P05 report file

Darya Ansaripour

July 31, 2024

Abstract

This report consists of an overview on the fifth project.

1 INTRODUCTION

As requested, DOG and gabor filters have been implemented and applied on a two-layer network, utilizing the STDP rule along with KWTA and lateral inhibition for learning purposes.

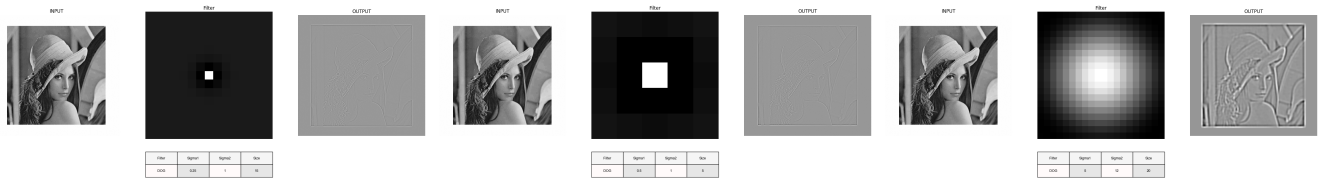
2 ANALYSIS AND BEHAVIORS

2.1 Part1

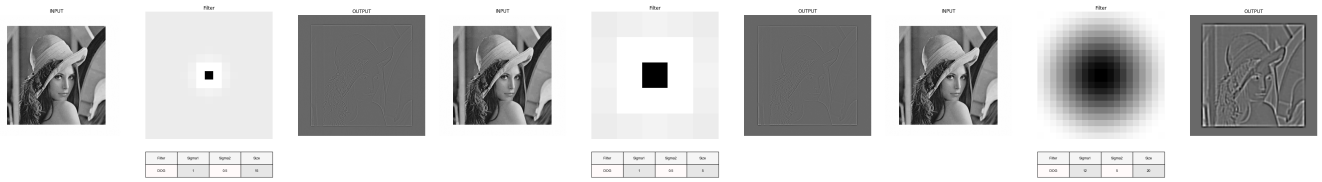
2.1.1 DOG Filter

The DoG filter is applied to the image below with varying sizes and parameters. The DoG filter works by subtracting one blurred version of an image from another, less blurred version of the same image. This process highlights edges and details in the image. In this example, it is observable that when sigma1 (the standard deviation of the first Gaussian blur) is greater than sigma2 (the standard deviation of the second Gaussian blur), the filter detects light pixels on a dark background. This occurs because the larger sigma1 causes a broader blur, which smooths out larger features and highlights smaller, lighter details against the darker, more diffused areas. Conversely, when sigma2 is greater than sigma1, the filter detects dark pixels on a light background. In this case, the smaller sigma1 emphasizes smaller, darker features, while the larger sigma2 blurs the surrounding areas more extensively, creating a light background (Look at the eyes of the person in the results).

Figure 1: applying DOG filter to an image.



