Visual Attention with Deep Learning

Erik Perillo[1], **Esther Luna Colombini**[1]

[1]Institute of Computing (IC) – University of Campinas (Unicamp)
Caixa Postal 6176 – 13.084-971 – Campinas – SP – Brazil

erik.perillo@gmail.com, esther@ic.unicamp.br

Abstract. *The high volume of sensorial data in vision is problematic because most of the information is often irrelevant. Humans realize sensorial filtering by what we call attention. We propose the application of Deep Learning for obtaining a visual salience system which behaves similarly to humans. We built a new convolutional neural network with relatively simple architecture, yielding a performance level consistently among the best ten state of the art models in MIT300 benchmark.*

Resumo. *A alta quantidade de dados sensoriais na tarefa de visão é problemática, havendo muitas vezes irrelevância de informação. Nos seres humanos, há um filtro sensorial realizado pela atenção. Propomos a aplicação de Deep Learning para a obtenção de um sistema de saliência visual que se comporte como o dos seres humanos. Construímos uma nova rede neural convolucional de arquitetura relativamente simples e com um desempenho que a coloca consistentemente entre os dez melhores modelos estado da arte no MIT300 benchmark.*

1. Introduction

One of the most challenging unsolved problems in Artificial Intelligence is vision. It is fundamental for the conception of systems that interact with the real, physical world. Such systems would be useful for applications that involve robotics and tasks in domestic houses, industry and agriculture, so there is great potential for the benefit of society.

Vision is remarkably data and computationally intensive: In humans, approximately half the brain is involved in vision-related tasks [Fixott 1957]. Even our brains can't handle all the sheer amount of sensorial information that we receive every second: We have attention, a fundamental mechanism that, among other functionalities, filters out irrelevant information – either visual or from other senses– and helps us focus our cognitive processes on what is important at a given moment. These facts are a strong evidence that, in order to solve vision, we need to have attention.

1.1. What is visual saliency?

Visual saliency can be defined as the delimitation of a certain spatial region on an image for further cognitive processing [Treisman and Gelade 1980]. The phenomenon of visual saliency may emerge from two fundamentally different processes: There is *top-down* attention, an internal stimuli of the agent (e.g. find a red apple in a tree because of hunger, which will automatically make red be reconizable more easily on the scene), and *bottom-up* attention, an external process which captures the agent's attention from abrupt changes in visual stimuli or high contrast. In this work, we focus on *bottom-up* attention.

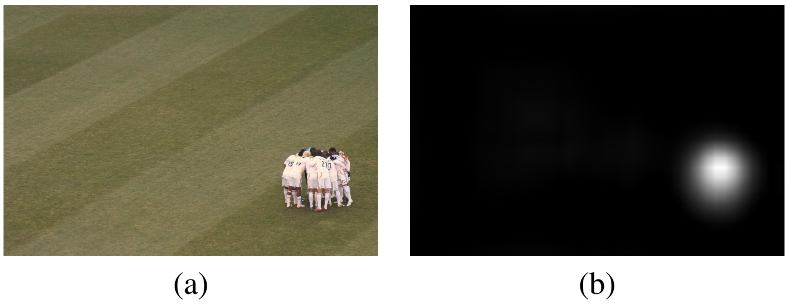

Figure 1. Example of visual saliency. b) is the salience map where brighter pixels represent regions more salient to humans on the original image a).

Visual salient regions on images are usually represented by *saliency maps*. Such maps are grayscale images generated such that areas with pixels close to white express high saliency of the same region on the original image, whereas regions with pixels close to black represent a region of low saliency. Datasets with pairs image-map are usually obtained by colleting eye-fixation data from humans that looked at the images.

1.2. 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* [Frintrop 2005] is a computational model that extracts features such as color contrast, orientation and luminance contrast from different scales of the image, features shown to be naturally salient to humans [Colombini 2014].

