

RecSal : Deep Recursive Supervision for Visual Saliency Prediction

Sandeep Mishra*

Indian Institute of Technology
Kharagpur, India

Oindrila Saha*

Abstract

State-of-the-art saliency prediction methods develop upon model architectures or loss functions; while training to generate one target saliency map. However, publicly available saliency prediction datasets can be utilized to create more information for each stimulus than just a final aggregate saliency map. This information when utilized in a biologically inspired fashion can contribute in better prediction performance without the use of models with huge number of parameters. In this light, we propose to extract and use the statistics of (a) region specific saliency and (b) temporal order of fixations, to provide additional context to our network. We show that extra supervision using spatially or temporally sequenced fixations results in achieving better performance in saliency prediction. Further, we also design novel architectures for utilizing this extra information and show that it achieves superior performance over a base model which is devoid of extra supervision. We show that our best method outperforms previous state-of-the-art methods with 50-80% fewer parameters. We also show that our models perform consistently well across all evaluation metrics unlike prior methods.

1 Introduction

Visual saliency is the probability of spatial locations in an image to attract human attention. Given an image, mimicking human saliency patterns is a key to solve many vision problems. To enable this, saliency prediction models must be presented with data in a similar manner as to humans. Essentially, when presented with an image a human subject will look at locations one-by-one with their next fixation depending on what they have already seen. Further, when a person looks at a particular region there will be a pattern as to how that region is scanned i.e. what are the most interesting points in that region. Thus the information of temporal sequence as well as region-specific saliency patterns is important to be recognized by a model to predict a saliency map imitating human fixation probabilities.

Data-driven approaches to saliency prediction depend upon the ground truth aggregate saliency map to train deep CNN models end-to-end. However, such a map is a crude average over observers, time and spatial regions. The saliency of an image can be very different

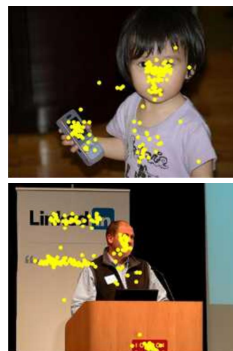


Figure 1: Some example OSIE [10] images with fixation points in yellow

across regions and time. Once an observer is presented with an image, they can start with any point in the image, then move on to fixate at any another location. Now the next location that a person fixates on will be dependent on the location he has already scanned. So a fixation at a given time is dependent on all the fixations of the observer before that [15]. This path of all fixations arranged wrt time is known as a scanpath [24]. The aggregate saliency map does not contain this temporal scanpath information. But this information is crucial in predicting saliency due to the successive dependency.

Since typical saliency prediction models predict a single map, fixation points in all spatial regions of the image are treated in the same way. Let us contrast this with a segmentation network where each object is assigned a different class channel in the final model output, resulting in the model being able to learn to treat separate regions differently. But in saliency prediction, irrespective of regions or objects, the ground truth is just a single map. It should also be noted that the fixation pattern varies wrt the types of objects in a region. Humans look at different regions in the image in different manner. For example, in the images in Figure 1 a face tends to have fixations where the features lie, like eyes, nose and mouth. The phone in the child’s hand has fixations on the screen, while uniform spaces like the background of the man in the bottom image has fixations on the texts. Treating these areas differently and separately learning them can allow the network to learn local saliency patterns.

Our prime contributions are as follows:

- We propose a multi-decoder network to exploit the contributions of features obtained from shallow and deep layers of the encoder to form the final saliency map.
- We design our model to predict multiple saliency maps each of which is trained on a separate loss so as to enable one model to do well in all evaluation metrics.
- We propose the use of temporally and spatially sequenced metadata to provide bio-inspired deep supervision to our model. To the best of our knowledge, this is the first method to use such extra data for supervision in a saliency prediction task.
- We further propose novel recursive model architectures to effectively use this metadata for sequential supervision, and finally show superior results in predicting the aggregate saliency maps, as compared to our baselines and previous state-of-the-art methods.

2 Related Work

Visual Saliency: Since the classical methods [8, 16, 33] which utilized hand-crafted features, saliency prediction models have come much closer to mimicking humans with the help of deep CNNs. Advances in model architectures have shown to obtain better performance. SalGAN [26] uses an added adversarial network for loss propagation. Dodge *et al.* [9] uses two parallel networks to fuse feature maps before prediction. We propose a novel base architecture with multiple decoders which utilize features extracted from coarse to finer levels for predicting saliency. Performance is measured using varied metrics and it has been shown that optimizing a model on a single one of these will not give good performance on the other metrics. Kummerer *et al.* [21] states that a probabilistic map output which can be post-processed for various metrics tackles this problem. Some methods incorporate more than one loss to train the final saliency output [3, 17]. We design our model to produce multiple output saliency maps each optimized on a different metric.

