real physical world. Such systems would be useful for applications in areas like domestic services, industry, and agriculture, with great potential for the benefit of society.

Vision is remarkable data and computationally intensive. In humans, approximately half of the brain is involved in vision-related tasks [6]. Even our minds cannot handle all the sheer amount of sensorial information received every second. In order to deal with this amount of data, humans have attentional systems, a fundamental mechanism that, among other functions, filters out irrelevant information – either visual or from other senses– and helps us focusing our cognitive processes on what is important at a given moment. These facts are a strong evidence that in order to help to solve vision problems, attention should be applied.

Visual attention can be defined as the delimitation of a certain spatial the region on an image for further cognitive processing [14]. The phenomenon emerges from two fundamentally different processes: the *top-down* mechanism that implements our longer-term cognitive strategies by biasing attention according to one's interests (e.g. find a red apple in a tree because of hunger, which will make red be more recognizable on the scene), and the *bottom-up* mechanism [4], a process generated through external stimuli that captures one's attention from its conspicuousness level. In this work, we focus on the latter, also named visual saliency.

Visually salient regions on images are usually represented by *saliency maps* (Figure 1). In these maps, images are generated such that areas with high-valued pixels express high saliency on the original image, whereas regions with low-valued pixels represent low saliency. Datasets with such maps are obtained by collecting eye-fixation data from humans while observing the scenes.

(a) (b)

Fig. 1. Example of visual saliency. b) is the saliency map where brighter pixels (warmer colors) represent regions more salient to humans on the original image a).

A. Related work

Early computational models of visual saliency were generally built based on filtering of images for extraction of a pre-selected set of features considered important for *bottom-up* attention. *Vocus* [7] is a computational model that extracts features which are shown to be naturally salient to humans such as color/luminance contrast and orientation from different scales of the image.

A rapid change of paradigm occurred around 2015 when *Deep Learning* techniques showed to be very effective in the generation of saliency maps. *Salicon* [8] demonstrated that the use of convolutional neural networks with weights initialized from image classification networks, e.g. *VGG-16* [12] could considerably increase the similarity of computed maps to those generated from humans, whereas *Salicon* uses different scales of the image as input to capture relevant information in the context of saliency, using the same network weights for each dimension. *ML-Net* [5] uses the output of different layers of *VGG-16*, combining them in many dimensions and various levels of abstraction. *DeepFix* [10] extends a pre-trained model with inception [13] layers – which use information from different scales of the image – and a component for center bias, a phenomenon that arises from our tendency to take pictures with relevant objects at the center of the image. *Salnet* [11] explores two models: a shallow convolutional network followed by a fully-connected layer and a convolutional neural network with first layers' weights initialized from *VGG-16*. Models that use weights from VGG-16 all use RGB images as input, subtracting channel-wise the mean of the dataset for each RGB channel.

However, current state of the art models are in general quite expensive computationally, partially because most of them are based on big pre-trained networks. The convolutional layers of *VGG-16* are composed of around 14.7 million parameters. While pre-trained weights from classification tasks showed to be effective for saliency prediction, it is reasonable to question whether creating a proper network from scratch could yield a smaller amount of parameters that are more efficient for the sole task of saliency prediction. Also, there are some data processing methods from previous work on psychology-based models that were not used in current models but are considered worthwhile to explore, such as using global information from the scene and a color space more closely related to human vision.

In this context, this work aims at building a visual saliency model that is a) effective, yielding results similar to another state of the art models, and b) relatively simple and computationally efficient. It is important that both criteria are matched in order to extend the model in the future for video and real-time computer vision applications such as navigating robots.

