

Shallow and Deep Convolutional Networks for Saliency Prediction

Junting Pan , Elisa Sayrol and Xavier Giro-i-Nieto
Image Processing Group
Universitat Politecnica de Catalunya
Barcelona, Catalonia/Spain
xavi.er.gi.ro@upc.edu

Kevin McGuinness and Noel E. O'Connor
Insight Center for Data Analytics
Dublin City University
Dublin, Ireland
kevin.mcguinness@insight-centre.org

Abstract

The prediction of salient areas in images has been traditionally addressed with hand-crafted features based on neuroscience principles. This paper, however, addresses the problem with a completely data-driven approach by training a convolutional neural network (convnet). The learning process is formulated as a minimization of a loss function that measures the Euclidean distance of the predicted saliency map with the provided ground truth. The recent publication of large datasets of saliency prediction has provided enough data to train end-to-end architectures that are both fast and accurate. Two designs are proposed: a shallow convnet trained from scratch, and a another deeper solution whose first three layers are adapted from another network trained for classification. To the authors' knowledge, these are the first end-to-end CNNs trained and tested for the purpose of saliency prediction.

1. Introduction

This work presents two approaches for end-to-end convolutional neural networks (convnets or CNNs) for saliency prediction. Our objective is to compute saliency maps that represent the probability of visual attention on an image, defined as the eye gaze fixation points. This problem has been traditionally addressed with hand-crafted features inspired by neurology studies. In our case we have adopted a completely data-driven approach, using a large amount of annotated data for saliency prediction. Figure 1 provides an example of an image together with its ground truth saliency map and the two saliency maps predicted by the proposed convnets: a shallow one and a deep one.

Convnets are popular architectures in the field of deep learning and have been widely explored for visual pattern recognition, ranging from a global scale image classification [16] to a more local object detection [7] or semantic segmen-

Figure 1. Input Image (top left) and saliency maps from the ground truth (top right), our shallow convnet (bottom left) and our deep convnet (bottom right).

tation [21]. The hierarchy of layers in convnets are inspired by biological models, and some works have pointed to a relation between the activity of certain areas in the brain and the hierarchy of layers in the convnets [4]. Provided with enough training data, convnets have shown impressive results, often outperforming other hand-crafted methods. The rise of convnets originated with image classification [23] in the context of increasing availability of annotated data. Large datasets like ImageNet [5] or Places [33] have provided enough visual examples to train the millions of parameters that most popular convnets contain. These datasets provide thousands of images for each discrete label typically associated to a semantic class.

The saliency prediction problem, however, poses two specific challenges that differentiate it from classic image

Equal contribution.

classification. First, collecting large amount of training data is much more costly because it requires capturing the fixation points of human observers instead of a textual label for each image. Our work has benefited from recent publications of two large datasets containing images and an annotation of their salient points for humans [14]. Collecting this level of data has been possible thanks to crowdsourcing approaches, the same strategy used to annotate the ImageNet and Places datasets.

The second challenge to address when using convnets for saliency prediction is that a saliency score must be estimated for each pixel in the input image, instead of a global-scale label for the whole image. The saliency map at the output must present a spatial coherence and a smooth transition between neighbouring pixels.

The main contribution of this work is addressing the saliency prediction problem from an end-to-end perspective, by using convnets for regression rather than classification. We apply this strategy with two different architectures trained with two different approaches: a shallow convnet trained from scratch, and a deep convnet that reuses parameters from the bottom three layer of a network previously trained for classification. To the authors' knowledge, these were the first convnets that formulate saliency prediction as an end-to-end regression problem.

This paper is structured as follows. Section 2 presents the previous and recent works using convolutional networks for saliency prediction and detection. Section 3 introduces the shallow convnet, while Section 4 presents the deep network. Section 5 compares both networks in terms of memory requirements. It also shows, prediction performance in the MIT Saliency Benchmark and LSUN Saliency Prediction Challenge 2015 and they are compared with other models. Conclusions and future directions are outlined in Section 6.

Our results can be reproduced with the source code and trained models available at <https://github.com/imatge-upc/saliency-2016-cvpr>.

