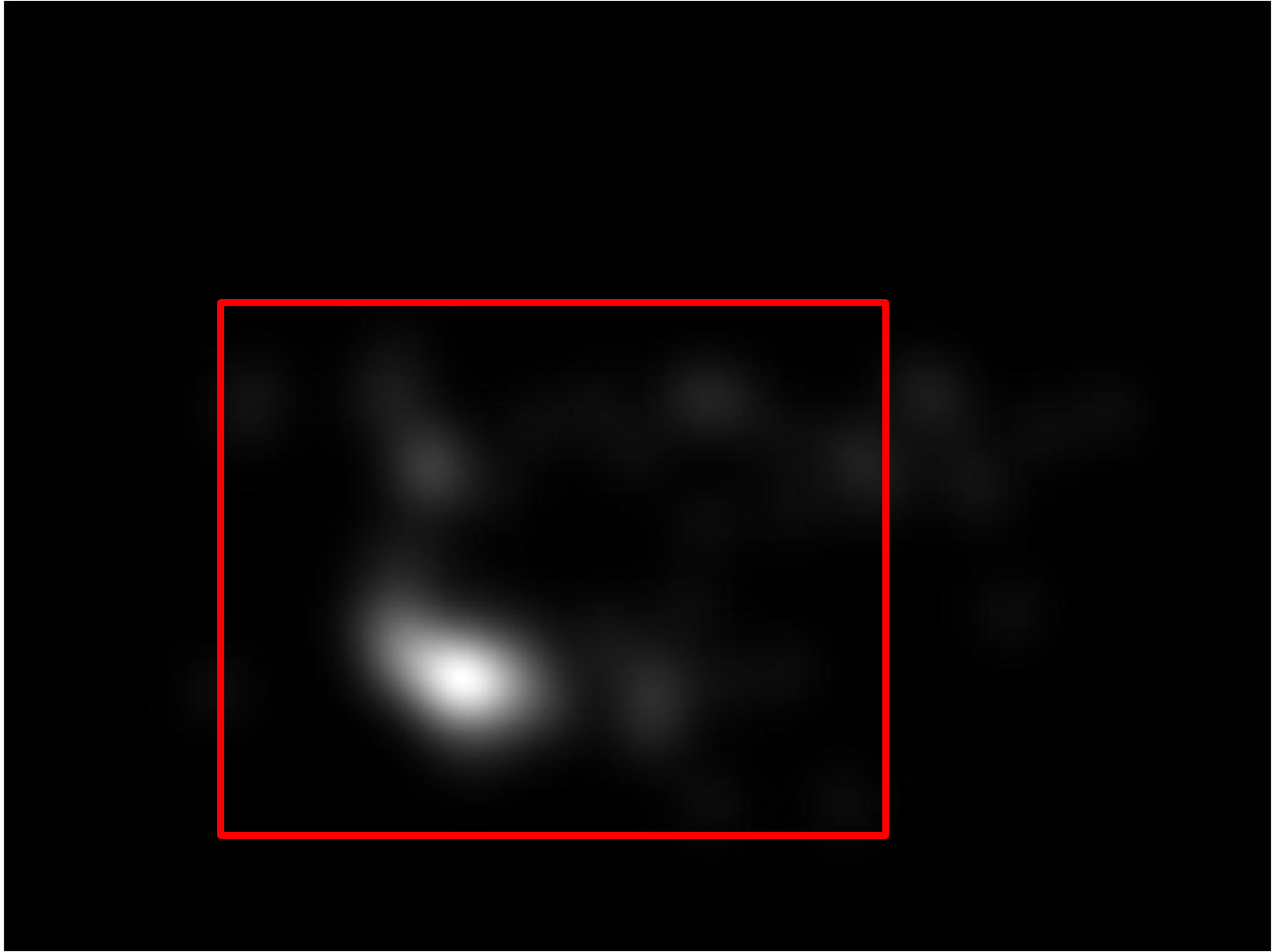
Robust Framework for Saliency Prediction in Images

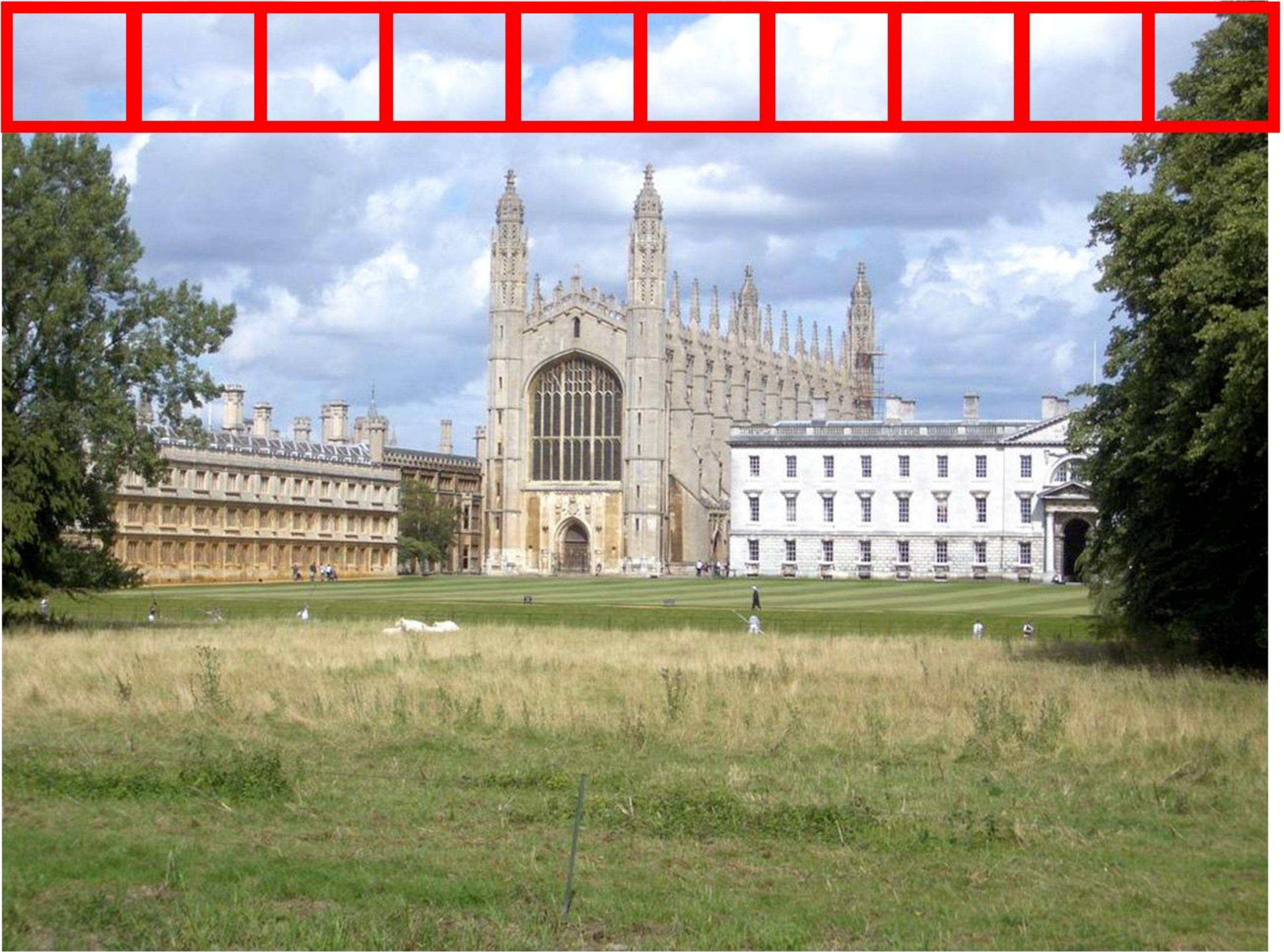
Where do you look on these images?