II. Proposed model

Figure 2 shows the overall architecture of the fully convolutional neural network proposed in this work. It extracts features from increasingly smaller dimensions of the input image. The network is composed of four main blocks:

1) The first level extracts low-level features from the input image, of dimensions $W \times H \times 3$ (width, height, depth), using a single layer with 48 convolution filters with ReLu activation followed by max-pooling that reduces the image by a factor of two. Experiments showed that by further decreasing the number of filters in this layer considerably hurts performance, which makes sense because it is important to capture high spatial frequency and high contrast information in the context of visual saliency.
2) The second level extracts low-medium level features from the input of dimensions $W/2 \times H/2 \times 48$ using two layers with 64 and 96 convolution filters, respectively, followed by ReLU activation and max-pooling.
3) The third level extracts medium-high level features from input with dimensions $W/4 \times H/4 \times 96$ using four convolution layers in sequence with 128, 128, 144 and 144 filters. Every convolution layer is followed by ReLu. Max-pooling is carried out at the end. A considerable depth in this level was found to be important for the network's performance.
4) The fourth and last level is composed of eight inception blocks that extract high level features from the input with dimensions $W/8 \times H/8 \times 96$. Great level of depth and Inception blocks were found to be very important at this level. A 1×1 convolution makes a linear combination of the output maps at the end of the 8 inception blocks, followed by ReLu, producing the final saliency map of dimensions $W/8 \times H/8 \times 1$. The saliency map is then resized to the original dimensions using bicubic interpolation.

Figure 3 illustrates the inception architecture [13] used in each block where filters of size 5×5, 3×3 (both preceded by 1×1 convolutions in order to reduce the number of input filters), 1×1, and a max-pooling of size 3×3 are applied. Each of these operations is executed in parallel from the same input and the outputs are concatenated at the output. Inception allows the network to use information from different spatial dimensions as well as previous layers (lower level saliency information) in the final map computation, which is considered to be important for visual saliency. The network has a total of 3717841 parameters, a very low number compared to other models. Table I details the filter configuration for the inception layers.

A. Data pre-processing

Input images were resized to dimensions $320 \times 240 \times 3$. Each image is normalized channel-wise by the subtraction of the channel mean and division by the standard deviation:

$$C = \frac{C - \mu_C}{\sigma_C}$$

Most models used for comparison and cited in this work use RGB images normalized channel-wise using statistics computed from the dataset. However, visual saliency is highly connected to the context of the image, hence saliency depends on the local context.

In our work, images are converted from RGB color space to the LAB color space. *Vocus* [7] cites that the LAB color space is more closely related to human vision once it encompasses red-green, yellow-blue and luminance maps.

B. Implementation

The network was implemented in two different frameworks: *Theano* `0.9.0.dev` along with *Lasagne* `0.2.dev1` and *Tensorflow* `1.4.0`. Experiments were run on a machine with *Ubuntu 16.04 LTS* and kernel *Linux* `4.8.0-54-generic`. Training was conducted on a GPU *NVIDIA GTX 1080* and the code is available at `https://goo.gl/5JZMjb`

C. Training

Two datasets were considered: *SALICON* [8], with 15000 images, and *Judd* [9], with 1003 images. The network was trained using Stochastic Gradient Descent with Nesterov Momentum of 0.9. *SALICON* was first used with data augmentation by flipping images horizontally and vertically and the target normalized by mean-std (Last conv layer had ReLu removed in this step). Mean-std normalization of targets was applied because it led to faster convergence.

The loss function to be minimized was the *Cross Correlation*, because it penalizes symmetrically false positives and

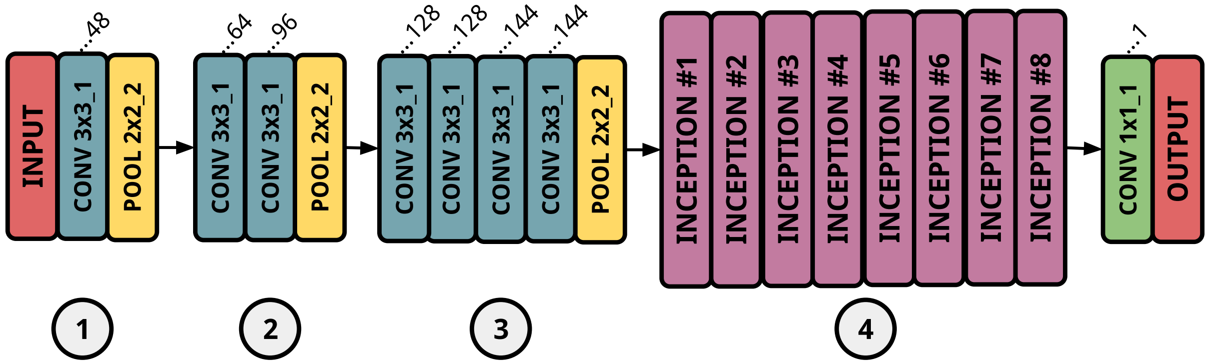

Fig. 2. Overview of the network. Filters sizes are in format width×height_stride.