A rapid change of paradigm ocurred around 2015 when *Deep Learning* techniques showed to be extremely effective in the generation of saliency maps. Models such as *Salicon* [Jiang et al. 2015] showed that applying convolutional neural networks with weights initialized from networks used for image classification, e.g. *VGG-16* [Simonyan and Zisserman 2014] could yield maps very similar to those generated from humans. Later works further explored this idea. *ML-net* [Cornia et al. 2016] uses the output from different layers of *VGG-16* to use information from various dimensions and levels of abstraction. *DeepFix* [Kruthiventi et al. 2015] extends a pre-trained model with new layers that account for global features and center bias. *Salnet* [Pan et al. 2016] is a work that explores two models that are remarkably simple yet provides good results. As of today, at least nine out of the ten best models in the ranking of the *MIT saliency benchmark* [mit 2016], a benchmark that uses a variety of metrics to rank models, use *Deep Learning* techniques.

1.3. Motivation

Current state of the art models are in general quite expensive computationally, partly because most of them are based on very 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 network from scratch could yield a smaller amount of parameters that are more efficient for the sole task of salience prediction. Also, there are some ideas from previous work on psychology that weren't found to be used in current models but are considered worthwhile to be explored.

1.4. Objectives

This work aims at building a visual saliency model that is a) effective, yielding results similar to other state of the art models, and b) relatively simple and computationally efficient. It is important that both criteria are matched because we aim at extending the model in the future for video and real time applications such as navigating robots. We build a novice fully convolutional neural network. A previously built classical model, which is strongly based on *Vocus*, is used for comparison purposes.

2. Methodology

Implementations were made using *Theano* version `0.9.0.dev` along with the framework *Lasagne* version `0.2.dev1`. Experiments were conducted on a machine with *Ubuntu 16.04 LTS* and kernel *Linux* version `4.8.0-54-generic`. We used a GPU *NVIDIA GTX 1080* for training the model.

2.1. Proposed model

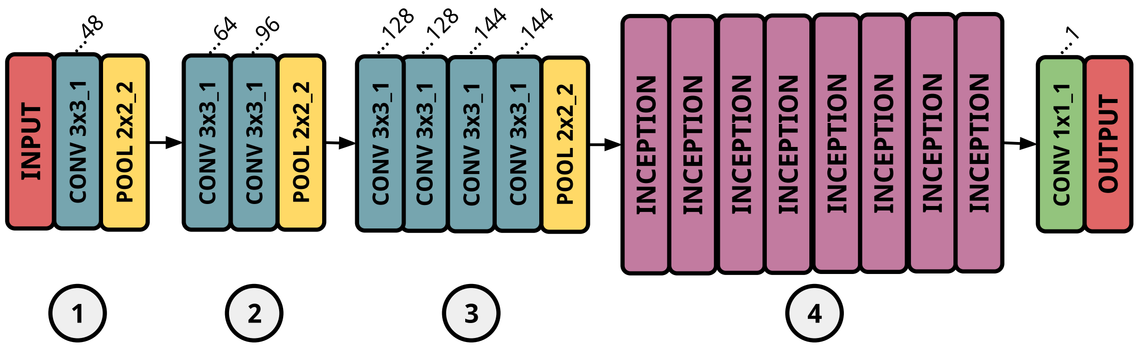

Figure 2. Overview of the network. Filters sizes are in format w$\times$h_stride.

Figure 2.1 illustrates the overall architecture of our fully convolutional neural network. The network extracts features from increasingly smaller dimensions of the input image. It is composed of four main blocks:

1. 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 image by a factor of two. It was found that 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. 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. Third level extracts medium-high level features from input with dimensions $W/4 \times H/4 \times 96$ using four convolution layers. The first two have 128 filters each, the last two have 144 filters each. Every convolution layer is followed by ReLu. Max-pooling is realized at the end. A considerable depth in this level was found to be important for the networks performance.

