Solution:

Problem 1:

[1]: **which aims at jointly extracting edges as well as their category information, has far-reaching applications in domains such as semantic segmentation, object proposal generation, and object recognition.**

[2]: **Generates thin edge-maps that are plausible for human eyes; it can be used in any edge detection task without previous training or fine tuning process. This is used in segmentation, image recognition , or even in the modern tasks such as image-to-image translation , photo sketching and so on.**

Problem 2:

1) Summarization:

[1]: SED is aimed at both detecting visually salient edges and recognizing their categories. The proposed DFF model consists of two main components: **a feature extractor with a normalizer** and **an adaptive weight fusion module.**

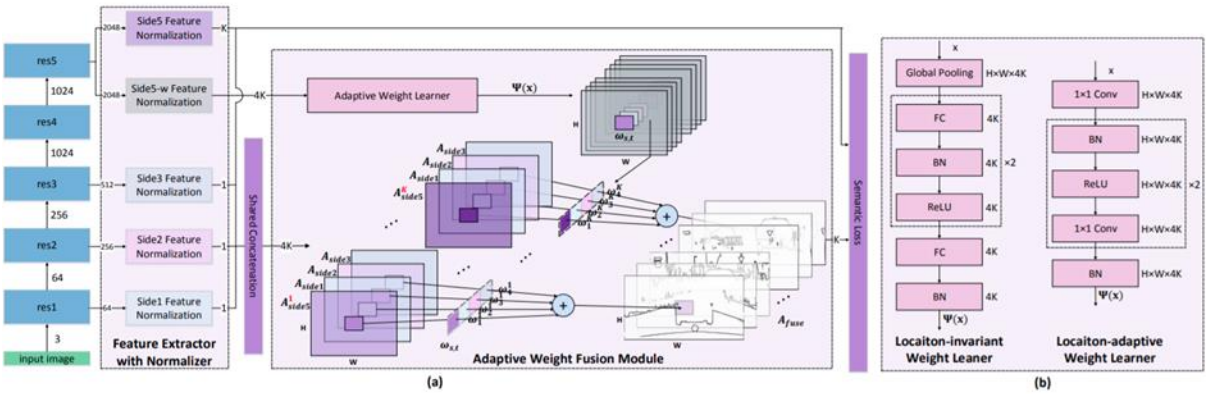


Figure 1. The Location-adaptive weight learner (in figure 1b) makes all the differences compared to previous architectures.

[2]:

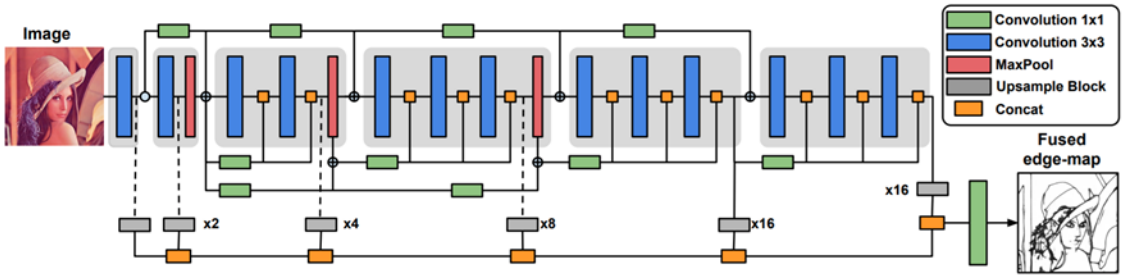


Figure 2. . Proposed architecture: Dense Extreme Inception Network, consists of an encoder composed by six main blocks (showed in light gray). The main blocks are connected between them through 1x1 convolutional blocks. Each of the main blocks is composed by sub-blocks that are densely interconnected by the output of the previous main block. The output from each of the main blocks is fed to an upsampling block that produces an intermediate edge-map in order to build a Scale Space Volume, which is used to compose a final fused edge-map.

2) The main differences:

- [1] Have semantic classification with many object categories in output.

- [2] Just aims at detecting the edges of objects regarless any category.

Problem 3

The final classification accuracy of ResNet 18 on the various datasets is tabulated below.

