Effect of converting 3d to 2d image and applying KAZE feature descriptor on CNN

PROJECT BY:

ABHIJIT TALLURI (11527014)
KAVYA SRI ACHANTA (11591814)

VENKATA NAGA PRASANNA KUMAR BITRA (11525657)
PRASANTH DAMARLA (11525662)

Introduction

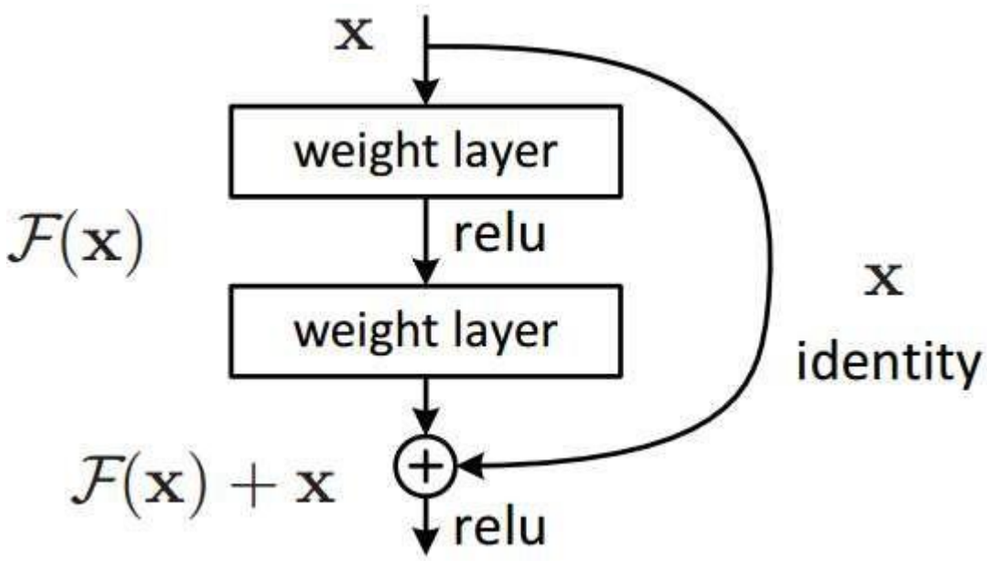
This project is mainly concentrated on the effects of converting 3D to 2D image and applying the KAZE feature descriptor on the CNN model. KAZE feature is a 2D feature detectionand the description method that works completely on the nonlinear scale space. The 3D imageis converted into 2D image by matrix summation without using the gray scale. Which will help us to get better results. And the model is trained using the prediction model. For the 3D images, we are performing the summation for the colors (RGB) to convert the original image to 2D image. Then KAZE is applied on the 3D image. The KAZE will extract the key Features of the image. Then our main task is to combine 2D image and the key features as a single image.

Background

- **Related work for your topic with linked references -** This project is our own through to developed on the Residual Network. It will give you better speed andaccuracy. The Residual network is designed for the deterioration problem. Which makes frames much easier to train the network than the previous frameworks. These layers are like the CNN (Convolutional Neural Networks). There are a few levels to reduce the framework depth. The main aim is to develop a project to convert the 3D image to 2D image by summation of the RBG then apply the KAZE on the 3D image to extract the main key features, combining the 2D image with the features extracted.