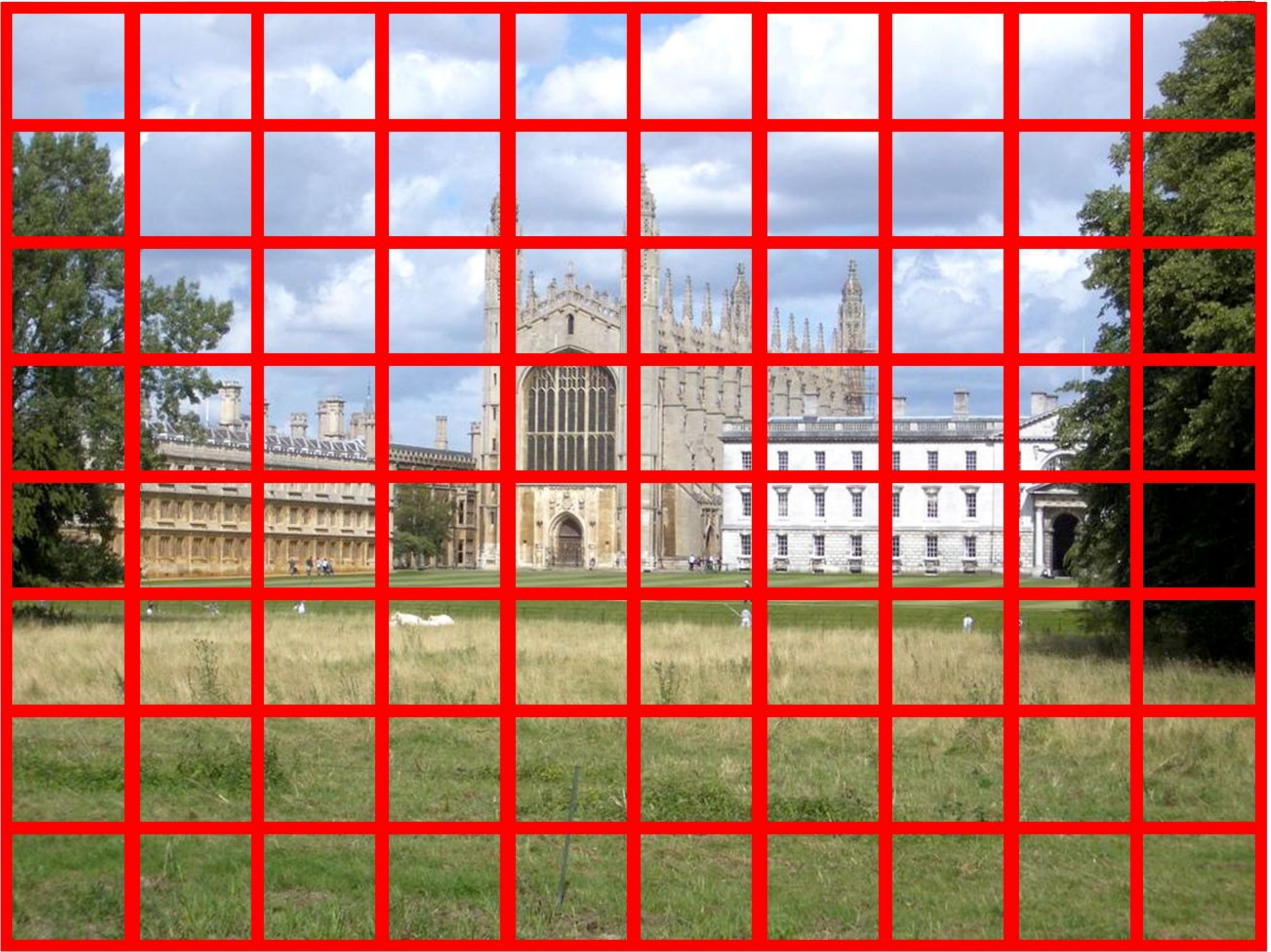
Imagine that you are a robot, and you've received this image from your camera. You need to run some expensive localization computations to decide where you are.

You could do this across the whole image....but you likely don't need to run the computation everywhere.

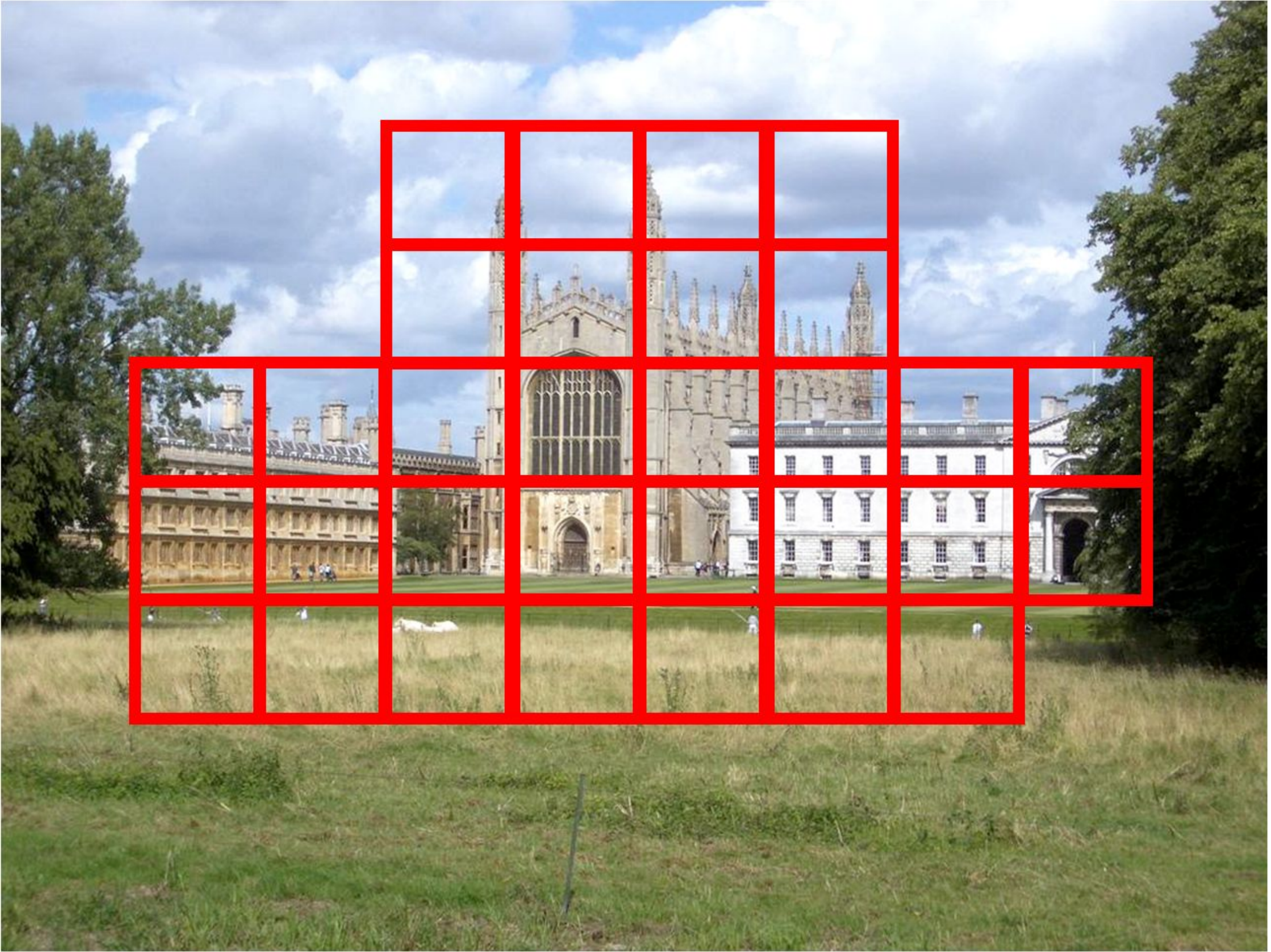
Running computations everywhere would take a long time .