2. Related work

The proposed networks presents the next natural step to two main trends in deep learning: using convolutional neural networks for saliency prediction and training these networks by formulating saliency prediction as an end-to-end regression problem. This section reviews related work in these directions.

An early attempt of predicting saliency with a convnet was the *ensembles of Deep Networks (eDN)* [27], which proposed an optimal blend of feature maps from three different convnet layers, that were finally combined with a simple linear classifier trained with positive (salient) or negative (non-salient) local regions. This approach inspired *DeepGaze* [17] to adopt a deeper network. In particular, *DeepGaze* used the existing AlexNet network [16], where

the fully connected layers were removed to keep the feature maps from the convolutional layers. The response of each layer were fed into a linear model and its weights learned. *DeepGaze* would be the first case of transfer learning from a convnet for classification used for saliency, as we propose in our deeper architecture. However, we do not train a linear model to combine feature maps but directly train a stack of new convolutional layers on top of the transferred ones. Other recent works have explored the combination of different convnets working at different resolutions to capture both global and local saliency. Liu *et al.* [20] proposed an architecture with three convnets working in parallel where the three final fully connected layers are combined in a single layer to obtain the saliency map. Unlike our work the network is trained with image regions centered on fixation and non-fixation eye locations.

On the other hand, *DeepFix* model (unpublished) captures information at different scales by using very deep networks, inspired by the VGG network architecture proposed by Simonyan and Zisserman [26].

Other approaches introduce new architectures and improvements in salient object detection. Zhao *et al.* [32] use also two parallel networks to obtain local and global context modeling. The input image consists of a superpixel-centered window that is preprocessed differently to feed each of the two convnets. Fully connected layers are combined at the end to obtain the salient objects. The work by Li and Yu [18] proposes three nested windows as inputs to three different convnet at different scales that are fused together to obtain an aggregated saliency map. Wang *et al.* proposed a different pipeline [28]: local estimation is carried out and the resulting information is used as input to obtain a global search. That is, first, to detect local saliency, a deep neural network (DNN-L) learns local patch features to determine the saliency value of each pixel. Second, the local saliency map together with global contrast and geometric information are used as global features to obtain object candidate regions. A deep neural network (DNN-G) is then trained to predict the saliency score of each object region based on global features.

Fully Convolutional Networks (FCNs) [21] addressed the semantic segmentation task to predict the semantic label of every individual pixel in the image. This approach dramatically improved previous results on the challenging PASCAL VOC segmentation benchmark [6].

Finally, the SALICON model [11] uses the same saliency evaluation metrics as loss function. The proposed architecture is very similar to our Deep Convnet, but in their work multiple scales are incorporated to consider selective attention at different resolutions.

In our work we are interested in finding saliency maps rather than salient object detection by training convnets end-to-end. We also focus on novel databases that are annotated for the purpose of saliency prediction.

Figure 2. Architecture of the shallow convolutional network.

3. Shallow Convnet

This section presents the first of our proposed convnets, which is based on a lightweight architecture whose parameters are trained from scratch.

3.1. Architecture

The network consists of five layers with learned weights: three convolutional layers and two fully connected layers. Each of the three convolutional layers is followed by a rectified linear unit non-linearity (ReLU) and a max pooling layers. Figure 2 shows a detailed description of each layer. The network has to a total of 64.4 million free parameters.

The network was designed considering the amount of available saliency maps for training it from scratch. Different strategies were considered to avoid overfitting the model. First, we used three convolutional layers rather than the five used in the classic AlexNet architecture [16] (and far less than very deep networks used recently such as the thirteen used in VGG-16 [26]). Second, the input images are resized to $[96 \times 96]$, a much smaller dimension than the $[227 \times 227]$ used in AlexNet [16]. The three max pooling layers reduce the initial $[96 \times 96]$ feature maps down to $[10 \times 10]$ by the last of the three poolings.

