ZUMMIT
INFO LABS
INVENT YOUR FUTURE TODAY

Computer Vision: Blurred image detection

Project #2 - Group-1

Yulia, Ruba, Jay, Dharmaraj

Computer vision: Blurred Image detection

❖ INTRODUCTION:

The technique used in this project is Computer vision - Computer vision enable computers to interpret and understand visual information from the world, such as images and videos. This can include tasks such as object recognition, image segmentation, and scene understanding. Computer vision algorithms can be used in a wide range of applications, including self-driving cars, security systems, and image-based search engines.

❖ OBJECTIVE:

The objective of computer vision in blurr image detection is to detect and classify images that are blurred or out-of-focus. This can be useful in a variety of applications, such as image and video processing, surveillance systems, and digital image enhancement

❖ GOAL:

The goal is to develop algorithms that can accurately detect and classify blurred images. (in order to improve image quality or to automatically discard low-quality images)

Computer vision: Blurred Image detection

❖ Data description:

The dataset consists of images, and some of them are blurred. The images are blurred using augmentation.

Augmentation : Image augmentation is the process of creating new images that can be used to improve the performance of machine learning models in computer vision, by artificially increasing the size and diversity of the dataset.

Data files:

i)train – a folder with images for Training

ii)test – a folder with images for Predicting/Testing

iii)train.csv – labels (answers) to the train sample: if 1, the image is blurred.

❖ SOURCES:

* Hsu, P., & Chen, B. Y. (2008, January). Blurred image detection and classification. In International conference on multimedia modeling (pp. 277-286). Springer, Berlin, Heidelberg.

* Su, B., Lu, S., & Tan, C. L. (2011, November). Blurred image region detection and classification. In Proceedings of the 19th ACM international conference on Multimedia (pp. 1397-1400).

* Liu, R., Li, Z., & Jia, J. (2008, June). Image partial blur detection and classification. In 2008 IEEE conference on computer vision and pattern recognition (pp. 1-8). IEEE.

Computer vision: Blurred Image detection

❖ Project Methodology:

1.Data collection: Gathering a dataset of both blurred and non-blurred images. This dataset will be used to train and test the computer vision model.

2.Data pre-processing: Perform necessary pre-processing steps on the dataset such as image resizing, normalization and selection of Batch size.

3.Using Laplacian Algorithm for Blurred Image Detection: The Laplacian algorithm is a method for detecting blur in an image. The basic idea is that a blurred image will have a low-frequency content, while a non-blurred image will have a higher frequency content.

4.Using CNN Algorithm: Defining the function for loading the data, initiating, training a model and Training with different CNN's

5.Model evaluation: Evaluate the performance of the trained model using metrics such as accuracy