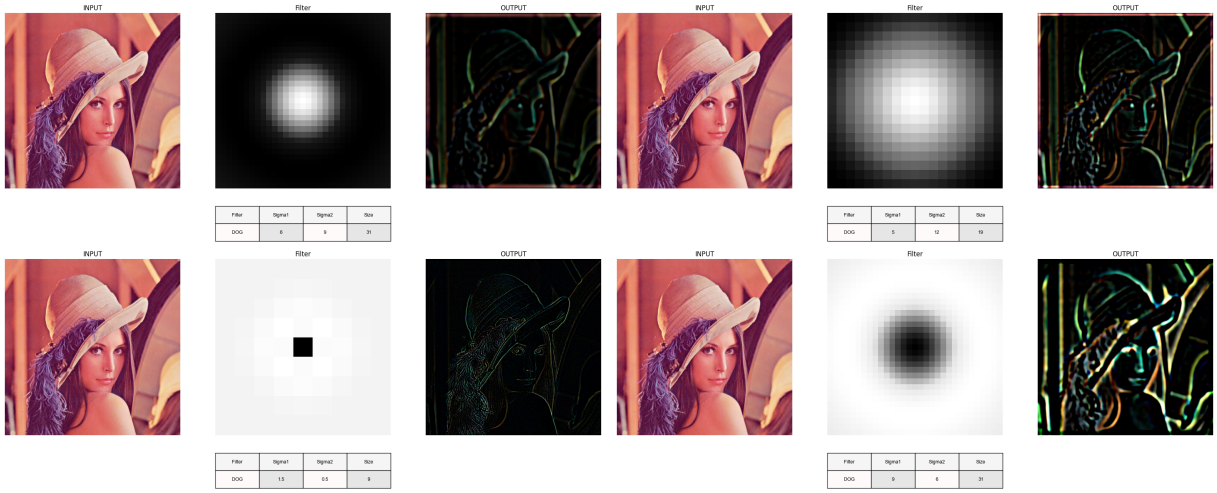
Sigma1 less than Sigma2



Sigma1 greater than Sigma2

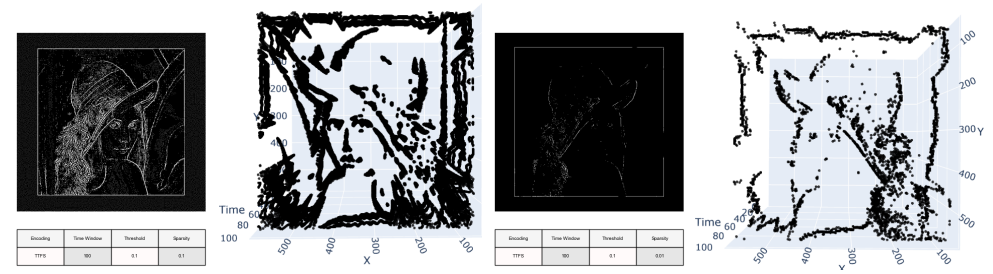
I applied the DoG filter to colored images by separately filtering each color channel (Red, Green and Blue). The result is shown in Figure 2.

Figure 2: applying DOG filter to a colored image.

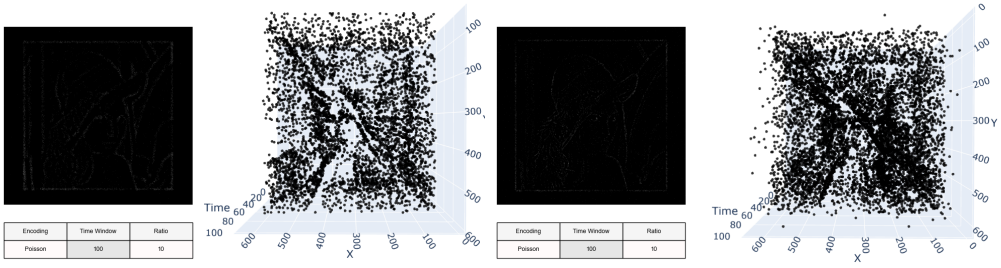


I experimented with two encodings on the filtered images: Poisson and TTFS. Interestingly, the TTFS encoding produced a more detailed result. And here's the cool part: by adjusting the sparsity in this encoding, we can actually manipulate the encoding to result in highlighting brighter edges of the actual picture. (Note: In part 2 Poisson encoding have been used mostly, which enhances the expected results. 2D results are the summation of the whole encoding in dimenstion zero)

Figure 3: applying DOG filter and encodings.



TTFS encoding 2D and 3D plots. (3D details in code.)



TTFS encoding 2D and 3D plots. (3D details in code.)

2.1.2 Gabor Filter

Gabor filters are applied to the image below with varying sizes and parameters. Gabor filters are a type of linear filter used for edge detection and texture analysis. The key parameters of Gabor filters include:

Frequency (f or wavelength λ): This determines the scale of the features the filter will detect. A higher frequency corresponds to finer details, while a lower frequency corresponds to coarser features.