Even with the above constraints, the network still overfits significantly. We found that norm constraint regularization for the maxout layers [8], which computes the *max* between pairs of the previous layers output, was essential to mitigate against this overfitting. We also tested using dropout after the first fully connected layer, with a dropout ratio of 0.5 (50% of probability to set a neuron’s output value to zero), but this did not improve overfitting much, and so was not included in the final model.

Notice that the 2,304-dimensional vector at the output is mapped into a 2D array of $[48 \times 48]$, which corresponds to the saliency map. This decrease in resolution is compensated at test time by resizing the dimensions of the output to match the input image and posterior filtering using a Gaussian

kernel with a standard deviation of 2.0.

This shallow convnet was implemented using Python, NumPy, and the deep learning library Theano [1]. Processing was performed on an NVIDIA GTX 980 GPU with 2048 CUDA cores and 4GB of RAM. It took between 6 and 7 hours to train for the SALICON dataset, and 5 to 6 hours for the iSUN dataset. Saliency prediction requires 200 ms per image.

3.2. Training

This shallow network was trained from scratch twice, each time from a different dataset. A first model was built using the 10,000 saliency maps from the SALICON dataset [14], and a second model using the 6,000 saliency maps from the iSUN dataset. Both datasets are described in detail in Section 5.2. Given the smaller amount of images available in the iSUN dataset, a slight modification was introduced in this second model: the depth of the third convolutional network was of 64 instead of 128, as depicted in Figure 2.

The weights in all layers were initialized from a normal Gaussian distribution with zero mean and a standard deviation of 0.01, with biases initialized to 0.1. The network was trained with stochastic gradient descent (SGD) and the Nesterov momentum method, which we found helps convergence. The learning rate changed over time, starting with a higher learning rate 0.03 and decreased during training to 0.0001. We trained the network for 1,000 epochs. For validation purposes, we split the training data into 80% for training and the rest for periodic validation. A data augmentation technique was used by mirroring all images. All considered saliency maps were normalized to $[0, 1]$.

We used regularized L2 (Euclidean) loss for this network as well the deep one presented in Section 4, with the standard L2 norm regularizer on the weights. We experimented with several other loss functions while developing our algorithm (including L1 loss and sigmoid cross entropy loss), but found that these often resulted in vanishing gradients, significantly slowing convergence. We considered designing a loss function that approximated one of the evaluation metrics directly. Unfortunately, many of these metrics are complex, and difficult to approximate with an easily differentiable function.

The filters learned in the first convolutional layer are shown in Figure 3. They present a similar pattern to other similar filters learned for classification convnets [30, 30], where edge detectors can be identified. It is noticeable how these type of filters arise also when training our network on saliency maps.

4. Deep Convnet

The second approach explored in this paper is the adaptation of an existing very deep convnet trained for image classification for the task of saliency prediction. Previous

Figure 3. Filters learned for the first convolutional layer of the shallow convnet (best viewed from a distance).

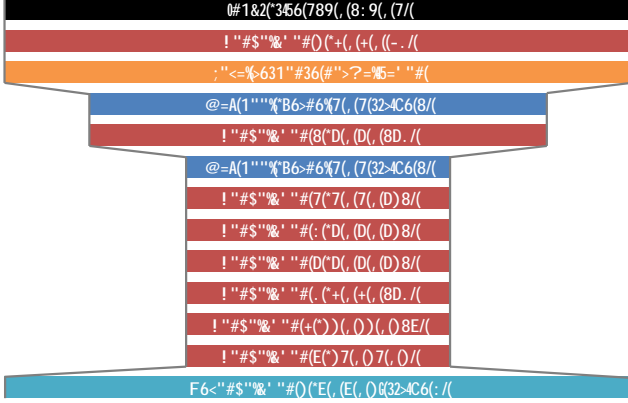


Figure 4. Architecture of the deep convolutional network

work [30] has noted how, in image classification tasks, the model parameters from the lowest levels in the convnets converge in a few epochs. This observation, together with visualization of the filters learned at these layers [25], suggest that these layers perform low-level visual task in vision, such as the detection of colors or textures. Our hypothesis is that these lower layers trained for classification can also be transferred for the task of saliency prediction. We propose a second convnet which adapts these pre-trained filters and combines them with new layers specifically trained for saliency.