Recursive Feature Extraction: Some methods [3, 34] use a recurrent module for refining the final saliency map recursively. Fosco *et al.* [7] uses multi duration data and uses a variant

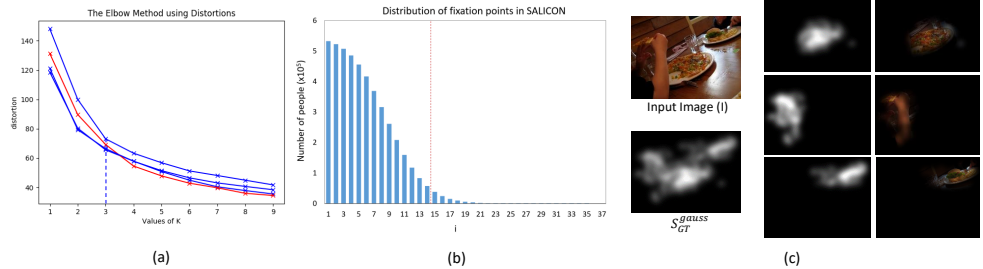


Figure 2: (a) Plot of Distortion vs number of clusters (K) where the usual elbow point is at 3. The red line highlights an exception where the elbow point is 4. (b) Plot of number of people who have at least ' i ' fixations vs ' i ' for SALICON train set. (c) For given input image and ground truth S_{GT}^{Gauss} , the region based separated maps ordered wrt the number of fixation points present in each cluster. The 3 salient regions visible: pizza, hand and burger are separated by our algorithm. The rightmost column is an overlay of the three maps on the image, which shows that the pizza is of highest interest, followed by the hand and then the burger.

of LSTM [10] based attention module to compute outputs denoting where people look for a given time duration after being shown the image. Jiang *et al.* [18] uses a Convolutional LSTM [55] based method for video saliency prediction. We use a recursive block (RB) to provide extra auxiliary supervision using sequential data to finally improve saliency prediction performance.

Additional Context Using Extra Data: Approaches like [14, 22] perform multi-task learning to solve both segmentation and saliency prediction, and show how the added context affects each task. Some methods [6, 53] use information from multiple datasets to solve a unified task – segmentation in this case. Zhao *et al.* [57] provides local and global context using the same input image with the help of two independent networks. Ramanishka *et al.* [22] uses caption generation in videos for performing more accurate saliency prediction. As opposed to these methods, we do not use any extra dataset or annotations for auxiliary supervision. We describe how we extract this data from eye gaze annotations in Section 3. We use temporally and spatially sequenced data for a single static image to provide extra context to the network to enhance performance.

3 Extra Supervision

We use SALICON-2017 [12] and MIT1003 [19] datasets for training our models. Rather than using extra annotations and enforcing the network to learn how to solve a divergent task, such as semantic segmentation [14], we propose to use data that is directly derived from eye gaze as that would provide more task specific context to the network. We create this meta-data using information already available in the eye gaze annotations of the above datasets to add deep supervision to our network. We separate and order aggregate fixation points based on spatial region-specific importance and temporal sequence as explained below.

3.1 Temporal Data

Both the SALICON and MIT1003 provide eye gaze data which contains fixation points in the order of the occurrence of those fixations, hence we choose to provide this additional temporal information to the network. This will enable the network to extract information

about the image as humans do, i.e. first on a coarser level and then go on to perceive the finer details as the image is seen for more time.

MIT1003 uses the second to sixth fixation point of all users to generate the final fixation map per image. They choose to ignore the first fixation for each user to avoid an extra point because of the initial centre bias. We separate these points based on order of occurrence wherein for a particular image we create five temporally sequenced fixation maps. Each map contains the i^{th} fixation point for all viewers who saw that image. However, SALICON uses all the fixation points for each user to form the final saliency map. On careful examination of the data we find that there were images which had as less as zero or one fixation and as high as 35 fixations per user. Therefore, we plot a histogram of the number of people having at least i fixations (Figure 2 (b)). We observe that the histogram follows an approximately gaussian distribution with the peak being at 1. We find the standard deviation (σ) of our curve to be 6.7 and using the '68-95-99.7' / '3-sigma' rule we choose the $\mu + 2\sigma$ point, before which 95% of the information lies, to be our number of temporally sequenced fixation maps. Thus the number of maps become fourteen, which we make as described for MIT1003 and put the rest all remaining fixations in one last map. So our total maps and thus timesteps here become fifteen. Note that we ignore the first fixation point of every user for MIT1003 but keep them for SALICON so that the aggregate of temporally sequenced fixation maps align with the ground truth fixation map provided in the datasets.