TABLE I

NUMBER OF FILTERS USED IN EACH INCEPTION BLOCK.

Block	pool	conv 1×1	3×3 reduce	conv 3×3	5×5 reduce	conv 5×5
1	96	128	96	192	48	96
2	64	128	80	160	24	48
3	64	128	80	160	24	48
4	64	128	96	192	28	56
5	64	128	96	192	28	56
6	64	128	112	224	32	64
7	64	128	112	224	32	64
8	112	160	128	256	40	80

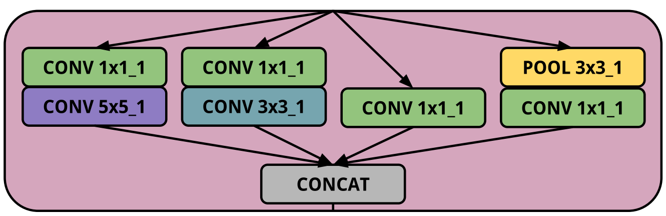

Fig. 3. Inception block layout.

false negatives. A more detailed explanation of the metric is given in section III. Training iterated for 5 epochs with learning rate of 0.009 and then for 3 epochs with learning rate of 0.001. Then, unit normalization on targets started being used. The network was trained for 1 epoch with learning rate of 3×10^{-5} and L2 regularization of 10^{-4}. Finally, Judd dataset was used with data augmentation by flipping images horizontally and the target normalized by unit normalization. Training iterated for 2 epochs with learning rate of 5×10^{-5} and L2 regularization of 3×10^{-5}. Batch sizes were 10 for SALICON and 2 for Judd. The complete training process took around two and a half hours.

III. MODEL EVALUATION

In order to assess how the maps generated by a model are similar to those generated by humans, a variety of metrics has been used in the literature [3]. Some of the most used metrics are:

- *Area Under ROC Curve (AUC)*. Area under curve of true positive rate and false positive rate which is calculated from the binarized saliency map and the fixation points from human data for different thresholds.
- *NSS*. Defined as:

$$NSS(P,Q) = \frac{1}{N}\sum_{i=1}^{N} \bar{P}_i Q_i$$

  Where N is the number of fixation points in ground truth, Q is the binary fixation points ground truth and $\bar{P}$ is the saliency map normalized by the standard deviation. NSS tends to penalize more false positives than AUC.
- *Similarity*. Defined as:

$$SIM(P,Q) = \frac{1}{N}\sum_{i=1}^{N} min(P_i, Q_i)$$

  Where P and Q are both continuous saliency maps normalized by their sum.
- *Cross-Correlation (CC)*. Defined as:

$$CC(P,Q) = \frac{cov(P,Q)}{\sigma(P)\sigma(Q)}$$

  Where P and Q are both continuous saliency maps in $[0,1]$. Similarity tends to penalize false negatives more than false positives. CC tends to penalize errors more symmetrically.

Visual saliency detection models are usually evaluated and ranked on *MIT saliency benchmark* [2], which uses the metrics