Your Model

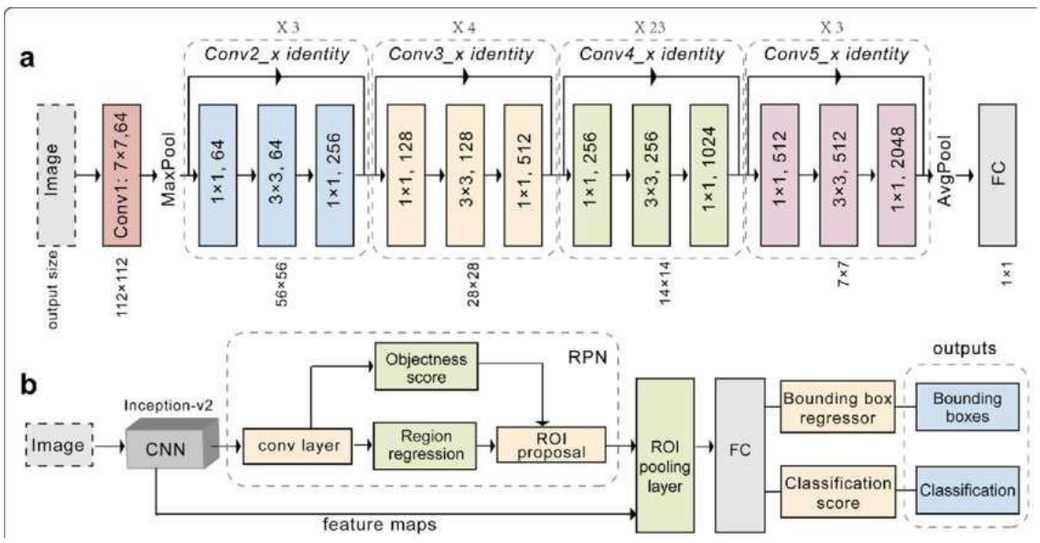
- **Architecture Diagram with explanation**



RESNET is used to improve the accuracy of the model and to overcome the drawbacks in the CNN (convolution network). In RESNET it takes the image as the input and sends the feedbackto the processed image. The f(x) is the output of the CNN for the original image x the output ofthe RESNET is h(x) which is f(x) + x.

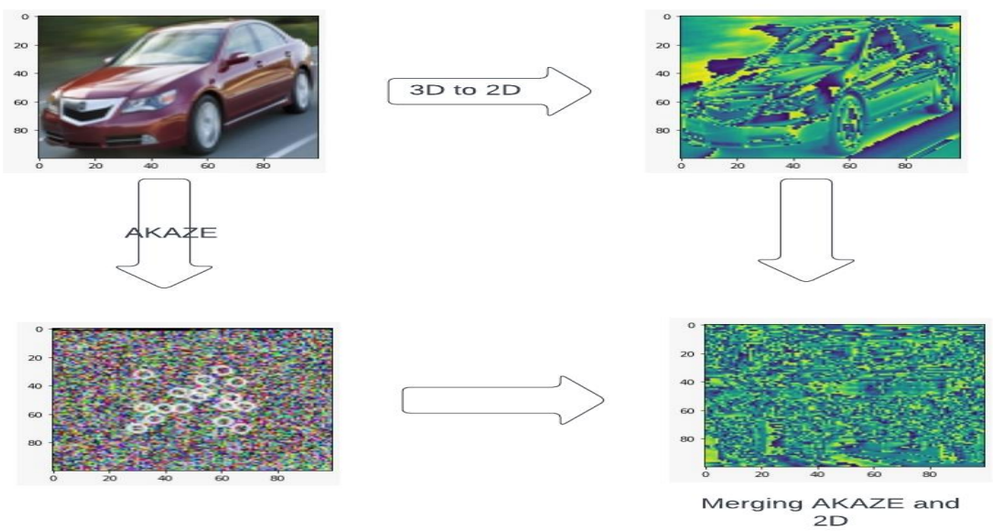
ResNet 50 is a neural network architecture that was developed as part of the Google Brain project. It is a type of Convolutional Neural Network (CNN) that has been trained on the ImageNet dataset, and this is what makes it so powerful. Resnet 50 is the latest and most advanced deep neural network architecture. It was designed to be faster, more accurate and more efficient than previous neural networks.

It can produce better results with less data, making it a perfect candidate for training in low-data environments.



The above is the workflow diagram of RESNET.