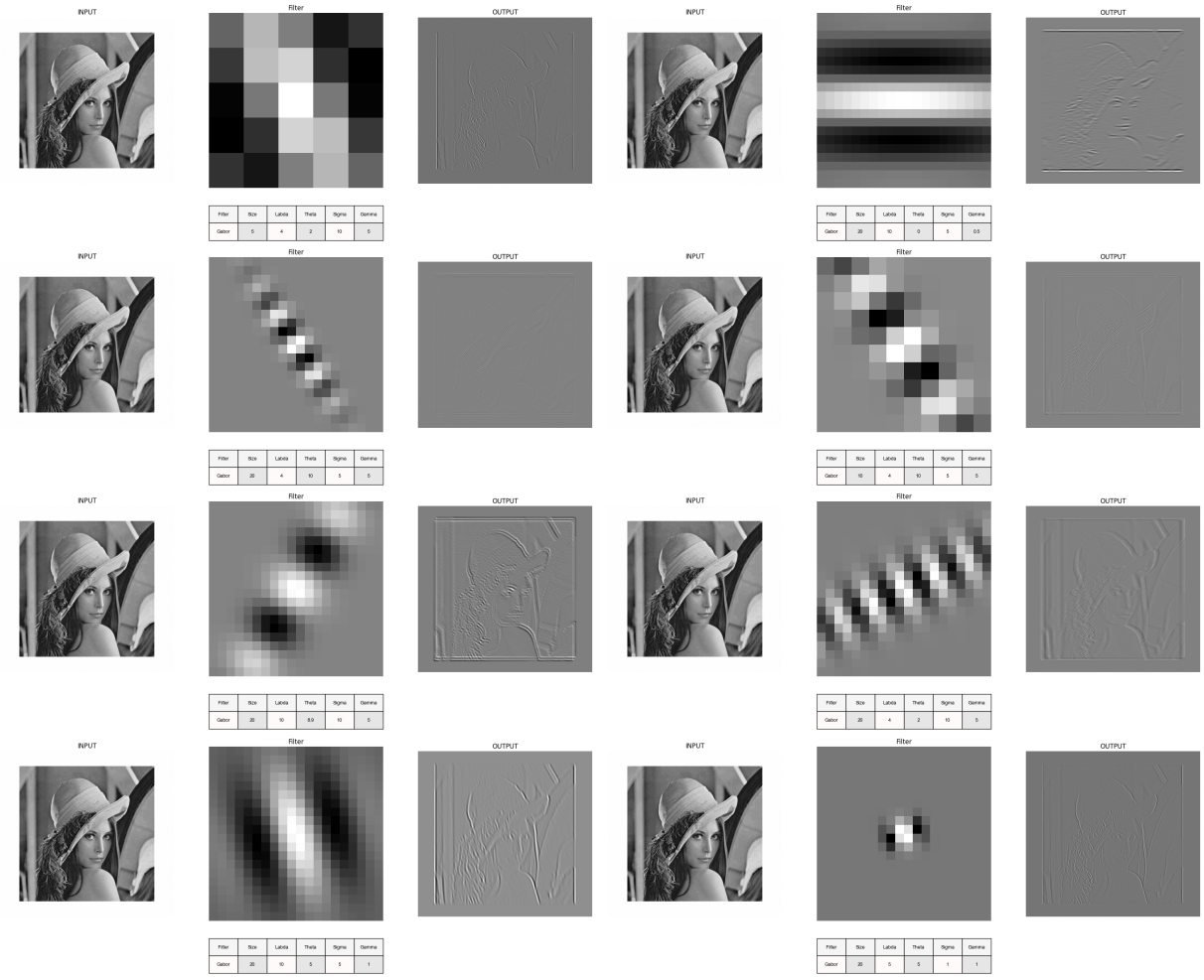
Orientation (θ): This defines the direction of the filter. By adjusting the orientation, Gabor filters can detect edges or textures at specific angles, such as horizontal, vertical, or diagonal edges.