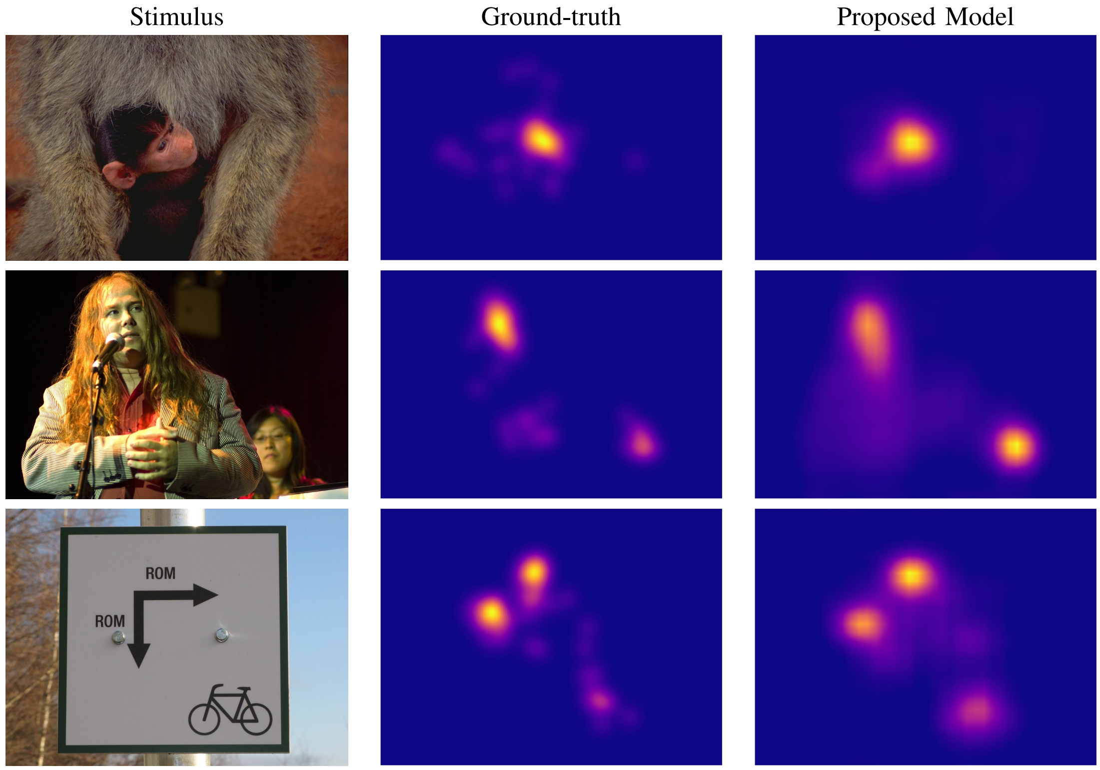

Fig. 4. Examples of predictions made by the proposed model.

TABLE II
STATE OF THE ART MODELS AND METRIC SCORES ON *MIT300 benchmark*.

Model	Num. parameters	AUC-Judd ↑	CC ↑	NSS ↑	Sim ↑	EMD ↓
Baseline: Infinite humans	-	0.92	1.0	3.29	1.0	0
DeepFix	≈16.7 million	0.87	0.78	2.26	0.67	2.04
Salicon	≈14.7 million	0.87	0.74	2.12	0.60	2.62
Proposed Model	**≈3.7 million**	**0.85**	**0.71**	**1.98**	**0.62**	**2.37**
ML-Net	≈15.4 million	0.85	0.69	2.07	0.60	2.53
SalNet	≈25.8 million	0.83	0.57	1.51	0.52	3.31

mentioned above – in addition to a variety of others metrics – to express how close generated saliency maps are to those created from human data. The most used dataset is *MIT300*, which contains 300 images of a variety of scenes and situations. The ground truth data of *MIT300* is held out.

IV. MODEL VARIATIONS STUDY

Some variations of the network are explored in this section. Networks are initially trained using images from *SALICON* dataset and further refined with the *Judd* dataset. Evaluation is carried out on *CAT2000* [1] dataset, which is comprised of 2000 varied images. The results showed in this section were used to guide the decisions in making the final model.

A. Image normalization

For every input image I_i, each channel I_{ic} is normalized by subtracting the mean μ and dividing by the stardard deviation σ. Such values were extracted in two manners:

TABLE III
RESULTS ON CAT2000 DATASET FOR DIFFERENT DATA PRE-PROCESSING TECHNIQUES.

μ, σ source	AUC-Judd ↑	CC ↑	NSS ↑	Sim ↑
Image	0.87	0.73	1.94	0.65
Dataset	0.86	0.69	1.85	0.63

1) From the mean μ_{ic} and standard deviation σ_{ic} of each individual image channel I_{ic};
2) From the mean μ_c and standard deviation σ_c of each channel computed accross all images in the training set.