4.1. Architecture

Figure 4 illustrates the layer architecture of the network, composed of 10 weight layers and a total of 25.8 million parameters. The architecture of the first 3 weight layers is compatible with that of the VGG network from [3]. Each convolutional layer is followed by a rectified linear unit non-linearity (ReLU). Pooling layers follow the first two convolutional layers, effectively reducing the width and height of the feature maps in the intermediate layers by a factor of four. A deconvolution layer follows the final convolution to produce a saliency map that matches the input width and height.

To choose the final network architectures, we experimented with many different different variants, testing each on a held-out validation set of 1,000 images. In general we found that: 1) adding more layers improves accuracy; 2) adding more feature maps per layer usually improves accuracy too; and 3) using dropout regularization did not

significantly improve accuracy but did increase training time. The final network design was primarily constrained in resolution, number of layers, and layer depth by the amount of available GPU memory.

We used transfer learning to initialize the weights for the first three convolutional layers with the pre-trained weights from the *VGG_CNN_M* network from [3]. This acts as a regularizer and improves the final network result. The remaining weights were initialized randomly using the strategy from [10].

4.2. Training

We trained our network on 9,000 of the 10,000 training images in the SALICON dataset, setting aside 1,000 images for validation (ground truth for the SALICON validation set had not yet been released when this network was first trained). We used several standard pre-processing techniques on both the input images and the target saliency maps. We subtracted the mean pixel value of the training set from the image pixels to zero center them and rescaled the resulting values linearly to be in the interval $[-1, 1]$. We similarly preprocessed the saliency maps by subtracting the mean and scaling to $[-1, 1]$. Both the input images and the saliency maps were downsampled by half to 320×240 prior to training.

The network was trained using stochastic gradient descent with Euclidean loss using a batch size of 2 images for 24,000 iterations. During training, the network was validated against the validation set after every 100 iterations to monitor convergence and overfitting. We used the standard L^2 weight regularizer (weight decay), and halved the learning rate every 100 iterations. The network took approximately 15 hours to train on a NVIDIA GTX Titan GPU running the Caffe framework [13]. We normalized the base learning rate by the number of predictions per image, to give a learning rate of $0.01/(320 \times 240) = 1.3 \times 10^{-7}$. Using a larger learning rate causes the learning to diverge.

The network was trained on inputs of size 320×240 , but in principle, it can handle images of any size, since it only consists of convolutional and pooling layers. In practice, the input size is constrained by the amount of GPU memory (or RAM) needed to store the outputs of the intermediary layers. Nevertheless, the network has the advantage that it can be sized to match the aspect ratio of any image, and indeed use this approach for the images in the MIT300 benchmark in the next section.

5. Experiments

5.1. Memory requirements

The architectures of the two networks present different requirements in terms of memory resources. These resources are dedicated to two different tasks: the parameters that de-

	Shallow	Deep
Data	2.29 MB	123.65 MB
Parameters	244.64 MB	98.44 MB
Total (train)	249.22 MB	345.74 MB
Total (test)	246.93 MB	222.09 MB

Table 1. Approximate memory requirements for each convnet.

fine the network, and the blob data that characterizes network response at the different processing stages.

The parameters that define the network are fit during training, and, together with the architecture layout, correspond to the actual characterization of the network. These parameters characterize the output of each neuron in the net, which can be defined as $f(w^T x + b)$, where w describes the filter parameters in the convolutional layers, b corresponds to the biases and f is the non-linearity. Each neuron, therefore, has parameters w and b , which are fit during backpropagation.

The data associated to the input image is the second source of memory requirements. The input image is hierarchically processed in the convnet, creating multiple intermediate feature maps (or data blobs) after each processing stage.

Table 1 presents the complementary memory requirements for each of the two convnets. These values have been obtained from the architectures of the shallow and very deep networks described in Figures 2 and 4, respectively. The estimation assumes 32-bit floating points to store parameters and layer output (4 bytes per value). The memory estimate for blob data assumes test time (forward pass only): at train time this value is doubled to account for the error signal during backpropagation.