Standard deviation of the Gaussian envelope (σ): This controls the extent of the Gaussian window over which the filter operates. A larger σ results in a wider Gaussian envelope, which can detect broader features, while a smaller focuses on finer details.

Aspect ratio (γ): This parameter defines the elongation of the Gaussian window. An aspect ratio of 1 results in a circular Gaussian window, while values less than 1 result in an elongated window, which can be useful for detecting oriented features.

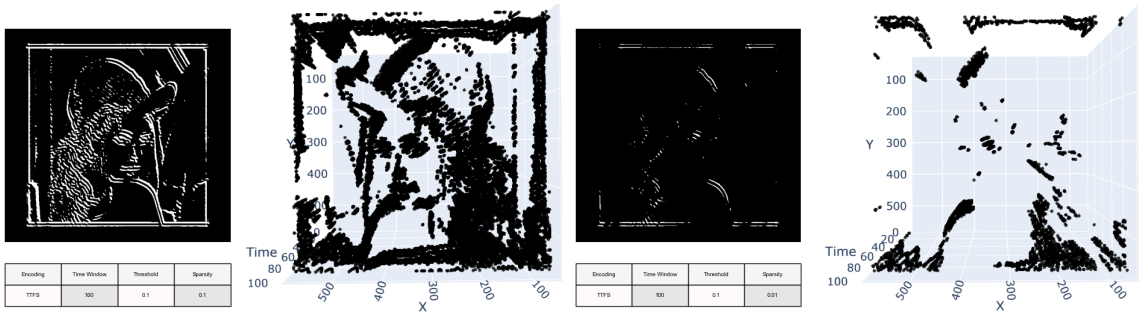
In this example, Gabor filters with different frequencies and orientations are applied to the image.

Figure 4: applying Gabor filter to an image.

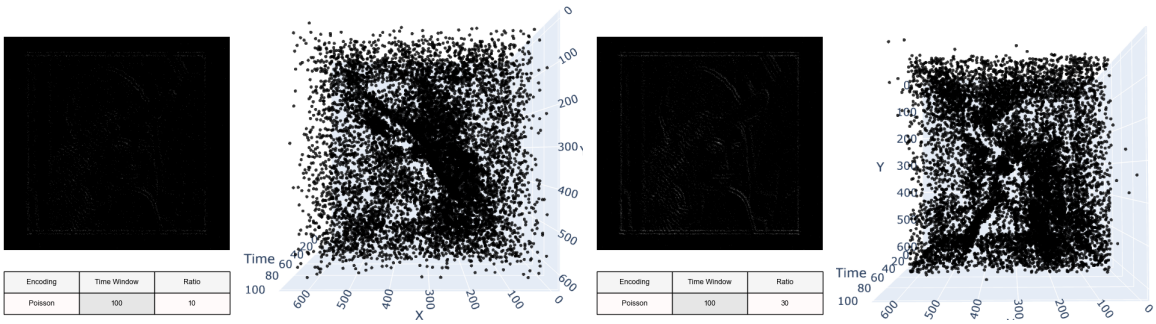


As we can see in both figures 4 and 2, TTFS encoding assigns a spike timing to each pixel based on its intensity, with brighter pixels spiking earlier than darker ones. This encoding method emphasizes the most salient features of the image by prioritizing information based on pixel intensity. It is observable that lowering the sparsity parameter in TTFS, leads to detecting brighter edges. Poisson encoding, on the other hand, converts pixel intensities into spike trains where the firing rate of each pixel follows a Poisson distribution proportional to its intensity.

Figure 5: applying Gabor filter and encodings.



TTFS encoding 2D and 3D plots. (3D details in code.)



TTFS encoding 2D and 3D plots. (3D details in code.)

2.2 Part2

The dataset that have been used in this part is CelebA dataset which consists of several human faces. I resized these images to 100*100 and encoded them using parameters mentioned in the tables.