Table III shows the results. Using normalization by individual image statistics performed better on every analyzed metric, so this was the configuration used in the final model. We hypothesize some reasons for these findings in section II-A.

TABLE IV
NUMBER OF INCEPTION BLOCKS AND PERFORMANCE ON CAT2000 DATASET.

Num. blocks	AUC-Judd ↑	CC ↑	NSS ↑	Sim ↑
2	0.81	0.51	1.34	0.54
4	0.84	0.59	1.56	0.58
8	0.87	0.73	1.94	0.65
16	0.86	0.72	1.90	0.64

TABLE V
RESULTS ON CAT2000 DATASET.

AUC-Judd ↑	CC ↑	NSS ↑	Sim ↑
0.87	0.73	1.94	0.65

B. *Number of inception blocks*

The number of inception blocks was also explored by training networks with 2, 4, 8 and 16 blocks. The network with 2 blocks had all inception blocks except for 1 and 8 (from table I) removed. The network with 4 blocks had blocks 3, 4, 5, 6 removed. The network with 16 blocks had block 3 replicated 3 times in sequence, block 5 replicated 3 times in sequence and block 7 replicated 2 times in sequence.

Table IV shows the results. Performance increases as the number of layers is doubled from 2 to 4 and from 4 to 8, but decreases as the number of layers goes up to 16. One explanation for this is that the hypothesis space becomes larger as the number of parameters increases and the fixed number of training samples causes the more complex network to overfit. We thus choose 8 blocks to be used in the final model.

V. RESULTS

Prediction took an average time of 8 milliseconds. Figure 4 shows some maps generated by the proposed model.

A. *Evaluation on the CAT2000 dataset*

The final model was evaluated on CAT2000 and results are shown in table V.

B. *Evaluation on the MIT300 benchmark*

Table II shows the resulting values for the most common metrics on *MIT300 benchmark*. The proposed model achieved results comparable to those of the state of the art while having, at least, one-fourth of the number of parameters.

VI. CONCLUSION

In this paper, we proposed a novel convolutional neural network for the prediction of visual saliency on images. The proposed model architecture was designed specifically for the task of saliency prediction, with data pre-processing methods specific for the context of saliency prediction. Our methods showed to be effective, yielding to a network with performance on *MIT300 benchmark* consistently among the ten best results on various metrics while having around $3/4$ fewer parameters than another state of the art models.

REFERENCES

[1] A. Borji and L. Itti. Cat2000: A large scale fixation dataset for boosting saliency research. *CVPR 2015 workshop on "Future of Datasets"*, 2015. arXiv preprint arXiv:1505.03581.

[2] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. Mit saliency benchmark. http://saliency.mit.edu/index.html, 2016.

[3] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand. What do different evaluation metrics tell us about saliency models? *arXiv preprint arXiv:1604.03605*, 2016.

[4] E. L. Colombini. *An Attentional Model for Intelligent Robotics Agents*. PhD thesis, 2014.

[5] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara. A deep multi-level network for saliency prediction. *arXiv preprint arXiv:1609.01064*, 2016.

[6] R. S. Fixott. Evaluation of research on effects of visual training on visual functions. *American Journal of Ophthalmology*, 44:230–236, 1957.

[7] S. Frintrop. *VOCUS: a visual attention system for object detection and goal-directed search*. PhD thesis, 2005.

[8] M. Jiang, S. Huang, J. Duan, and Q. Zhao. Salicon: saliency in context. *CVPR*, 2015.

[9] T. e. a. Judd. Learning to predict where people look, 2016.

[10] S. S. S. Kruthiventi, K. Ayush, and R. V. Babu. Deepfix: A fully convolutional neural network for predicting human eye fixations. *arXiv preprint arXiv:1510.02927*, 2015.

[11] J. Pan, E. Sayrol, X. G. i Nieto, K. McGuinness, and N. OConnor. Shallow and deep convolutional networks for saliency prediction. *arXiv preprint arXiv:1603.00845*, 2016.

[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[14] A. M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognit Psychol*, 1980.