Running computations everywhere would take a long time .



Running computations everywhere would take a long time .



Instead, doing it just here, or doing it here first could save you a lot of time.

Therefore,
need to prioritize the
visual information
and decide what is most important

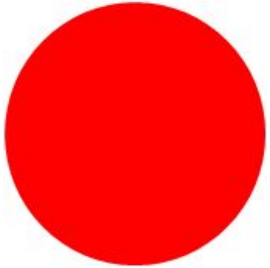
Understanding attention enables applications in computer graphics & vision, design

- Image Cropping/Thumbnailing
- Image and Video Compression
- Non-Photorealistic rendering
- Scene Understanding
- Advertising and Package Design
- Web Usability

- Localization/Recognition
- Object Detection
- Navigational Assistance
- Robot Action Vision
- Surveillance Systems
- Assistive Technology for blind or low-vision people

Where we move our eyes is dictated by two mechanisms

- Bottom-Up Mechanisms
- Top-Down Mechanisms



Visual Attention Mechanisms

Bottom-Up

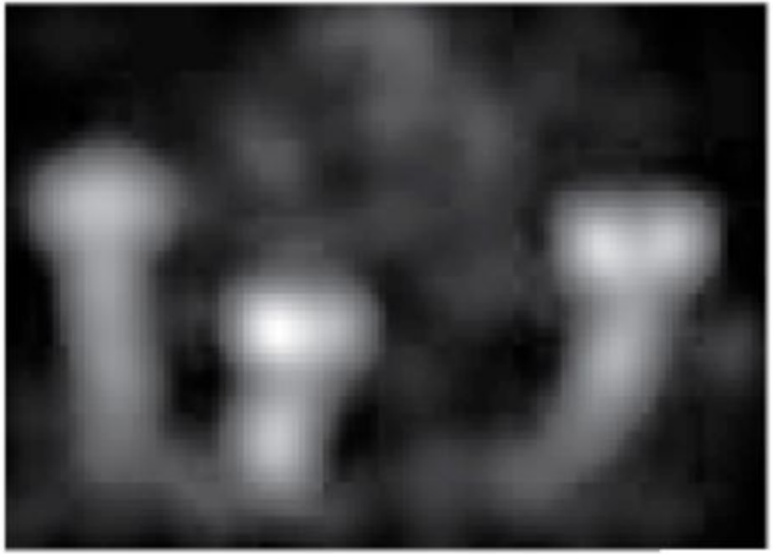
- Automatic
- Reflexive
- Stimulus-driven

Top-Down

- Subject’s Prior Knowledge
- Expectations
- Task Oriented
- Memory
- Behavioural Goals

Researchers create computational models of visual attention to predict where people look

Image



Saliency Map

Proposed Solution

The Ingredients

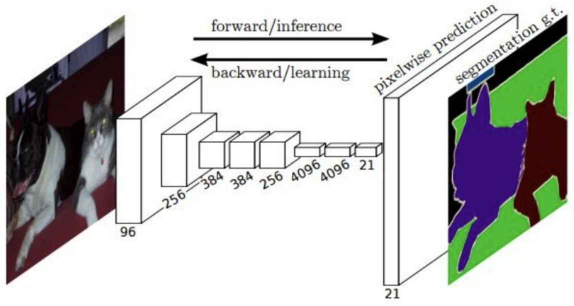
Very Deep Network

- 20 layers
- Small kernel sizes

The Ingredients

Fully Convolutional Network

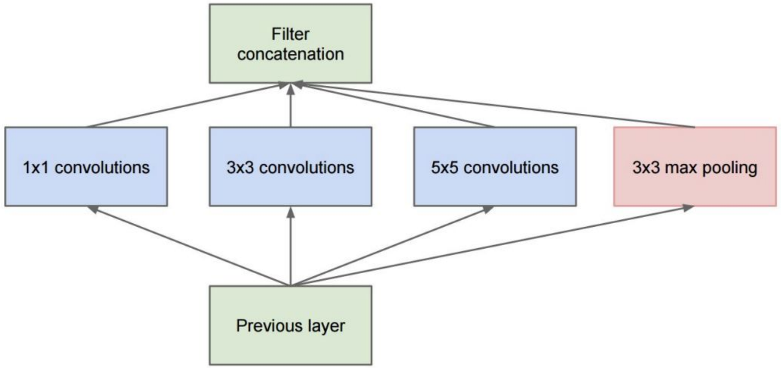
- Fully connected layers at the end are replaced by convolutional layers with very large receptive fields.
- They capture the global context of the scene.
- End-to-end training



The Ingredients

Inception Layers

- GoogLeNet
- Different kernel sizes operating in parallel.