2.2.1 Extracting Small Features

In these experiments, our kernel size is relatively small compared to the image size. As a result, we anticipate that the extracted features will encompass edges, lines, and dark points against bright backgrounds, as well as light points against dark backgrounds.

Figure 6: Small features extracted from number of faces (poisson encoding).

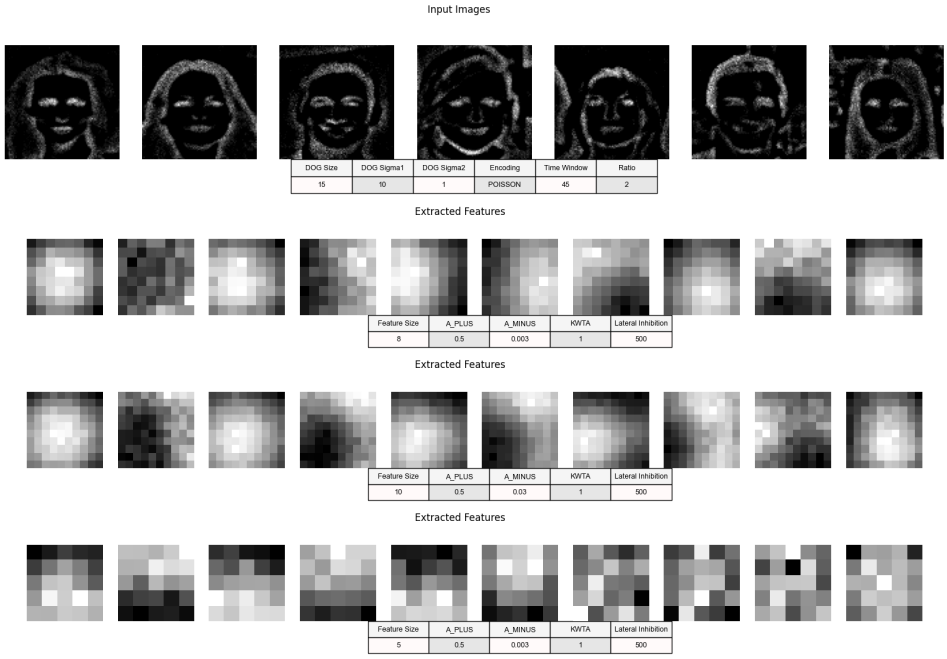
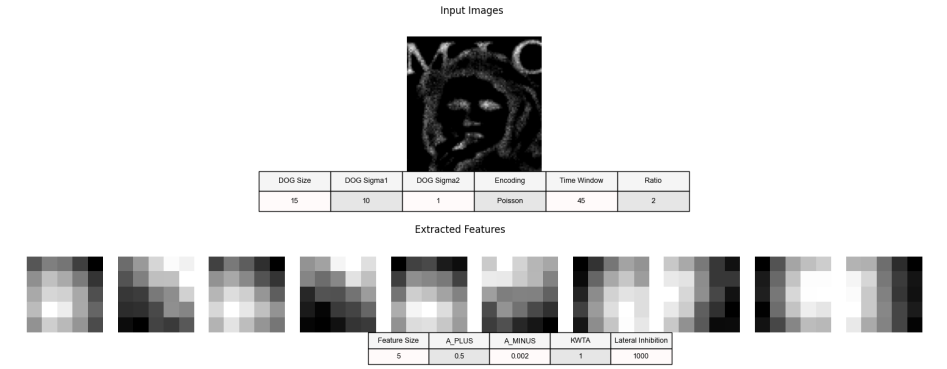


Figure 7: Small features extracted from one face (poisson encoding).



2.2.2 Extracting Medium Features

In these experiments, our kernel size is about one third of the image size. As a result, we anticipate that the extracted features will include nose, ears, eyes and human face features.

Figure 8: Medium features extracted from one face (poisson encoding).