The number of parameters for both networks are much lower than the very deep networks used in classification. For example, the 19 layers version of VGG net requires 144 million parameters [26].

Our shallow network requires far less memory for the layer outputs, but has significantly more parameters (due to the fully connected layers). This explains why our deep network does not overfit, whereas stronger regularization is necessary to fit the shallow one. Since the shallow network needs less memory for the layer outputs, it is possible to make batch size on this network very large at test time, allowing it to process many more images at once.

5.2. Datasets

We used three datasets, presented in Table 2:

SALICON [14]: This is the largest dataset available for saliency prediction and was used to train our models. It was built from images of the *Microsoft CoCo: Common Objects in Context* [19] dataset, which inspired the SALICON naming: *SALiency in CONtext*. However, the saliency maps in

SALICON were not collected with eyetrackers as in most popular datasets for saliency prediction, but with mouse clicks captured in a crowdsourcing campaign. **iSUN** The iSUN dataset has been built with an online game using webcams to track player eye gaze. The dataset uses natural scene images from the SUN database [29], a large dataset organized in 397 scene categories. **MIT1003 and MIT300 [15]** This dataset is the most well-known among saliency prediction researchers. It is accompanied by an online benchmark maintained by its authors. The MIT1003 dataset consists of both images and fixation points that can be used for training. The fixation points for the MIT300 dataset are not public: the dataset can only be used for benchmarking. The stimuli images in these datasets consist of indoor and outdoor natural scenes from the Flickr Creative Commons and LabelMe [24] datasets.

Our two models were tested on a total of 7,300 images coming from three different datasets. The dataset sizes and diversity in terms of observers and nature (eye gaze and clicks) provide a higher statistical significance than previous works [18, 20, 28]. Although larger in size, SALICON and iSUN datasets are more exposed to quality degradation because they were built via crowdsourcing. On the other hand, MIT1003 and MIT300 are considered cleaner because fixations points were captured in a controlled environment.

5.3. Results

The evaluation of saliency prediction has received the attention of several researchers, resulting in various proposed approaches. Our experiments consider several of these, in a similar way to the MIT saliency benchmark [2]. Some of these metrics compare the predicted saliency maps with the maps generated from the fixation points of the ground truth, while some other metrics directly compare with the fixation points. In the result tables that follow, we have sorted the different techniques based on the AUC Judd metric.

Where not otherwise stated, our convnets were trained with images from the SALICON [14] dataset and tested on images from iSUN and MIT300 datasets to avoid overfitting. The one exception to this is our submission for the LSUN 2015 challenge, where our shallow network was trained with training and validation data from iSUN, and assessed on the test partition.

Figure 5 presents a qualitative comparison of the two networks, showing the predicted saliency maps alongside the ground truth fixation maps. These examples show a different behaviour between the two networks, with the shallow one presenting a bias towards the central part of the image. The deep network, on the other hand, offers a higher spatial resolution thanks to its architecture with larger feature maps.

Our shallow convnet was the winner of the 2015 LSUN saliency prediction challenge. This challenge required participants to evaluate their algorithms on the test partitions

Dataset	Description	Capture device	Observers	Train	Validation	Test
SALICON [14]	Microsoft CoCo [19]	Mouse clicks	Crowd	10,000	5,000	5,000
iSUN	SUN [29]	Eyetracker	Crowd	6,000	926	2,000
MIT300 [2]	Flickr and LabelMe [24]	Eyetracker	39	-	-	300

Table 2. Description of the three datasets used in our experiments.

Figure 5. Saliency maps generated by our shallow and deep network on the SALICON and iSUN validation data.

of the iSUN and the SALICON datasets. Our network was trained only with images from the training and validation partitions of each dataset separately, so images from different datasets were never mixed for these experiments. Table 3 and Table 4 include the results provided by the organizers of the challenge for the iSUN and SALICON datasets. The scores obtained for every measure considered demonstrate the superior performance of our shallow network compared with the other participants.