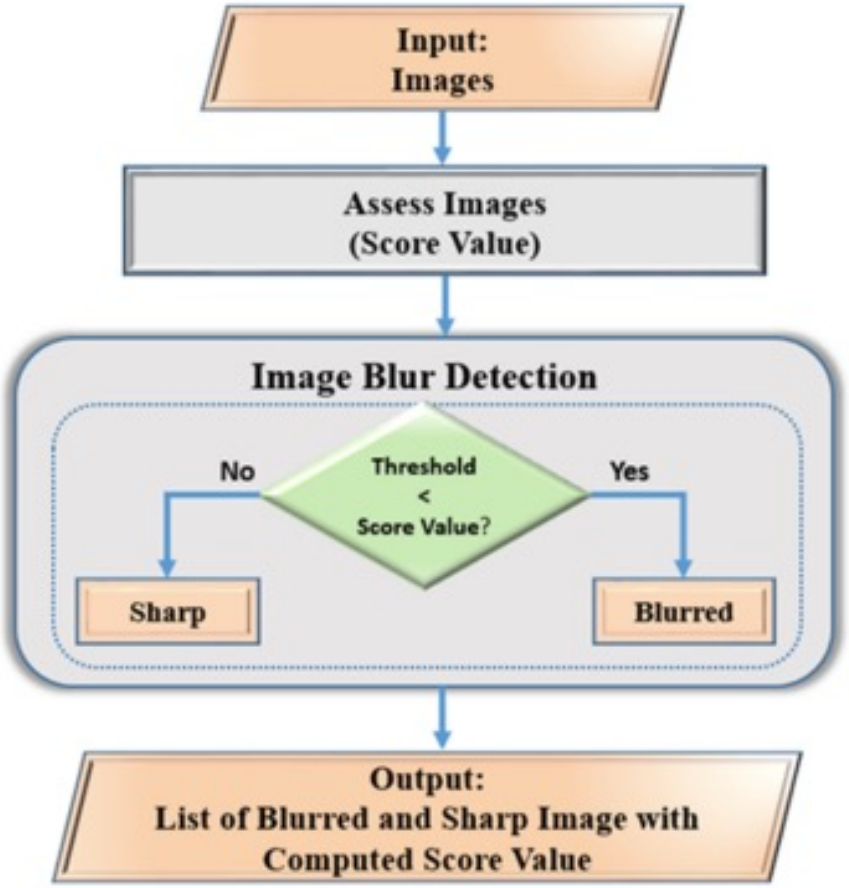
6.Model fine-tuning: Based on our evaluation results, fine-tune the model by adjusting the hyperparameters

7.Comparing the Models: Comparing Laplacian Algorithm and CNN.

Laplacian vs CNN

Flowchart of Image Blur Detection Techniques Processes

- Fast Fourier Transform (FFT)
- HaarWavelet Transform (HWT)
- Modified Laplacian (MLAP)
- Tenengrad (TEN)
- **Laplacian Operator (LAP)**



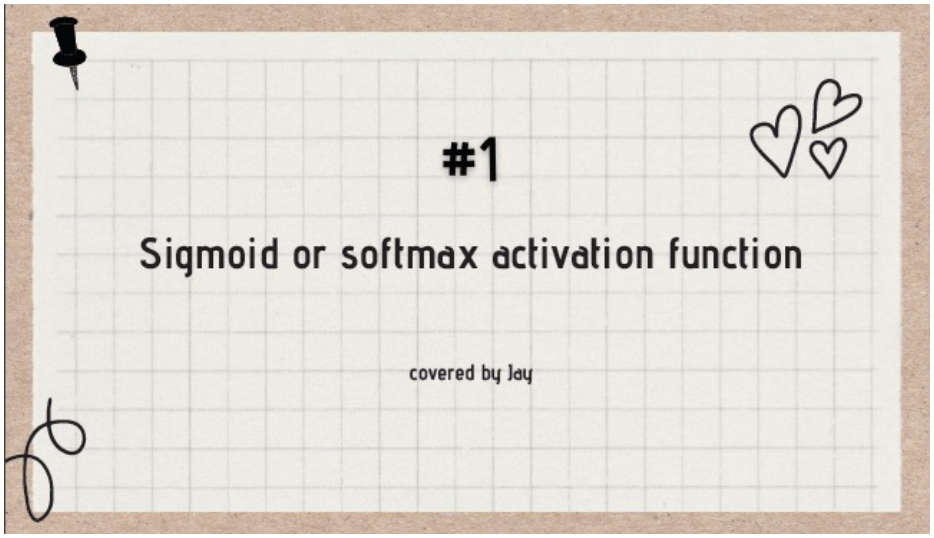
LAP got the highest recall score

Pech-Pacheco et al, 2000; Abdel-Qader et al, 2003; Ali and Mahmood, 2018

CNNs

1) Laplacian is too subjective
2) We trained 5 different CNNs
3) Jay will tell you the details about the CNN architecture
4) Ruba will tell you about the best CNN we chose and why