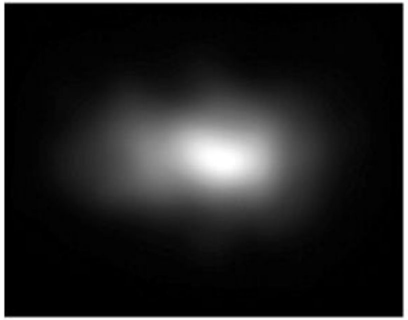


The Ingredients

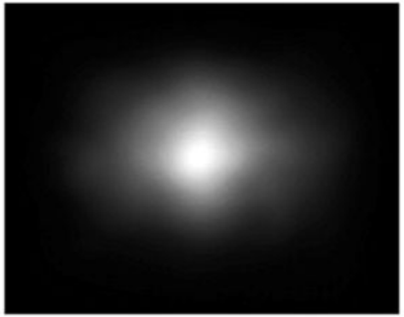
Center Biased Convolutional (LBC) layer

- Human Eye Fixations are Center Biased
  - Photographer Bias
  - Viewing Strategy

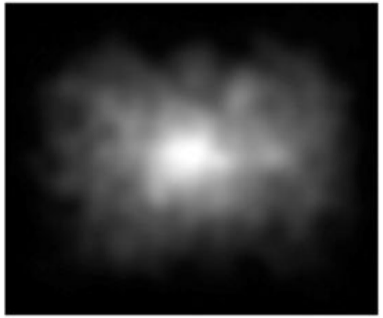
- Introducing CBC layer to model Center Bias



Bruce & Tsotsos (2005)



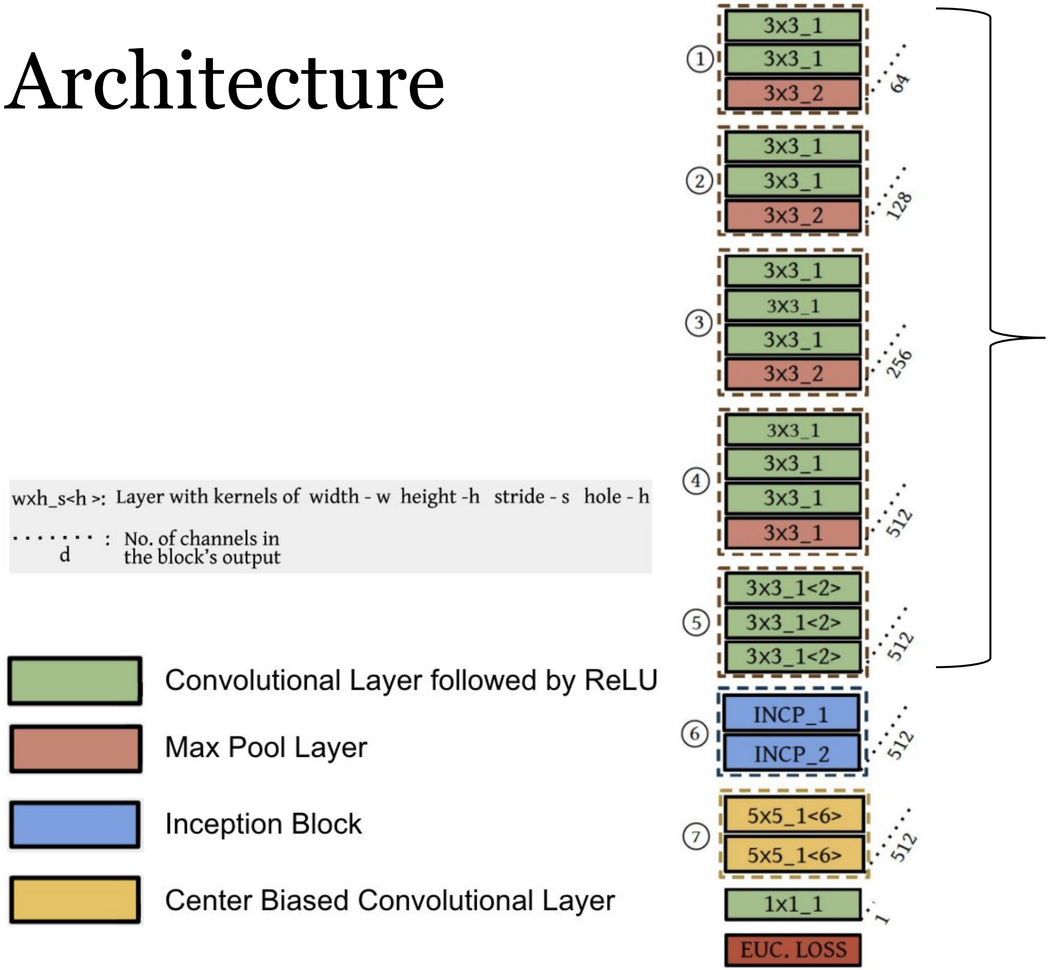
Judd (2009)



Judd (2011)

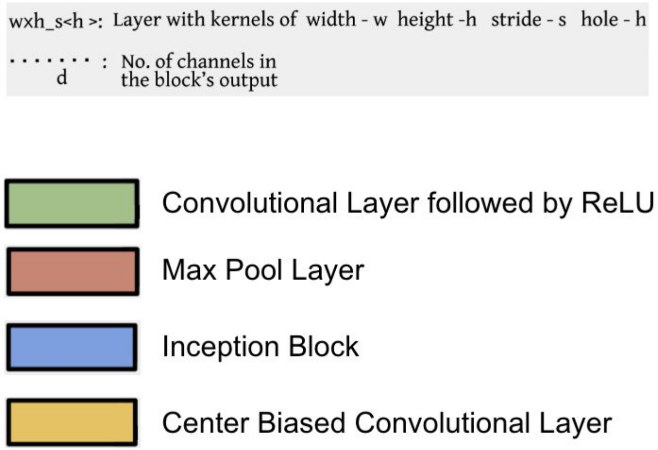
The Network

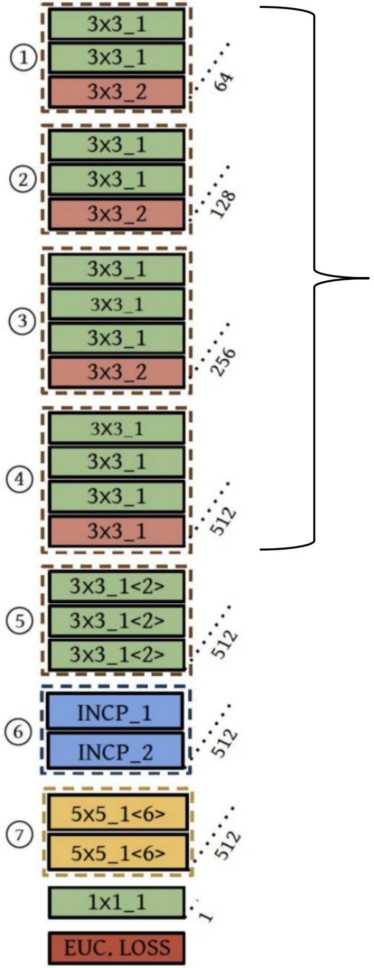
Architecture



Small convolutional filters of 3x3 with stride of 1 to allow a large depth without increasing the memory requirement