3.1.1 Temporal Order vs Duration

The temporally sequenced fixation maps generated above can be used in two ways. One is to simply arrange these maps in order of occurrence as the output target for each time-step of our recurrent module. The second way is to modify these maps as $m_t = m_{t-1} + m_{t-2} + \dots + m_0$. This means that the t^{th} map consists of all the locations that have been looked at till the t^{th} time-step i.e. in a given **duration** from 0 to t . We denote the maps arranged in temporal order as non-incremental data and the duration-wise arranged maps as incremental data.

3.2 Spatial Data

Close observation of the gaze data suggests that people tend to focus more on certain regions than just hovering through the whole image which tells us that there are certain spatial locations which are regions of interest and hence attracting viewers' gaze. Thus, given an image stimulus we enable the model to identify which regions are more salient and sequence them in order of importance to help form the final saliency map better. This is just the mapping of the relative importance of these regions in a single prediction.

Given the aggregate fixation points of all users across time, we use Gaussian Mixture Models [29] over the 2D point maps to create clusters. We use the elbow method on the WSS (within-cluster sum of square) vs number of clusters plot for each map of both SALICON and MIT1003 and find that the optimal number of clusters is three (Figure 2 (a)). Thus, we divide the fixation points into five sets and order them with respect to the total points in each set, more points in a set denoting spatial regions of higher interest.

Note that the same technique as described in Section 3.1.1, can be performed for the spatially sequenced saliency maps also. We use the terminology of **incremental** vs **non-incremental** similarly as temporal data for spatial data as well. Note that we will denote the incremental metadata maps as $M^I : \{m_0^I, m_1^I, \dots, m_{T-1}^I\}$ (as the last map m_T^I is same as the final saliency map S); and the non-incremental maps as $M^{NI} : \{m_0^{NI}, m_1^{NI}, \dots, m_T^{NI}\}$ from now onwards. We compare the effect of above methods of deep supervision in Section 6.

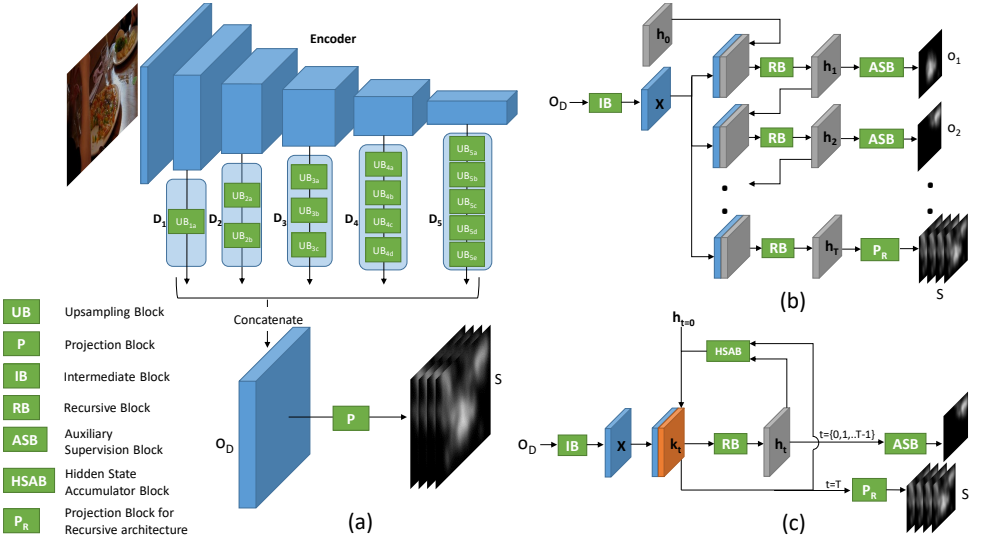


Figure 3: (a) Base encoder-multi decoder architecture, (b) Recursive Module for Incremental Data (RB), (c) Recursive Module for Non-Incremental Data built upon RB with the addition of HSAB module

4 Model Architecture

4.1 Base Architecture

We design a deep convolutional encoder decoder network for our base architecture. As described in Itti *et al.* [14], features like color, intensity and textures contribute in determining saliency. Hence we choose to use feature maps from the shallowest to the deepest levels of the encoder for final saliency prediction. We do so using multiple decoders which consume feature inputs from the encoder just before every downsampling operation. The first decoder is placed just before the second downsampler and we use a total of five decoder blocks which gives us the advantage of using features from five different scales.