The presented shallow and deep convnets are compared quantitatively in Tables 5 and 6. The deep convnet usually outperforms shallow in all cases but on iSUN validation. We hypothesize this better results are because: 1) it retains the aspect ratio of the input image (there are no fully connected layers), and 2) that it produces a higher resolution output, which can often better match more complex patterns in the

saliency map.

Table 6 also compares our results with some other top performers in the MIT300 benchmark. Our deep convnet achieves similar results to the ones obtained by Deep Gaze 1 [17] at the upper part of the table. The shallow convnet performs worse but still in the upper part of a table which, in its full version, compares 47 different models. SALICON [11] and DeepFix obtain better scores than our deep convnet in MIT300, but their complexity is also higher. DeepFix requires 22 layers and SALICON uses 16 layers and multiple scales of the image, while our simpler models are defined with 10 (deep) or 5 (shallow) layers on a single scale.

Results also indicate a robustness of our models across datasets. A detailed analysis of Table 6 suggests a robust behaviour of our convnets across datasets. Our networks were trained purely on SALICON data but still obtained top

	Similarity	CC	AUC shuffled	AUC Borji	AUC Judd
Shallow Convnet (iSUN)	0.6833	0.8230	0.6650	0.8463	0.8693
Xidian	0.5713	0.6167	0.6484	0.7949	0.8207
WHU IIP	0.5593	0.6263	0.6307	0.7960	0.8197
LCYLab	0.5474	0.5699	0.6259	0.7921	0.8133
Rare 2012 Improved [22]	0.5199	0.5199	0.6283	0.7582	0.7846
Baseline: BMS [31]	0.5026	0.3465	0.5885	0.6560	0.6914
Baseline: GBVS [9]	0.4798	0.5087	0.6208	0.7913	0.8115
Baseline: Itti [12]	0.4251	0.3728	0.6024	0.7262	0.7489

Table 3. Results for the iSUN test set, according to the LSUN Challenge 2015.

	Similarity	CC	AUC shuffled	AUC Borji	AUC Judd
Shallow Convnet	0.5198	0.5957	0.6698	0.8291	0.8364
WHU IIP	0.4908	0.4569	0.6064	0.7759	0.7923
Rare 2012 Improved [22]	0.5017	0.5108	0.6644	0.8047	0.8148
Xidian	0.4617	0.4811	0.6809	0.7990	0.8051
Baseline: BMS [31]	0.4542	0.4268	0.6935	0.7699	0.7899
Baseline: GBVS [9]	0.4460	0.4212	0.6303	0.7816	0.7899
Baseline: Itti [12]	0.3777	0.2046	0.6101	0.6603	0.6669

Table 4. Results for the SALICON test set, according to the LSUN Challenge 2015.

iSUN (validation)	CC	AUC Shuffled	AUC Borji
Shallow Convnet	0.59	0.64	0.79
Deep Convnet	0.53	0.63	0.80
SALICON (val.)			
Shallow Convnet	0.58	0.67	0.83
Deep Convnet	0.61	0.73	0.86
SALICON (test)			
Shallow Convnet	0.60	0.67	0.83
Deep Convnet	0.62	0.72	0.86

Table 5. Comparison of our shallow and deep convnets.

performing results on the MIT benchmark when compared with other models trained on MIT1003 Deep Gaze 1 [17], eDN [27], and Judd [15]).

6. Conclusions

We propose a novel end-to-end approach for training convnets in the task of saliency prediction. The excellent results of both architectures in state-of-the-art benchmarks demonstrate the superior performance of our convnets with respect to hand-crafted solutions and highlight the importance of an end-to-end formulation of saliency prediction.

The comparison between our shallow and deep networks

trained on SALICON data has provided similar results for the iSUN dataset, but a better result for the deep network on MIT300. On the other hand, the shallow network requires less memory at train time and generates saliency maps much faster because it has fewer layers. Both networks rank highly in the MIT300 benchmark despite not being trained on this dataset. This clearly demonstrates the generalization performance of the networks and robustness to dataset biases.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under grant number SFI/12/RC/2289, project Big-Graph TEC2013-43935-R, funded by the Spanish Ministerio de Economía y Competitividad and the European Regional Development Fund (ERDF), and SGR14 Consolidated Research Group sponsored by the Catalan Government (Generalitat de Catalunya) through its AGAUR office. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GeForce GTX Titan Z used in this work.