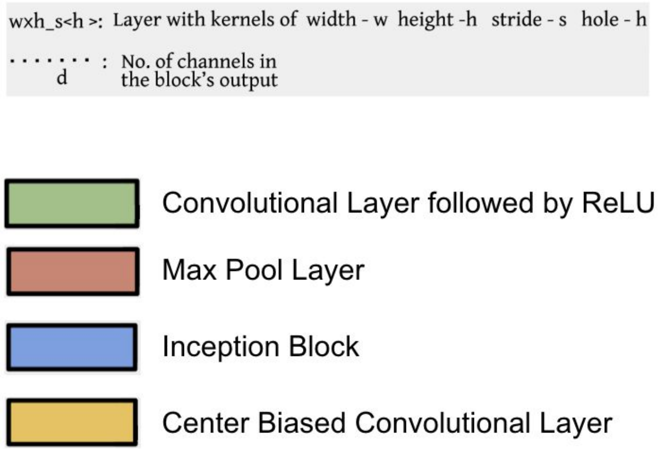
Architecture

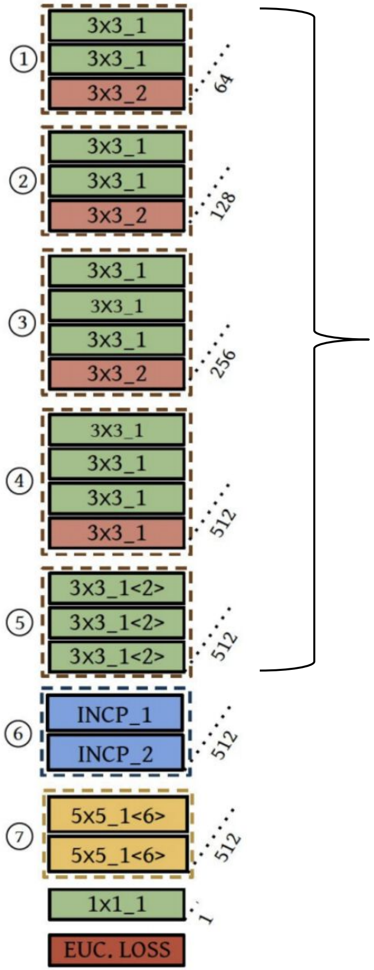




Max pooling layers (in red) reduce computation.

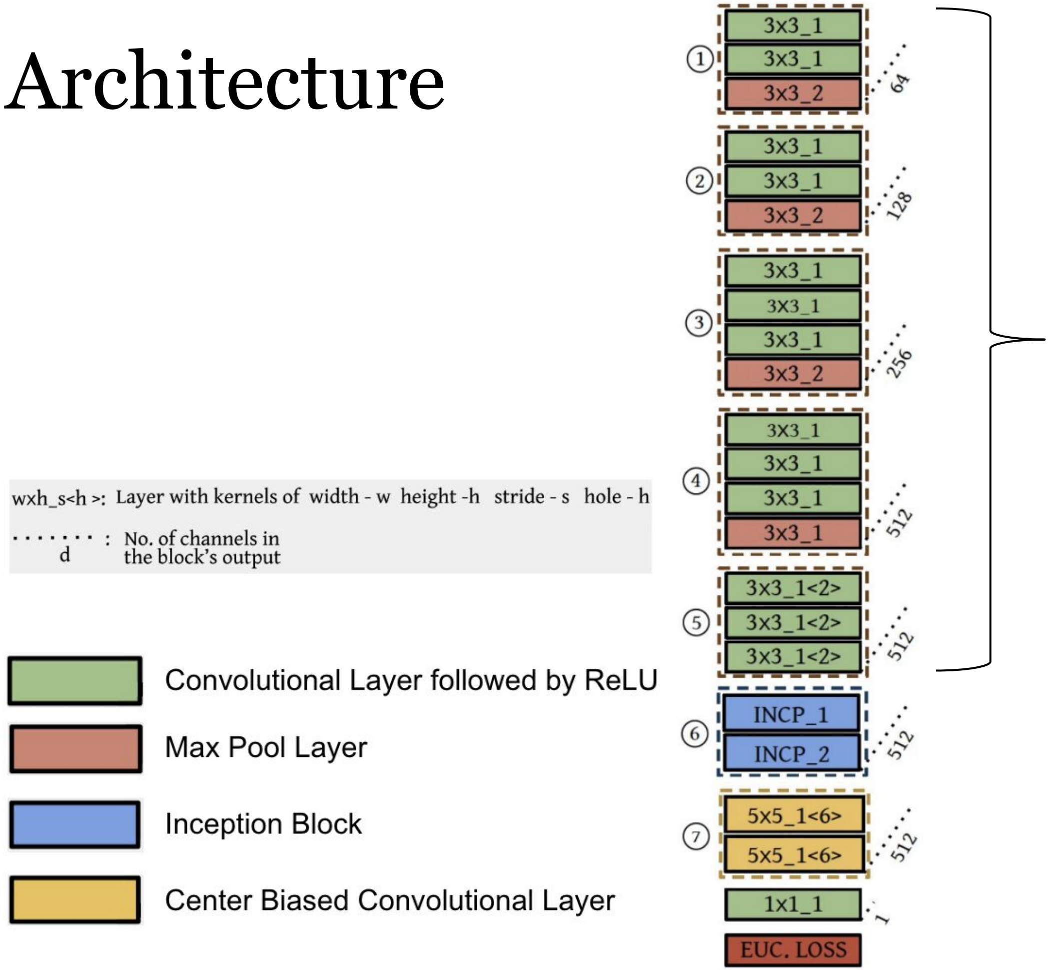
Architecture





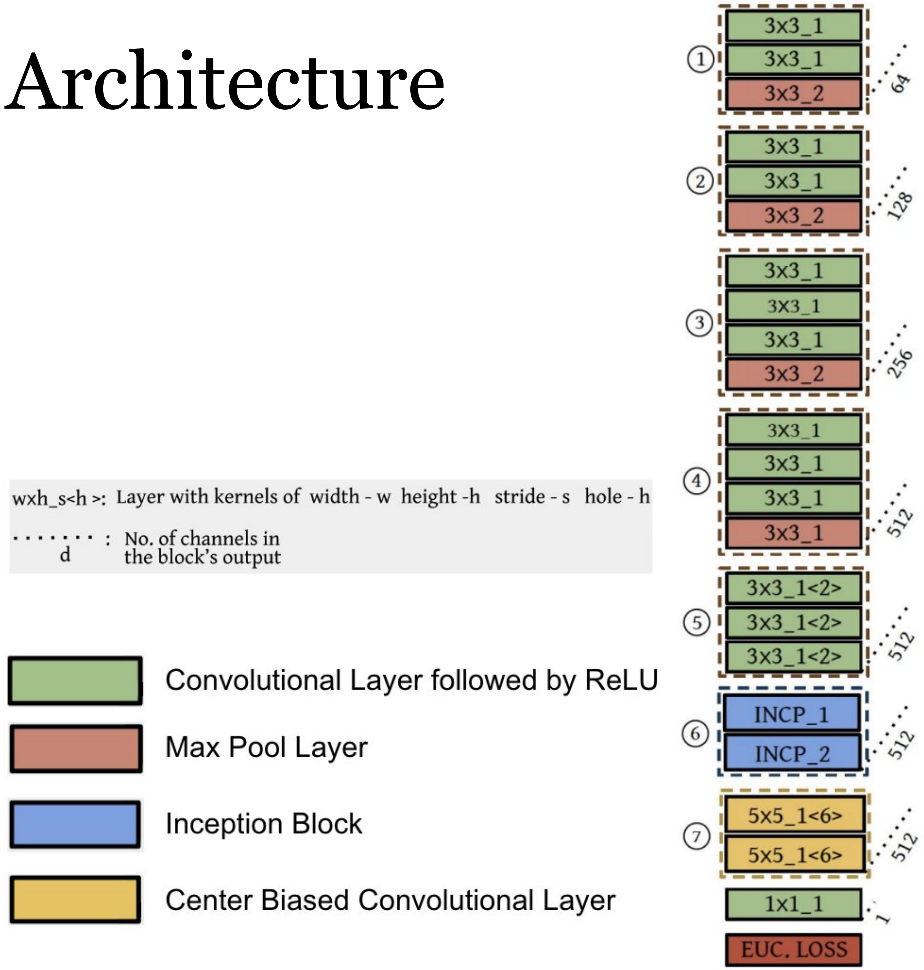
Gradual increase in the amount of channels to progressively learn richer semantic representations: 64, 128, 256, 512...