For the decoders we use stacks of up-sampling blocks, which are bilinear upsampling operations followed by a convolution + batch norm [13] + ReLU [10] layer, such that final outputs of each decoder has same spatial size. Finally the outputs of all decoders are concatenated to form O_D (where $D : [D_1, D_2, D_3, D_4, D_5]$) which is passed through the projection convolution block: P to form the final output multiple saliency maps as shown in Figure 3 (a). Note that each output map optimises on a different loss as explained in Section 5.

4.2 Recurrent Module for Incremental Data

We need to modify the above base architecture for effectively using the metadata (M) described in Section 3. We first consider using the incremental metadata. Since this extra data can directly be used to construct the final saliency ground truth map without extensive operations, one option is to have an intermediate activation map: X from the projection convolution block (P) trained to predict these generated maps m_t via an auxiliary loss: $L_{aux}(X, M)$ while still retaining the output of P to be S (the final aggregate saliency maps). But such an architecture will not be able to exploit the relation between each m_t effectively, as the channel order does not affect the learning of the model. Therefore, to use the sequential data

meaningfully, we shall have to predict each m_t successively. But X itself is not self-sufficient to create a sequential mapping between each m_t . As a result we will need a recursive formulation where each m_t will be predicted with the help of X and a hidden state h_{t-1} which encodes all information till $t - 1$. Using an initial state of no information (h_0) and X , we can create the first hidden state h_1 which subsequently when passed through an Auxiliary Supervision Block: ASB will generate o_1 (see Figure 3). Now this map h_1 along with X can be used to generate h_2 , hence forming a recurrence relation as below where h_t contains all the information up to time step t (here f_1^{IC} is RB and f_{aux} is ASB):

$$h_t = f_1^{IC}(X, h_{t-1}); \quad o_t = f_{aux}(h_t); \quad (1)$$

Once we generate $o_t \forall t$, we concatenate them in order of t to form O (where $O : [o_1, o_2, \dots, o_{T-1}]$) and use it to provide $L_{aux}(O, M)$ for supervision. Here we exclude $o_{t=T}$ because this map contains decoded information about all time steps from 0 to t . We send this map to the Projection block (P_R) which then generates the final saliency map S . To implement (1) we need to design a recursive unit which will be able to encode the dependency sequentially. Now, since spatial information is crucial to determining all of m_t or S , it is clear that we cannot use the vanilla fully connected RNN or LSTM architectures, but need to use convolution operations on our activation maps. [RS] uses convolution operations on the input and hidden state spatial maps. However, the input to our recurrent block is always a fixed (X) for all t , which means that determining the next h_t will only be dependant on hidden state (h_{t-1}) and cell state (c_{t-1}) according to the equations of ConvLSTM. Thus, we need to encode a more complex relationship between X and h_{t-1} rather than linear addition to produce h_t . To incorporate this, we propose a Recurrent Convolutional Block (RB) which is constructed by stacking three convolution layers each followed by a batchnorm and ReLU. Figure 3 (b) shows the recursive block used in our model. We provide X and h_{t-1} concatenated as input to this block, which then gives us h_t . This can be formulated as a non-linear function on both X and h_{t-1} i.e. $h_t = f(X, h_{t-1})$. But, if we have a decouple W and U as suggested by ConvLSTM, then the relation becomes $h_t = f_W(X) + f_U(h_{t-1})$. Note that all f , f_W and f_U have more than one convolution layers each followed by batchnorm and ReLU; and thus are non-linear functions. Since f is a function over the combined space of X and h_{t-1} it can learn much more complex relation among the two rather than a linear sum of independent functions applied on them.

4.3 Recurrent Module for Non-Incremental Data

The above formulation will not help us in our non-incrementally arranged data. Now the metadata maps m_t will only contain information of the time step t and not all the information up to it. Since h_t is optimized to predict m_t as we apply the supervision loss L_{aux} on the output of $ASB(h_t)$; it tends to have less information about all the previous states while focusing only on the current state. This makes it much more difficult for RB to predict h_t from h_{t-1} . Assuming RB can learn to output h_t which contains information about time step t only, the next input to RB then cannot be the same h_t since it does not contain all the information up to time t which is necessary for the sequence to be generated in order. Hence, we introduce a Hidden State Accumulator Block ($HSAB$), which keeps track of all the hidden states up to time t and generates an accumulated output k_t which now contains all the information from time 0 to time t . Now that we have modified RB to generate $h_t \forall t$, we pass each of them through ASB and then concatenate all the outputs to form O (where $O : [o_1, o_2, \dots, o_T]$).