4. Fourth and last level is composed of eight inception blocks that extract high level features from the input with dimensions $W/8 \times H/8 \times 144$. 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 with dimensions $W/8 \times H/8 \times 1$.

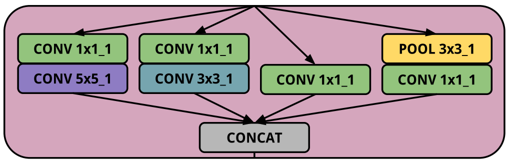

Figure 3. Inception block layout.

Figure 2.1 illustrates the inception architecture used in each block: We use filters of size 5×5, 3×3 (both preceeded by 1×1 convolutions in order to reduce number of input filters), 1×1 (which allows for carrying ahead of previous layers), and a max-pooling of size 3×3. Each of these operations is realized 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, which is a very low number compared to other models.

Table 1. Number of filters used at 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	58	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

2.2. Training

We resize images to dimensions $320 \times 240 \times 3$ during training. Each image is normalized channel-wise subtracting the channel mean and dividing by the standard deviation. We consider normalizing per image, rather than per dataset, to be more reasonable because visual saliency is highly connected to the context of the image. We also convert the images to the LAB colorspace. *Vocus* [Frintrop 2005] cites that the LAB colorspace is more closely related to human vision by encopassing red-green, yellow-blue and luminance maps. We conjecture that these maps facilitate extraction of important luminance and color contrasts by the learned convolution filters. Our prior tests showed slighly better performance using image-wise normalization and LAB instead of the commonly used RGB.

We aimed at maximizing (or minimizing the negative) of the *Correlation Coefficient* of the ground-truth saliency map G and the predicted map P: $CC(P, G) = cov(P, G)/(\sigma(P)\sigma(G))$. There is a variety of metrics for evaluating saliency predictions [Bylinskii et al. 2016], but we consider CC to be the most appropriate because it simmetrically penalizes both false positives and false negatives.

We considered two datasets for training: *SALICON* [sal 2016], with 15000 images, and *Judd* [Judd 2016], with 1003 images. The network was trained using Stochastic Gradient Descent with Nesterov Momentum of 0.9. We first used SALICON 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). We first used mean-std normalization of targets because it was noted that it allowed for a faster convergence. The network was trained for 5 epochs with learning rate of 0.009 and then for 3 epochs with learning rate of 0.001. We then switched to target normalized by unit normalization. The network was trained for 1 epoch with learning rate of 3×10^{-5} and L2 regularization of 10^{-4}. Finally, we used Judd with data augmentation by flipping images horizontally and the target normalized by unit normalization. The network was trained for 2 epochs with learning rate of 5×10^{-5} and L2 regularization of 3×10^{-5}. The whole training process took around two and a half hours.

3. Results

Table 2. State of the art models and results on MIT300 benchmark

Model	Num. parameters	AUC-Judd $\uparrow$	CC $\uparrow$	NSS $\uparrow$	Sim $\uparrow$	EMD $\downarrow$
Inf 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
Ours	**3.72 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

- Old method
- New method, mit300 bm

4. Conclusion

- Ey, that's pretty good
- next steps

References

(2016). Mit saliency benchmark.

(2016). Salicon dataset.

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

Colombini, E. L. (2014). *An Attentional Model for Intelligent Robotics Agents*. PhD thesis.

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

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

Frintrop, S. (2005). Vocus: a visual attention system for object detection and goal-directed search. In *IN LECTURE NOTES IN ARTIFICIAL INTELLIGENCE (LNAI)*. Springer.

Jiang, M., Huang, S., Duan, J., and Zhao, Q. (2015). Salicon: saliency in context. *CVPR*.

Judd, T. e. a. (2016). Learning to predict where people look.

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

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

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

Treisman, A. M. and Gelade, G. (1980). A feature-integration theory of attention. *Cognit Psychol*.