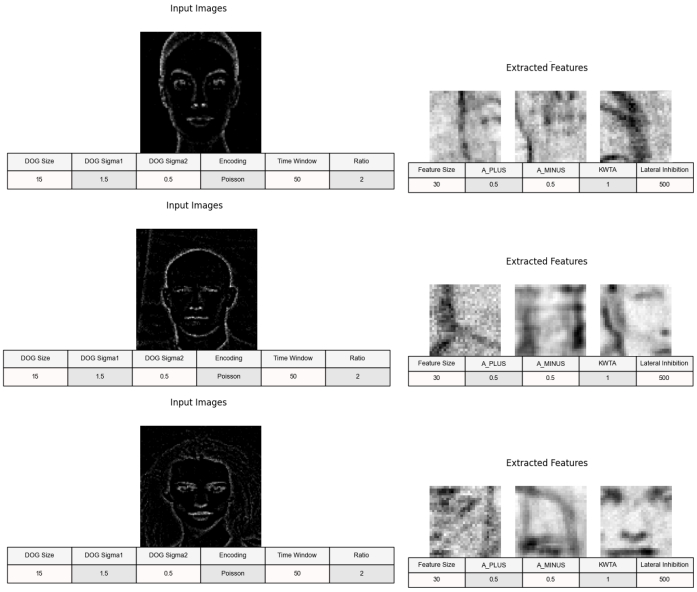
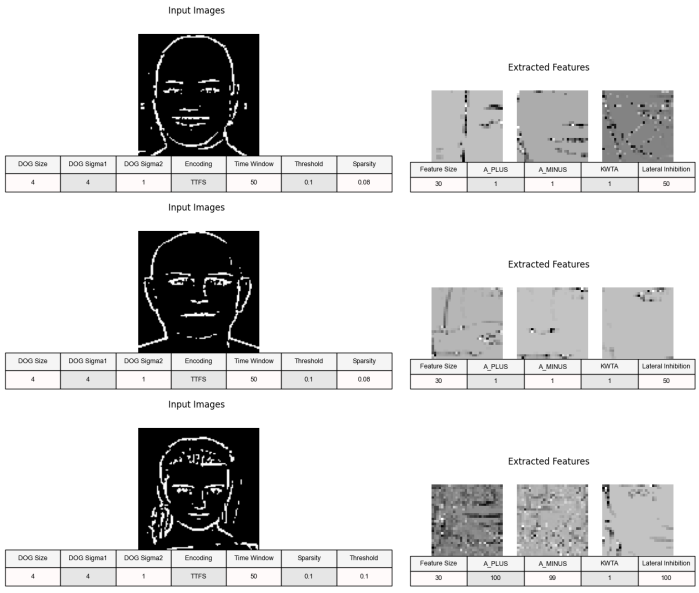


Figure 9: Medium features extracted from one face (ttfs encoding).



When using a large number of pictures, the most frequent 30*30 feature is the box including eyes and nose, and the box including lips and chin, this can be seen in figures 10 and 11 perfectly.

Figure 10: Medium features extracted from number of faces (poisson encoding).