This O is then used to compute the supervision loss $L_{aux}(O, M)$. The last h_T is again passed through $HSAB$ to get k_T which can now be passed through P_R (see Figure 3) to get the final saliency maps since k_T contains all the information up to the last time instant T . This can be expressed as follows where f_1^{NIC} is RB , f_2^{NIC} is $HSAB$ and f_{aux} is ASB .

$$h_t = f_1^{NIC}(X, k_{t-1}); \quad k_t = f_2^{NIC}(h_t, k_{t-1}); \quad o_t = f_{aux}(h_t) \forall t; \quad (2)$$

5 Losses

Recent works suggest using saliency evaluation metrics like KL (Kullback–Leibler divergence), SIM (Similarity), CC (Pearson’s Correlation Coefficient) and NSS (Normalized Scan-path Saliency) as losses during training. According to [30] and [21], metrics for comparison of saliency are not coherent i.e. every metric penalizes different aspects in the saliency map. So, training with a single map optimizing on all these metrics [8, 2, 17] will not be able to bring out the best performance of the model for each score. Thus, we propose to have multiple saliency map outputs from our model so as to optimize each map on a different loss. [30] shows how different each metric is from each other, thus to optimize our model on most metrics we choose KL , CC , SIM and NSS to train our network. Since all kinds of $AUCs$ are not differentiable, we exclude them in our losses. We use four output saliency maps such that our total loss is as follows.

$$L_{sal}(S, S_{GT}) = \alpha L_{KL}(S_1, S_{GT}^{gauss}) + \beta L_{CC}(S_2, S_{GT}^{gauss}) + \gamma L_{SIM}(S_3, S_{GT}^{gauss}) + \delta L_{NSS}(S_4, S_{GT}^{pts}) \quad (3)$$

Here, L_{KL} is the standard KL loss as defined in [9], L_{CC} is $1 - CC$, L_{SIM} is $1 - SIM$ and L_{NSS} is $-NSS$. Note that S_{GT}^{pts} is the ground truth map with fixation points, while S_{GT}^{gauss} is the S_{GT}^{pts} blurred using antonio gaussian kernel as in [19]. The values of α , β , γ and δ are chosen after experimentation as described in Section 6.

As our metadata maps, we can have both M_{GT}^{pts} and the blurred M_{GT}^{gauss} . However, the effects of the antonio gaussian function depends on the spatial distribution of points in the binary fixation map. This means that now when a binary fixation map is broken down into $m_{0 \leq t \leq T}^{fix}$ maps, the spatial distribution of points change drastically with respect to the original fixation map. As a result in each $m_{0 \leq t \leq T}^{gauss}$ obtained from corresponding $m_{0 \leq t \leq T}^{fix}$, the intensity values at a given region (where fixation points are present in m_t^{pts}) are different from that of the same region in S_{GT}^{gauss} . So, using M_{GT}^{gauss} would make learning the final map S harder. Therefore, since only NSS uses the fixation points for error calculation we use only M_{GT}^{pts} for supervision and hence $L_{aux}(O, M) = \frac{1}{T} \sum_{t=0}^T L_{NSS}(o_t, m_t^{pts})$.

6 Experiments

For empirical evaluation, the proposed network architectures are trained and tested on 3 publicly available datasets. We conduct a detailed ablation study to find our best performing model settings. Thereafter, we use this model to compare against state-of-the-art methods.

Dataset: Commonly used datasets like MIT1003 [19] and OSIE [56] are not sufficient enough for training huge models with millions of parameters. We use the MIT1003 dataset which uses gaze tracking devices on 15 subjects per image and total of 1003 images to create their dataset. Another similar dataset is the OSIE dataset which contains eye tracking data

on 700 images. While the images in these datasets cover a variety of scenes, the number of images isn't sufficient enough to train deep models. Therefore, for training purpose we use SALICON [12] which has 10000 training and 5000 validation images with well defined target saliency maps and also accepts submissions in their online competition named Large Scale Scene-Understanding (LSUN 2015 and 2017). We perform all our experiments on the 2017 data. For this competition they provide 5000 test images and the results are to be submitted at the given website. SALICON images have a consistent size of 480x640 and they use mouse tracking data to create the corresponding annotations. For the other datasets, we perform fine-tuning on the model trained on SALICON with their respective train sets.