References

- [1] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a cpu and gpu math expression compiler. In *Conference on Python for Scientific Computing*, volume 4, page 3, 2010.

	Similarity	CC	AUC shuffled	AUC Borji	AUC Judd
Baseline: Infinite Humans	1.00	1.00	0.80	0.87	0.91
SALICON [11]	0.60	0.74	0.74	0.85	0.87
DeepFix	0.67	0.78	0.71	0.80	0.87
Deep Gaze 1 [17]	0.39	0.48	0.66	0.83	0.84
Deep Convnet	0.52	0.58	0.69	0.82	0.83
BMS [31]	0.51	0.55	0.65	0.82	0.83
eDN [27]	0.41	0.45	0.62	0.81	0.82
GBVS [9]	0.48	0.48	0.63	0.80	0.81
Judd [15]	0.42	0.47	0.60	0.80	0.81
Shallow Convnet	0.46	0.53	0.64	0.78	0.80
Mr-CNN [20]	0.48	0.48	0.69	0.75	0.79
Rare 2012 Improved [22]	0.46	0.42	0.67	0.75	0.77
Baseline: One human	0.38 – 0.46	0.52 – 0.65	0.63 – 0.67	0.66 – 0.71	0.80 – 0.83

Table 6. Results of the MIT300 dataset.

- [2] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. Mit saliency benchmark. <http://saliency.mit.edu/>.
- [3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.
- [4] R. Cichy, A. Khosla, D. Pantazis, A. Torralba, and A. Oliva. Mapping human visual representations in space and time by neural networks. *Journal of vision*, 15(12):376–376, 2015.
- [5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.
- [6] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2014.
- [7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.
- [8] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1319–1327, 2013.
- [9] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems*, pages 545–552, 2006.
- [10] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.
- [11] X. Huang, C. Shen, X. Boix, and Q. Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 262–270, 2015.
- [12] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (11):1254–1259, 1998.
- [13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, volume 2, page 4, 2014.
- [14] M. Jiang, S. Huang, J. Duan, and Q. Zhao. SALICON: Saliency in context. In *IEEE conference on Computer Vision and Pattern Recognition*, 2015.
- [15] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *IEEE International conference on Computer Vision*, pages 2106–2113, 2009.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [17] M. Kümmerer, L. Theis, and M. Bethge. Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet. In *International Conference on Learning Representations*, 2015.
- [18] G. Li and Y. Yu. Visual saliency based on multiscale deep features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5455–5463, 2015.
- [19] 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 *Computer Vision–ECCV*, pages 740–755, 2014.
- [20] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu. Predicting eye fixations using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 362–370, 2015.
- [21] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.
- [22] N. Riche, M. Mancas, M. Duvinage, M. Mibulumukini, B. Gosselin, and T. Dutoit. Rare2012: A multi-scale rarity-based saliency detection with its comparative statistical anal-

ysis. *Signal Processing: Image Communication*, 28(6):642–658, 2013.

- [23] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2121–2131, 2015.
- [24] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173, 2008.
- [25] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *International Conference on Learning Representations*, 2014.
- [26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- [27] E. Vig, M. Dorr, and D. Cox. Large-scale optimization of hierarchical features for saliency prediction in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2798–2805, 2014.
- [28] L. Wang, H. Lu, R. Xiang, and M.-H. Yang. Deep networks for saliency detection via local estimation and global search. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3183–3192, 2015.
- [29] J. Xiao, J. Hays, K. Ehinger, A. Oliva, A. Torralba, et al. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, 2010.
- [30] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV*, pages 818–833, 2014.
- [31] J. Zhang and S. Sclaroff. Saliency detection: A boolean map approach. In *IEEE International Conference on Computer Vision*, pages 153–160, 2013.
- [32] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1265–1274, 2015.
- [33] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems*, pages 487–495, 2014.