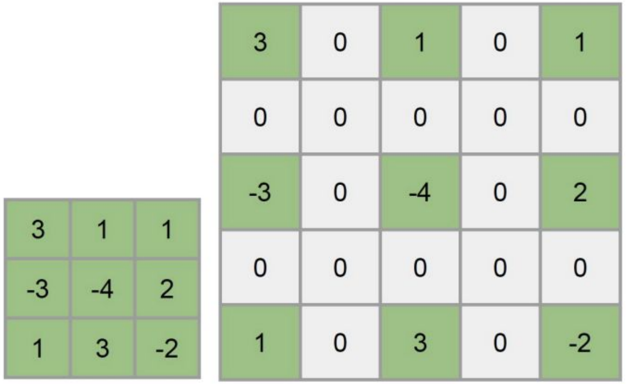
Architecture



Weights initialized from VGG-16 net for stable and effective learning

Architecture





(a) Conv. kernel of size 3x3 (b) Conv. kernel of size 3x3 with hole - 2

Convolution kernel 3x3 with hole size 2 have a receptive field of 5x5

Architecture

wxh_s<h >: Layer with kernels of width - w height -h stride - s hole - h

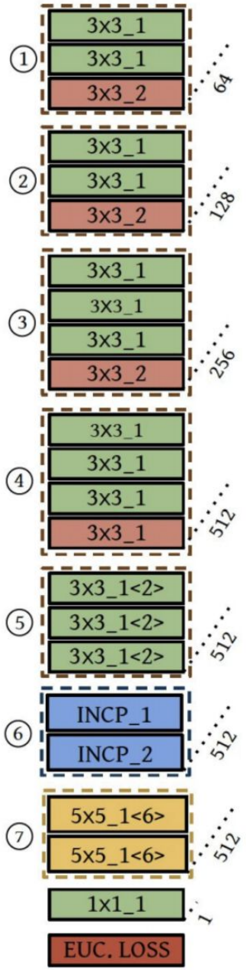
. . . . : No. of channels in the block's output
d

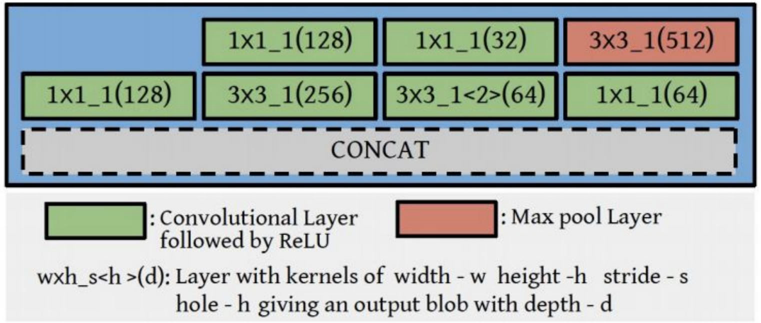
Convolutional Layer followed by ReLU

Max Pool Layer

Inception Block

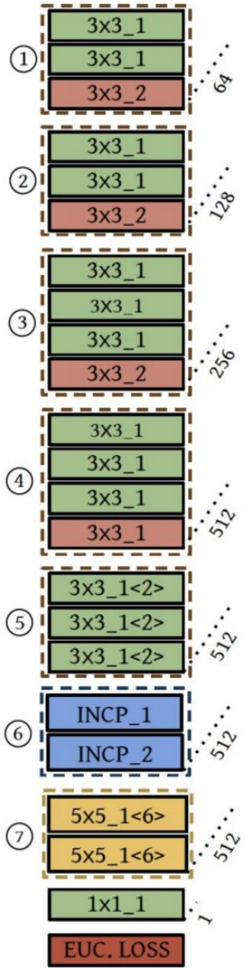
Center Biased Convolutional Layer





Capture multi-scale semantic structure using two inception style convolutional modules