Dataset	CIFAR-10	CIFAR-100	SVHN	Tiny ImageNet
Train Accuracy	0.9851	0.9655	0.9764	0.9389
Test Accuracy	0.7347	0.4083	0.9267	0.2674

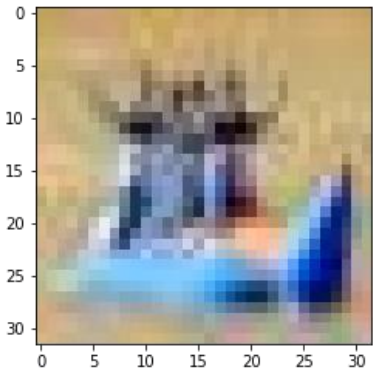
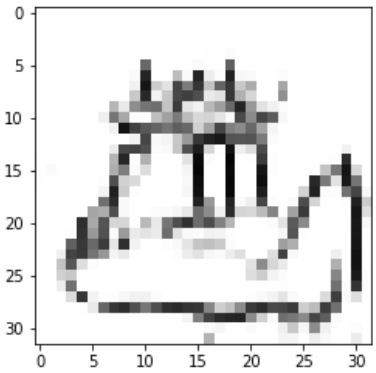
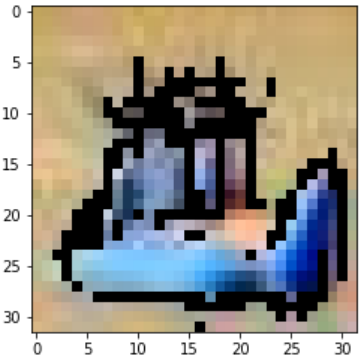
Problem 4

A sample of edges detected is shown below. The first, second and third row correspond to the original image, the edges and the enhanced images respectively.

Problem 5

A sample of edges detected using DexiNed is shown below. The first, second and third row correspond to the original image, the edges and the enhanced images respectively.

Focusing on a particular example:

Problem 6

The final validation accuracy of ResNet-18 on the edges generated from various datasets are tabulated below.

Dataset	**CIFAR-10**	**CIFAR-100**	**Tiny ImageNet**	**SVHN**
Edges	0.7850	0.6175	0.4244	0.6786

Problem 7

The final validation accuracy of ResNet-18 on the edge enhanced images generated from various datasets are tabulated below.

Dataset	**CIFAR-10**	**CIFAR-100**	**Tiny ImageNet**	**SVHN**
Edge Enhanced	0.7938	0.4844	0.5450	0.8488

Problem 8

The final **validation set accuracy** from the previous problems are tabulated below:

Dataset	**CIFAR-10**	**CIFAR-100**	**Tiny ImageNet**	**SVHN**
Original	0.7347	0.4148	0.3181	0.9267

Edges	0.7850	0.6175	0.4244	0.6786
Edge Enhanced	0.7938	0.4844	0.5450	0.8488

Hence we can observe that there is some improvement in classification accuracy after “edge enhancement’.

Although compared to other image classification algorithms, CNNs actually use very little preprocessing. This means that they can **learn** the filters that have to be hand-made in other algorithms. However, enhancing edges seems to be beneficial for classification. CNNs work by extracting features from images i.e. they are learned while the network trains over the images.

Now enhancing the edges in the dataset can make it easier for the neural network to extract high level features from the images, thus we observe higher accuracy.

Problem 9

Edge detection is one of the most commonly used operations in image analysis. An edge is defined by a discontinuity in gray level values. In other words, an edge is the boundary between an object and the background. The shape of edges in images depends on many parameters: The geometrical and optical properties of the object, the illumination conditions, and the noise level in the images. The most commonly used edge detection algorithms are ISEF, Canny, Marr-Hildreth, Sobel, Kirsch, Lapla1 and Lapla2.

The existing edge detectors may be classified in the following categories:

1) Gradient edge detectors: Which contains classical operators and uses first directional derivative operation. Their pitfall is that they are sensitive to noise and thus inaccurate.
2) Zero Crossing: They use second derivative and include Laplacian operator and second directional derivative. Their pitfalls are re-detecting to some of the existing edges and sensitivity to noise.
3) Gaussian Edge Detectors: Which is symmetric along the edge and reduces the noise by smoothing the image. The significant operators here are Canny and ISEF (Shen-Castan) which convolve the image with the derivative of Gaussian for Canny and ISEF for Shen-Castan. The pitfalls are that reque Complex Computations, False zero Crossing and are Time consuming.

In general, edge detectors suffer from the classical drawback of disjointed contours which is a critical problem in the case of document binarization.

One way to address the pitfalls is to use Deep Learning models by applying CNNs at different scales to a large set of images together with the training regularization techniques.

The network generates edges from each convolutional block constructing a multi-scale learning architecture. The training process uses a modified cross entropy loss function for each predicted edge-maps. We can feed every output from each convolution from every block, or perform a set of fusion backward processes, with the data of each output.