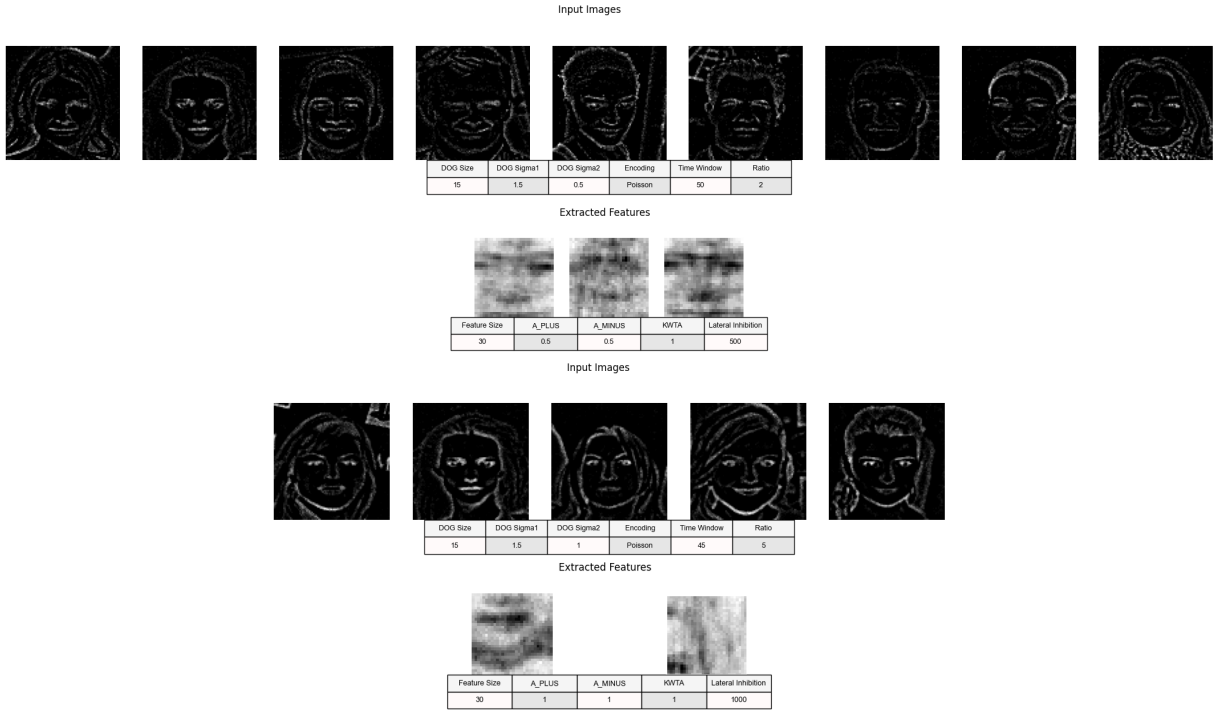
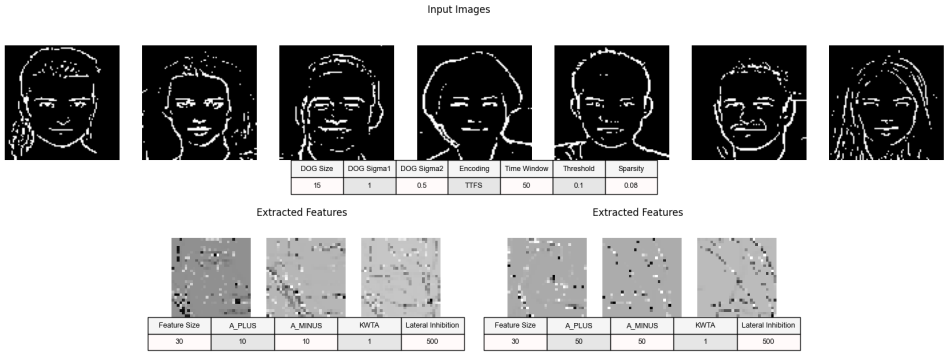


Figure 11: Medium features extracted from number of faces (ttfs encoding).



2.2.3 Extracting Faces

When we increase the filter size (or kernel size), we anticipate that the neural network will learn more global features. In the context of face detection, this might mean capturing the entire face as a frequent feature. Larger filters have a broader receptive field, allowing them to capture more context. However, we also want to avoid overfitting. If the extracted face becomes too similar to the input images, it might not generalize well to unseen data. By adding homeostasis and reducing learning rates, we aim for stability. Ideally, each neuron group becomes sensitive to specific patterns (like individual faces) rather than being overly influenced by noise or irrelevant features.

Figure 12: Large features extracted from one face (poisson encoding).

Figure 13: Large features extracted from one face (ttfs encoding).

Figure 14: Large features extracted from number of faces (poisson encoding).