- **Workflow diagram with explanation**



Workflow diagram

Workflow for building resnet50 model

1. import the resnet50 model with ImageNet weights

2. use the resenet50 preprocess input function to preprocess the image

3. generating the train and test data with custom preprocessing function

4. changing the input layer of reset to work with a single channel image

5. creating the full model

7. compiling the model

8. fitting the model

Workflow for the building resnet50 model with AKAZE feature detection and changing the image to single channel

1. importing the resnet50 model with ImageNet weights

2. defining the custom preprocessing function for the resnet50 model applying AKAZE feature detection and matching and changing the image to single channel

    - Appling Transformations to be applied to the images: -

single channel image: here we are converting the image to single channel by taking the matrix addition of the three channels of the image and then we are normalizing the image for better results. Applying AKAZE feature detection: here we are applying AKAZE feature detection and matching to the image and then we are drawing the key points on the image and then we are converting the image to single channel

3. generating the train and test data with custom preprocessing function

4. changing the input layer of reset to work with a single channel image

5. creating the full model

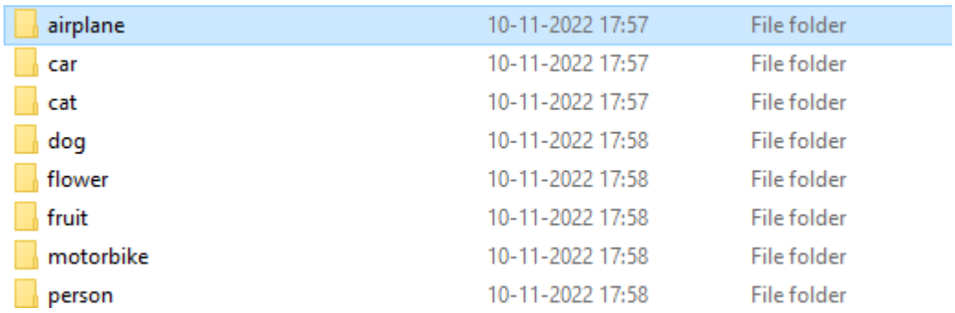
7. compiling the model

8. fitting the model

Dataset

- Detailed description of Dataset

In the project we are using the dataset from Kaggle. The dataset contains the airplane, car, cat, dog, flower, fruits, motorbike and person images. Using this dataset, we can train the model to get the accuracy. We are down sampling the dataset because the dataset size is very huge. And preparing our own dataset.