Evaluation Metrics: Previous literature suggests various metrics for evaluating saliency prediction and it is general convention to provide results on many of them for fair comparison since each metric has its own way of measuring performance. We use *NSS*, *KL*, *CC*, and *SIM* to compute our validation scores and to evaluate ablation study, while our results on testing dataset of SALICON are also evaluated on *sAUC*, *AUC_judd*, *AUC_borji* and *IG*. For a detailed discussion on the definition and properties of the mentioned metrics please refer [12].

Training Methodology: We perform initial experiments using the base architecture structure (Figure 3 a) alongwith multi-channel output. First we chose ResNet18 [9] pre-trained on ImageNet1K [8] as our encoder network and trained the model with four output maps using the losses mentioned in Section 5. We train for 10 epochs with cosine learning rate scheduler [23] wherein we start with a optimal lr of $1e-3$, with a batch size of 35 and use SGD optimizer. Since we train on SALICON which has consistent image sizes, hence we choose not to resize the image and train the network with full size image to avoid losing information during resizing. The parameters used in our loss function were found to be $\alpha = 2$, $\beta = 2$, $\gamma = 5$ and $\delta = 1$ after an extensive search. These parameters were chosen so as to obtain good results on all the metrics.

On achieving best possible validation scores with ResNet18 we shifted to DenseNet121 [10] to observe effects of the accuracy of ImageNet pretraining affecting saliency prediction. Even though ResNet18 has almost double parameters of DenseNet121, it still performs worse on ImageNet classification. Results on comparison of the two encoders are shown in Table 1. Since DenseNet121 clearly has a much superior performance than ResNet18 when used in our setup, we use it in all our other experiments. We choose not to use any bigger encoder than DenseNet121 to avoid any further increase in number of parameters. Note that we use encoders pre-trained on ImageNet1K following prior art methods for fair comparison of performance.

Training with Metadata: After evaluating the best base model we move to investigate our recursive model based on usage of temporal metadata. We train using both the incremental and non-incremental data in their corresponding architectures and evaluate. We observe that training takes slightly more time to converge than the base architecture. Hyperparameters were kept the same as the base architecture to ensure fair comparison and the auxiliary loss L_{aux} was given a weight of 0.01 after extensive search. Similar experiments were performed for spatial metadata as well. All the results of the ablation study for these various settings are recorded in Table 1. We observe that the recursive model trained on non-incrementally arranged spatial metadata performs the best among all the variations. Hereafter, we compare this **non-incremental spatial** model - **RecSal-NIS** with prior state-of-the-art methods on various datasets. The results of comparison on MIT1003 and SALICON validation sets are recorded in Table 3 and Table 2. Note that all the other architectures in the prior art use much heavier encoders like ResNet-50, VGG-16 and DenseNet-161 which have close to 100M parameters as compared to our 15.56M. This shows that our method achieves

Training data	Architecture	Training Procedure	SALICON				MIT1003			
			KL	CC	SIM	NSS	KL	CC	SIM	NSS
SALICON	ResNet18 + D + P_R	-	0.239	0.876	0.743	1.983	0.71	0.723	0.542	2.872
	DenseNet121 + D + P_R	-	0.224	0.887	0.761	1.998	0.698	0.747	0.551	2.941
SALICON + Temporal MetaData	DenseNet121 + D + RB + P_R + ASB	Incremental	0.215	0.891	0.786	2.009	0.687	0.756	0.569	3.032
	DenseNet121 + D + RB + HSAB + P_R + ASB	Non-Incremental	0.219	0.894	0.792	2.016	0.685	0.761	0.56	3.035
SALICON + Spatial MetaData	DenseNet121 + D + RB + P_R + ASB	Incremental	0.215	0.901	0.792	2.014	0.672	0.781	0.576	3.051
	DenseNet121 + D + RB + HSAB + P_R + ASB	Non-Incremental	0.206	0.907	0.803	2.027	0.665	0.784	0.583	3.074

Table 1: Ablation study over architectures and metadata types for validation sets of SALICON and MIT1003

Method	SALICON				Method	MIT1003			
	KL ↓	CC ↑	SIM ↑	NSS ↑		KL ↓	CC ↑	SIM ↑	NSS ↑
MDNSal [28]	0.217	0.899	0.797	1.893	DPNSal [43]	0.368	0.692	0.813	2.678
SimpleNet [28]	0.193	0.907	0.797	1.926	DeepFix [44]	-	0.720	0.540	2.580
EML-NET [17]	0.204	0.890	0.785	2.024	SAM-VGG [9]	-	0.757	-	2.852
RecSal-NIS	0.206	0.907	0.803	2.027	SAM-ResNet [9]	-	0.768	-	2.893
					RecSal-NIS	0.665	0.784	0.583	3.074

Table 2: Comparison with prior art in SALICON validation dataset

Table 3: Comparison with prior art in MIT1003 validation dataset

competitive performance in much fewer parameters.