Architecture



wxh_s<h >: Layer with kernels of width - w height -h stride - s hole - h

. . . . : No. of channels in the block's output
d

Convolutional Layer followed by ReLU

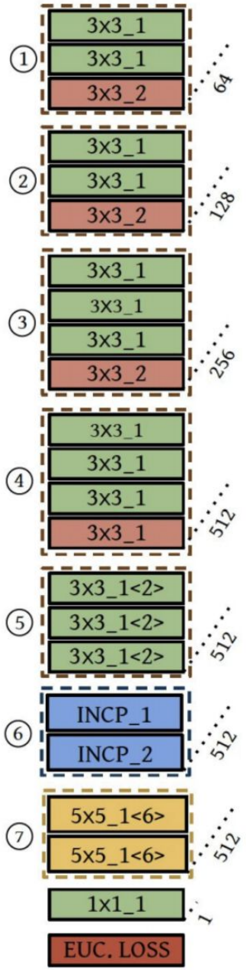
Max Pool Layer

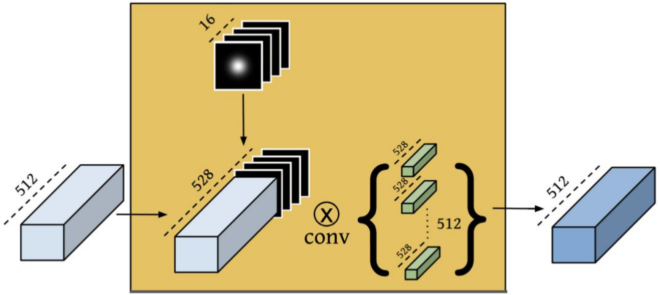
Inception Block

Center Biased Convolutional Layer

Very large receptive fields of 25x25 by introducing holes of size 6 in kernels

Architecture





wxh_s<h >: Layer with kernels of width - w height -h stride - s hole - h

d : No. of channels in the block's output

Convolutional Layer followed by ReLU

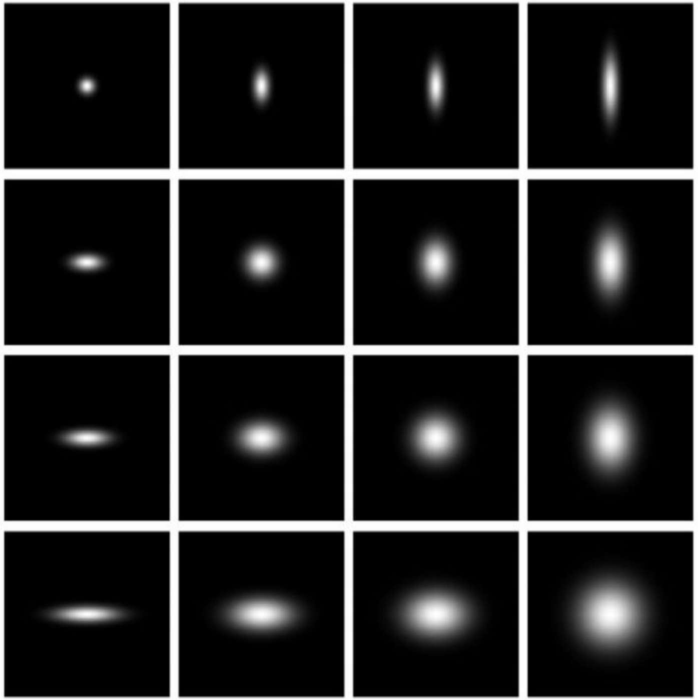
Max Pool Layer

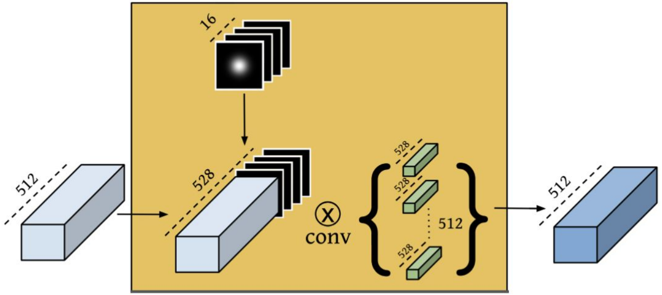
Inception Block

Center Biased Convolutional Layer

Center Biased Convolutional (CBC) layers

Architecture



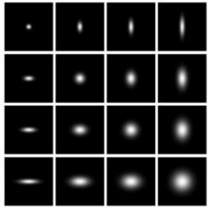


Center Biased Convolutional (LBC) layers

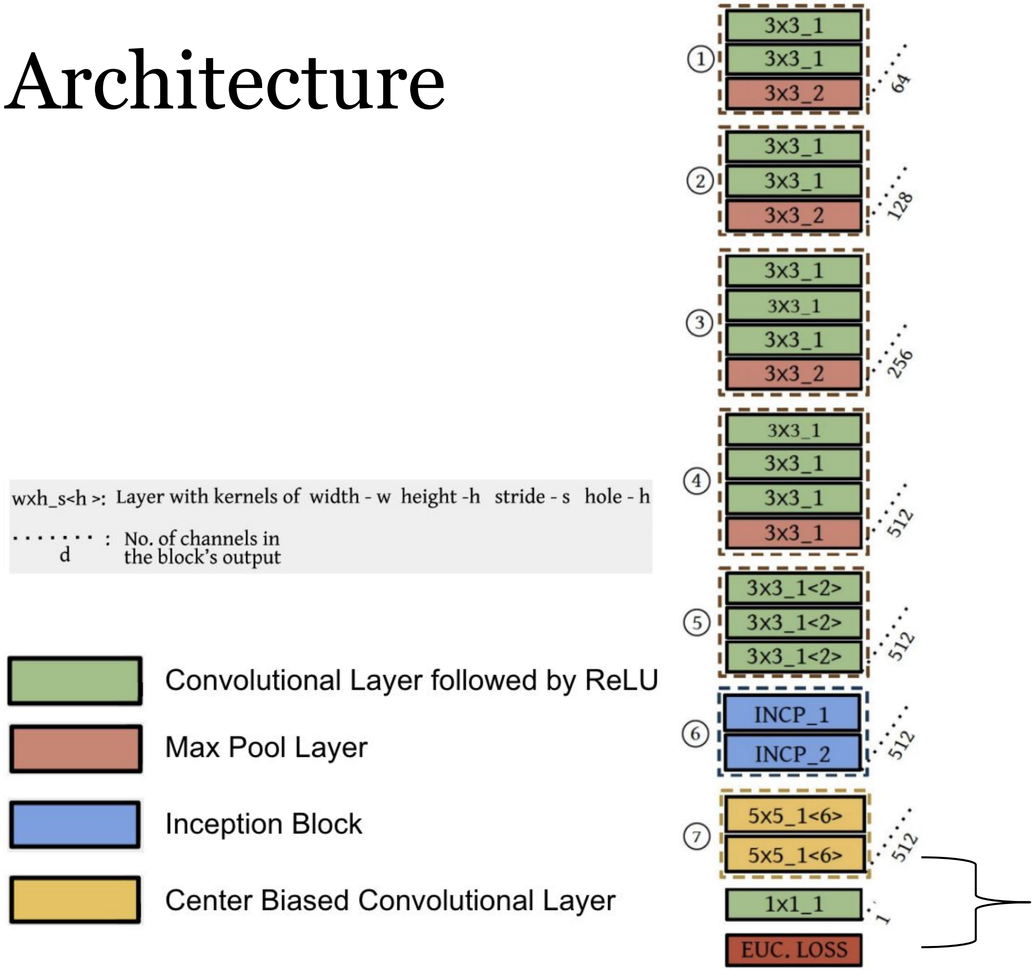
Architecture

$$R_c(x,y) = \mathcal{R}\left(\sum_{i,j} \left(\overbrace{\mathbf{I}(x+i, y+j)}^{\text{input blob}} * \overbrace{\boldsymbol{W}_{\boldsymbol{c}}(i,j)}^{\text{weights from c'th filter in a convolutional layer}} + \right. \right.$$

$$\left. \left. \underbrace{\mathbf{L}(x+i, y+j)}_{\text{constant during training}} * \underbrace{\boldsymbol{W}'_{c}(i,j) + b_c}_{\text{learnt during training}} \right) \right)$$