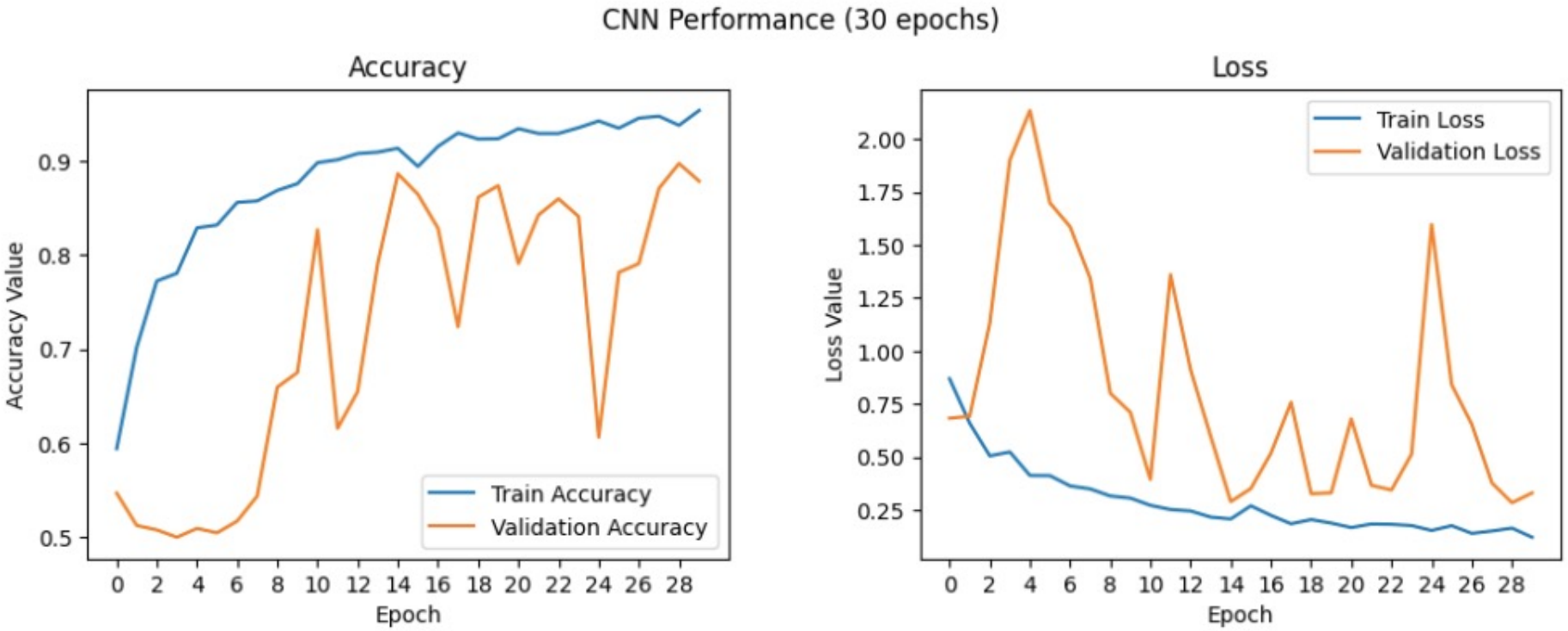
CNN architecture details



- Model 5

In [104]:
```
def create_model_5(input_shape):
    model = Sequential()
    model.add(Conv2D(32, (3, 3), padding='same', input_shape=input_shape))
    model.add(Activation('relu'))
    model.add(Conv2D(32, (3, 3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Dropout(0.25)) # regularization

    model.add(BatchNormalization()) # added
#     model.add(Conv2D(64, (3, 3), padding='same'))
#     model.add(Activation('relu'))
    model.add(Conv2D(64, (3, 3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Dropout(0.25)) # regularization

    model.add(BatchNormalization()) # added
    model.add(Flatten())
    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer='Adamax',
                  loss="binary_crossentropy", metrics=["accuracy"])
    return model
```

The best CNN architecture



Final conclusion

ZUMMIT
INFO LABS
INVENT YOUR FUTURE TODAY

Thank you!