LSUN challenge 2017: We search the hyperparameter space again for improving on $sAUC$ score since it is used for ranking in LSUN challenge. As it has been shown in [52] that NSS is very closely related to $sAUC$, hence we choose to provide maximum weightage to L_{NSS} during training. The results of this training were submitted for the challenge where we secured the second position (Table 4) which is commendable given the low parameter count of our model (Table 5). We consistently rank among top 5 in each metric with the exception of SIM where we stand at eighth position. Note that model trained with $\alpha, \beta, \gamma, \delta$ optimized for performing best on all the metrics (as in ablation study) when evaluated on the SALICON test set performs better for all other metrics but misses out on top 2 $sAUC$ score.

We also attempted submitting our results for the MIT300 and CAT2000 saliency benchmark, but their servers are down. Hence we also validate our methods on the OSIE eye-tracking dataset and compared them with prior art in Table 6.

7 Conclusion

Experimental results demonstrate that applying recursive supervision using temporally and spatially sequenced data improves the performance over a given base model. We also find that the non-incrementally arranged spatial metadata method works better than all other variations. We believe it could be because separated spatial cues make it easier for the network

Method	SALICON						
	sAUC ↑	IG ↑	NSS ↑	CC ↑	AUC ↑	SIM ↑	KL ↓
SimpleNet [28]	0.743	0.880	1.960	0.907	0.869	0.793	0.201
SAM-ResNet [9]	0.741	0.538	1.990	0.899	0.865	0.793	0.610
EML-NET [17]	0.746	0.736	2.050	0.886	0.866	0.780	0.520
MDNSal [28]	0.736	0.863	1.935	0.899	0.865	0.790	0.221
MD-SEM [9]	0.746	0.660	2.058	0.868	0.864	0.774	0.568
RecSal-NIS	0.747	0.854	2.043	0.900	0.866	0.789	0.237

Table 4: Comparison with LSUN'17 leaderboard¹(ranking based on sAUC)

Method	Parameters	sAUC \uparrow
MD-SEM [10]	30.9 M	0.746
SAM-ResNet [10]	~ 70 M	0.741
EML-NET [10]	> 100 M	0.746
SimpleNet [10]	> 86 M	0.743
RecSal-NIS	15.56 M	0.747

Table 5: Parameters vs sAUC comparison with prior art

Method	OSIE			
	KL \downarrow	CC \uparrow	SIM \uparrow	NSS \uparrow
SALICON (implemented by [10])	0.545	0.605	0.762	2.762
DenseSal [10]	0.443	0.659	0.822	3.068
DPNSal [10]	0.397	0.686	0.838	3.175
RecSal-NIS	0.326	0.864	0.720	3.843

Table 6: Comparison with prior art on OSIE validation set

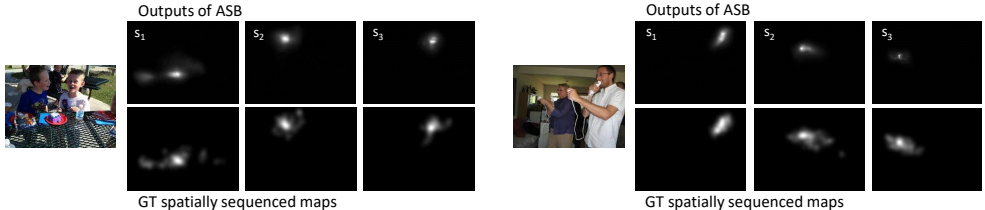


Figure 4: Outputs produced by ASB of model supervised with non-incremental spatial data - RecSal-NIS. s_1 , s_2 and s_3 are output maps of ASB after each iteration of RB in order of occurrence

to extract important features specific to those regions which contribute to the final saliency pattern. Our work suggests that improvement in performance does not necessarily require higher parameters, but rather an efficient usage of data.

References

- [1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.
- [2] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 41(3):740–757, 2018.
- [3] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. Predicting human eye fixations via an lstm-based saliency attentive model. *IEEE Transactions on Image Processing*, 27(10):5142–5154, 2018.
- [4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [5] Konstantin Dmitriev and Arie E Kaufman. Learning multi-class segmentations from single-class datasets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9501–9511, 2019.
- [6] Samuel F Dodge and Lina J Karam. Visual saliency prediction using a mixture of deep neural networks. *IEEE Transactions on Image Processing*, 27(8):4080–4090, 2018.
- [7] Camilo Fosco, Anelise Newman, Pat Sukhum, Yun Bin Zhang, Aude Oliva, and Zoya Bylinskii. How many glances? modeling multi-duration saliency. In *SVRHM Workshop at NeurIPS*, 2019.