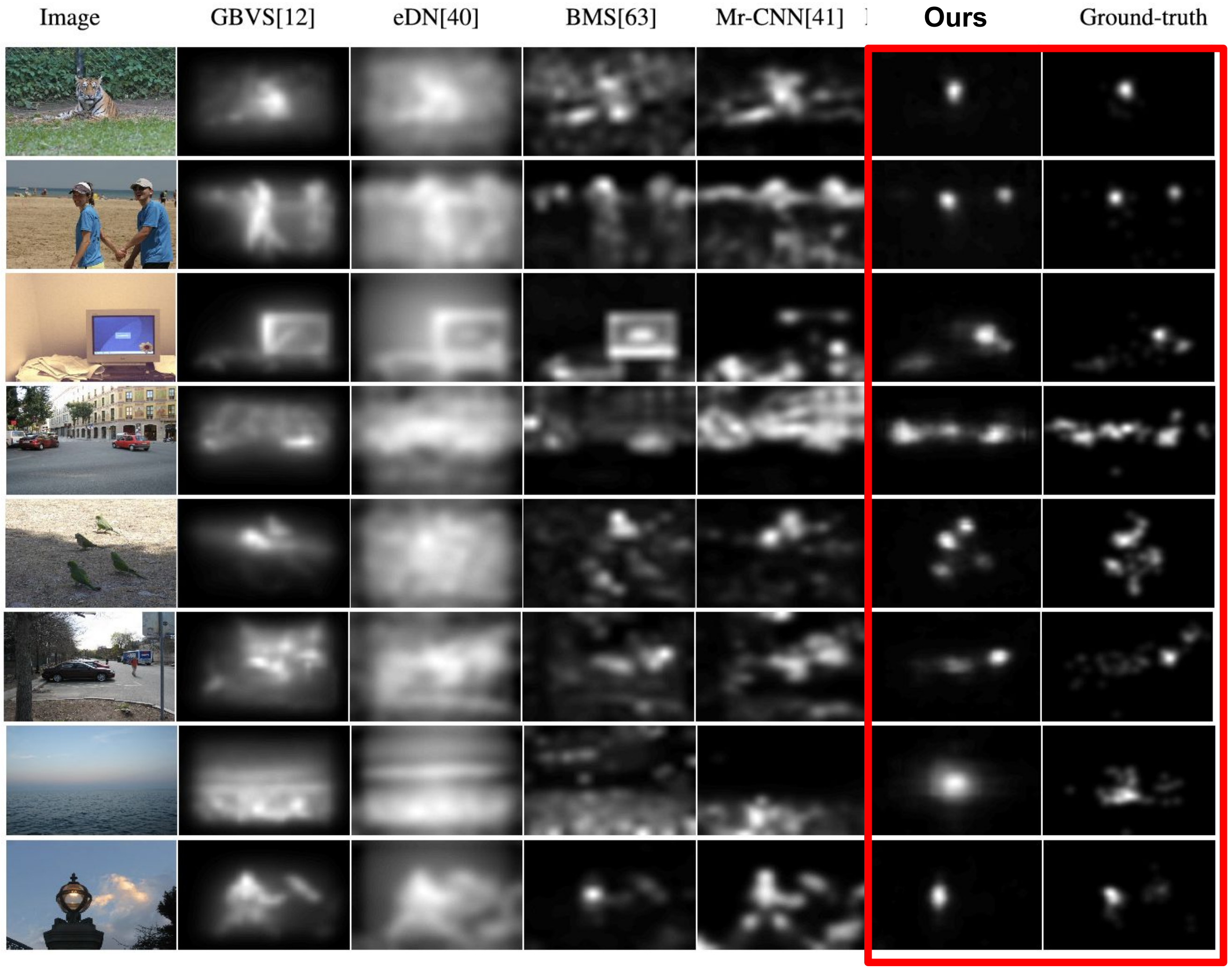
Architecture



Final output W/8xH/8 is upsampled.

Experiments

Results



Comparison of Ground Truth and Predicted Saliency Map

Various metrics are used to evaluate the performance of a given Saliency Model

- AUC - Judd
- AUC - Borji
- Shuffled - AUC
- Earth Mover's Distance
- Similarity
- Correlation Coefficient
- Normalized Saliency Scanpath
- Kullback-Leibler divergence

Results

TABLE I
Experimental Evaluation on CAT2000 Test Set

Method	AUC-Judd	SIM	EMD	AUC-Borji	Shuff. AUC	CC	NSS
Ours	**0.87**	**0.75**	**1.11**	**0.81**	**0.57**	**0.88**	**2.29**
CAS[68]	0.77	0.50	3.09	0.76	0.60	0.42	1.07
Judd[57]	0.84	0.46	3.61	0.82	0.56	0.54	1.30
GBVS[12]	0.80	0.51	2.99	0.79	0.58	0.50	1.23

Results

TABLE II
Experimental Evaluation on MIT300 Test Set

Method	AUC-Judd	SIM	EMD	AUC-Borji	Shuff. AUC	CC	NSS
Ours	**0.87**	**0.67**	**2.04**	**0.80**	**0.71**	**0.78**	**2.26**
Mr-CNN[41]	0.77	0.45	4.33	0.76	0.69	0.41	1.13
DG-I[38]	0.84	0.39	4.97	0.83	0.66	0.48	1.22
BMS[63]	0.83	0.51	3.35	0.82	0.65	0.55	1.41
eDN[40]	0.82	0.41	4.56	0.81	0.62	0.45	1.14
CAS[68]	0.74	0.43	4.46	0.73	0.65	0.36	0.95
Judd[57]	0.81	0.42	4.45	0.80	0.60	0.47	1.18
GBVS[12]	0.81	0.48	3.51	0.80	0.63	0.48	1.24

Results

TABLE III
Experimental Evaluation on PASCAL-S DataSet

Method	AUC-Judd	SIM	EMD	AUC-Borji	Shuff. AUC	CC	NSS
Ours	**0.91**	**0.65**	**0.54**	**0.82**	**0.73**	**0.78**	**2.60**
SU[47]	0.89	0.59	0.73	0.81	0.72	0.69	2.22
JN[69]	0.88	0.50	1.04	0.86	0.69	0.68	1.90
eDN[40]	0.89	0.39	1.29	0.87	0.65	0.55	1.42
BMS[63]	0.80	0.41	1.32	0.78	0.67	0.44	1.28
GBVS[12]	0.84	0.43	1.16	0.82	0.65	0.51	1.36

Results

TABLE IV
Experimental Evaluation on OSIE DataSet

Method	AUC-Judd	SIM	EMD	AUC-Borji	Shuff. AUC	CC	NSS
Ours	**0.91**	**0.66**	**1.04**	**0.83**	**0.79**	**0.80**	**3.04**
eDN[40]	0.82	0.36	2.02	0.82	0.68	0.40	1.16
BMS[63]	0.83	0.43	1.89	0.82	0.76	0.46	1.47
GBVS[12]	0.82	0.42	1.67	0.80	0.68	0.44	1.35
AWS[70]	0.82	0.42	1.93	0.81	0.76	0.45	1.45

Results

TABLE V
Experimental Evaluation on FIGRIM DataSet

Method	AUC-Judd	SIM	EMD	AUC-Borji	Shuff. AUC	CC	NSS
Ours	**0.90**	**0.66**	**1.10**	**0.84**	**0.67**	**0.80**	**2.51**
eDN[40]	0.87	0.37	2.88	0.86	0.62	0.50	1.38
BMS[63]	0.76	0.38	3.00	0.73	0.74	0.34	1.05
GBVS[12]	0.82	0.43	2.29	0.81	0.62	0.45	1.26
AWS[70]	0.72	0.36	3.20	0.74	0.64	0.29	0.89

Thanks!