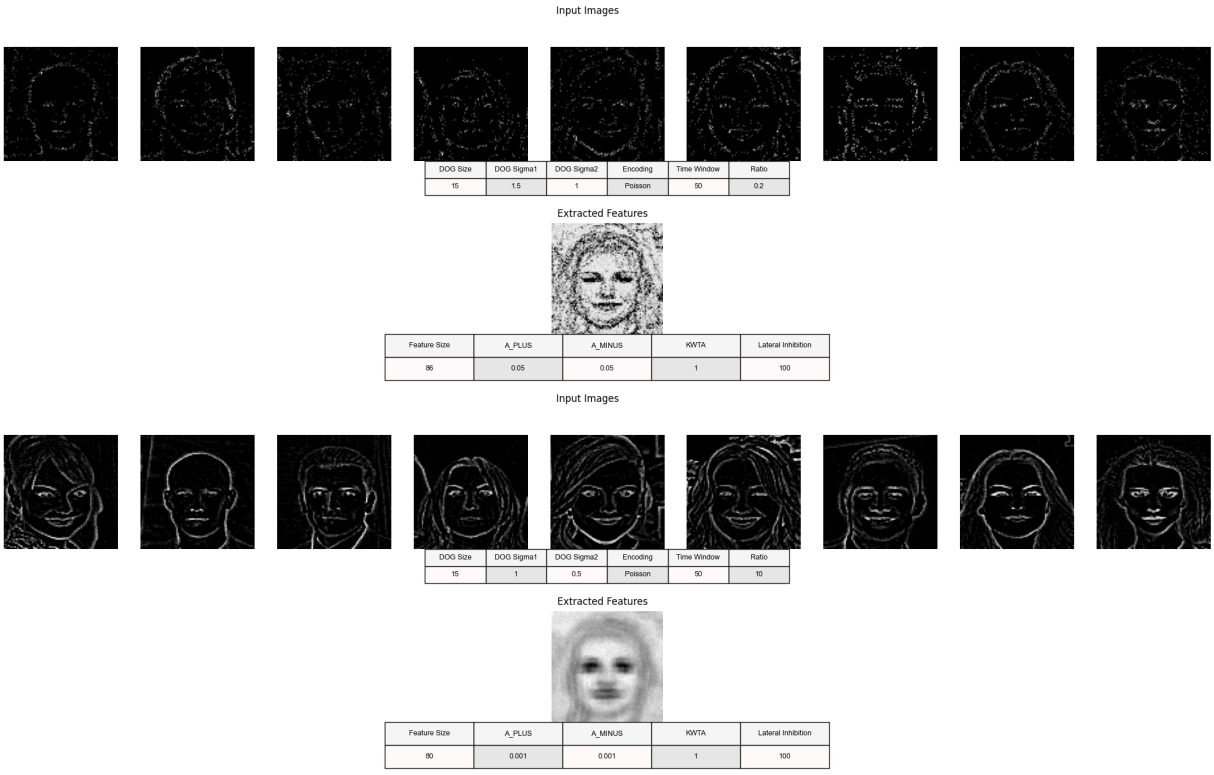
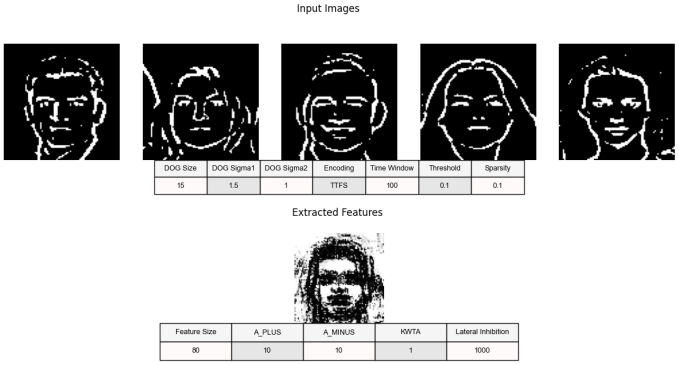


Figure 15: Large features extracted from number of faces (ttfs encoding).



As mentioned previously, adding homeostasis results in difference neuron groups reacting to different input images.

Figure 16: Large features extracted from number of faces (poisson encoding+Homeostasis).