-
- [8] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in neural information processing systems*, pages 545–552, 2007.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
- [12] Xun Huang, Chengyao Shen, Xavier Boix, and Qi 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.
- [13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- [14] Md Amirul Islam, Mahmoud Kalash, and Neil DB Bruce. Semantics meet saliency: Exploring domain affinity and models for dual-task prediction. *arXiv preprint arXiv:1807.09430*, 2018.
- [15] Laurent Itti and Pierre F Baldi. Bayesian surprise attracts human attention. In *Advances in neural information processing systems*, pages 547–554, 2006.
- [16] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998.
- [17] Sen Jia and Neil DB Bruce. Eml-net: An expandable multi-layer network for saliency prediction. *Image and Vision Computing*, page 103887, 2020.
- [18] Lai Jiang, Mai Xu, Tie Liu, Minglang Qiao, and Zulin Wang. Deepvs: A deep learning based video saliency prediction approach. In *Proceedings of the european conference on computer vision (eccv)*, pages 602–617, 2018.
- [19] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *2009 IEEE 12th international conference on computer vision*, pages 2106–2113. IEEE, 2009.
- [20] Srinivas SS Kruthiventi, Kumar Ayush, and R Venkatesh Babu. Deepfix: A fully convolutional neural network for predicting human eye fixations. *IEEE Transactions on Image Processing*, 26(9):4446–4456, 2017.
- [21] Matthias Kummerer, Thomas S. A. Wallis, and Matthias Bethge. Saliency benchmarking made easy: Separating models, maps and metrics. In *The European Conference on Computer Vision (ECCV)*, September 2018.

- [22] Xi Li, Liming Zhao, Lina Wei, Ming-Hsuan Yang, Fei Wu, Yueting Zhuang, Haibin Ling, and Jingdong Wang. Deepsaliency: Multi-task deep neural network model for salient object detection. *IEEE transactions on image processing*, 25(8):3919–3930, 2016.
- [23] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.
- [24] David Noton and Lawrence Stark. Scanpaths in eye movements during pattern perception. *Science*, 171(3968):308–311, 1971.
- [25] Taiki Oyama and Takao Yamanaka. Influence of image classification accuracy on saliency map estimation. *CAAI Transactions on Intelligence Technology*, 3(3):140–152, 2018.
- [26] Junting Pan, Cristian Canton Ferrer, Kevin McGuinness, Noel E O’Connor, Jordi Torres, Elisa Sayrol, and Xavier Giro-i Nieto. Salgan: Visual saliency prediction with generative adversarial networks. *arXiv preprint arXiv:1701.01081*, 2017.
- [27] Vasili Ramanishka, Abir Das, Jianming Zhang, and Kate Saenko. Top-down visual saliency guided by captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7206–7215, 2017.
- [28] Navyasri Reddy, Samyak Jain, Pradeep Yarlagadda, and Vineet Gandhi. Tidying deep saliency prediction architectures. *arXiv preprint arXiv:2003.04942*, 2020.
- [29] Douglas A Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, 741, 2009.
- [30] Nicolas Riche, Matthieu Duvinage, Matei Mancas, Bernard Gosselin, and Thierry Dutoit. Saliency and human fixations: State-of-the-art and study of comparison metrics. In *Proceedings of the IEEE international conference on computer vision*, pages 1153–1160, 2013.
- [31] Oindrila Saha, Rachana Sathish, and Debdoot Sheet. Learning with multitask adversaries using weakly labelled data for semantic segmentation in retinal images. In *International Conference on Medical Imaging with Deep Learning*, pages 414–426, 2019.
- [32] Hamed R Tavakoli, Fawad Ahmed, Ali Borji, and Jorma Laaksonen. Saliency revisited: Analysis of mouse movements versus fixations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1774–1782, 2017.
- [33] Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.
- [34] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan. Salient object detection with recurrent fully convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1734–1746, 2018.
- [35] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

-
- [36] Juan Xu, Ming Jiang, Shuo Wang, Mohan S. Kankanhalli, and Qi Zhao. Predicting human gaze beyond pixels. *Journal of Vision*, 14(1):1–20, 2014.
- [37] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1265–1274, 2015.