Dataset of the cars


motorbike_0000.jpg


motorbike_0001.jpg


motorbike_0002.jpg


motorbike_0003.jpg


motorbike_0004.jpg


motorbike_0005.jpg


motorbike_0006.jpg


motorbike_0007.jpg


motorbike_0011.jpg


motorbike_0012.jpg


motorbike_0013.jpg


motorbike_0014.jpg


motorbike_0015.jpg


motorbike_0016.jpg


motorbike_0017.jpg


motorbike_0018.jpg


motorbike_0022.jpg


motorbike_0023.jpg


motorbike_0024.jpg


motorbike_0025.jpg


motorbike_0026.jpg


motorbike_0027.jpg


motorbike_0028.jpg


motorbike_0029.jpg


motorbike_0033.jpg


motorbike_0034.jpg


motorbike_0035.jpg


motorbike_0036.jpg


motorbike_0037.jpg


motorbike_0038.jpg

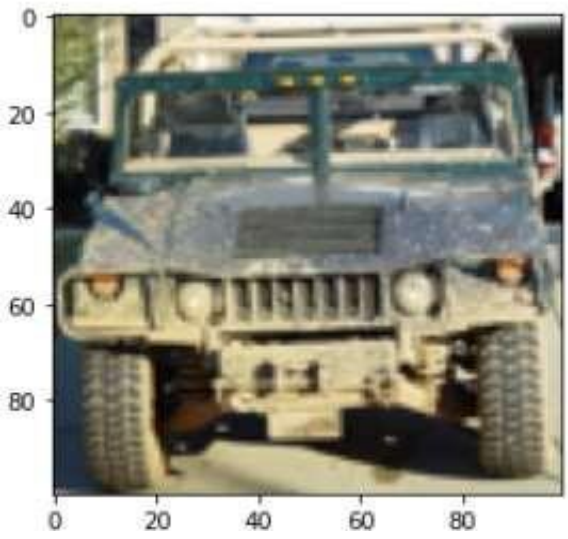

motorbike_0039.jpg


motorbike_0040.jpg

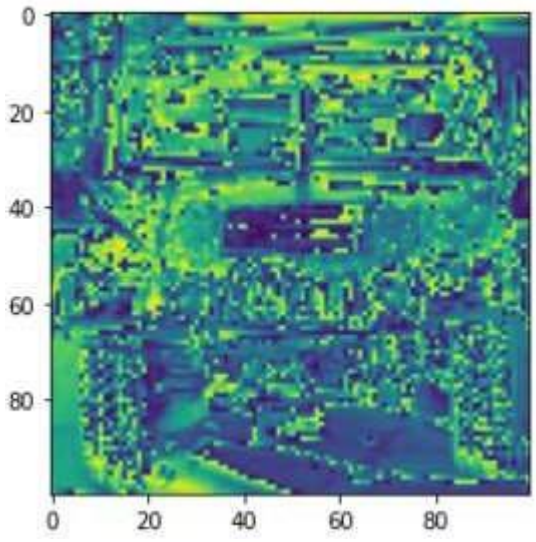
- ## Detail design of Features with diagram

1. **KAZE**: The technique identifies and characterizes 2D properties in a non-linear scalespace to increase separation and placement accuracy. A non-linear scale space is constructed from an input image by discretizing its scale space in a series of octaves and sublevels while keeping the original image's resolution. Then, from pixels, translate the collection of discrete scale levels in time units I. The non-linear scale space is then built iteratively using additive operator splitting (AOS) and variable conductance diffusion techniques to reduce noise and retain the borders of the objects in the image.

To find 2D features, the scaled normalization determinant of a Hessian matrix, producedat various scale levels with maxima as important aspects, is employed. By figuring out thekey point orientation, a scale- and rotation-invariant descriptor is produced, and this serves as the description. The computational cost of this approach is high.



Original image



KAZE filter

2. AKAZE – It is the extension of the KAZE. A feature extractor is a machine learning algorithm that extracts features from the input data. AKAZE is a Python library to create a feature extractor. AKAZE's algorithms can identify objects in images, detect faces, recognize text from images and videos, extract metadata from videos and detect emotions from images. It can be used to create a simple or complex feature extractor, depending on the user's needs.

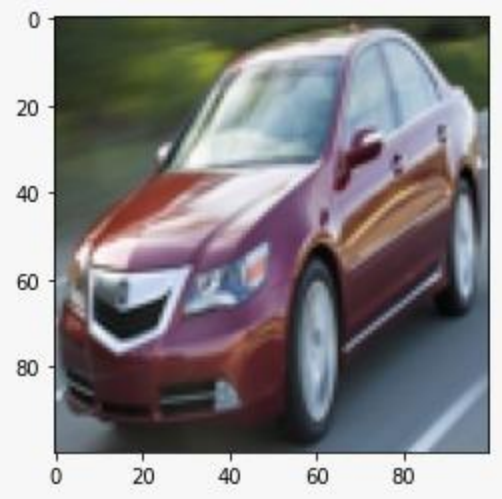
AKAZE can be used in various applications such as:

- Feature extraction in data mining
- Feature extraction for machine learning models
- Feature extraction for deep neural networks

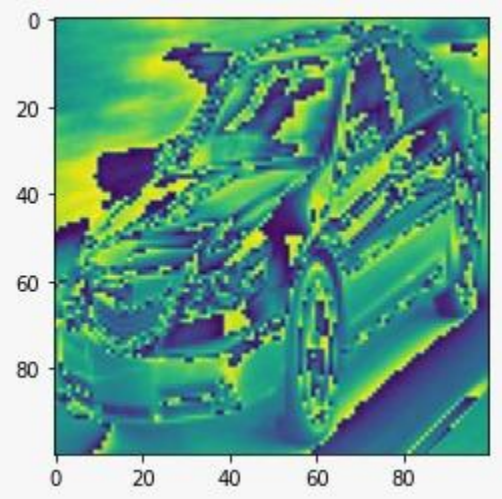
AKAZE is a feature extractor that has been designed to help people with complex data problems.

AKAZE Feature Extractors are the best way to find out which features are the most important for a given dataset or problem. It can be used for many different purposes, such as finding the best features for classification, regression and clustering.

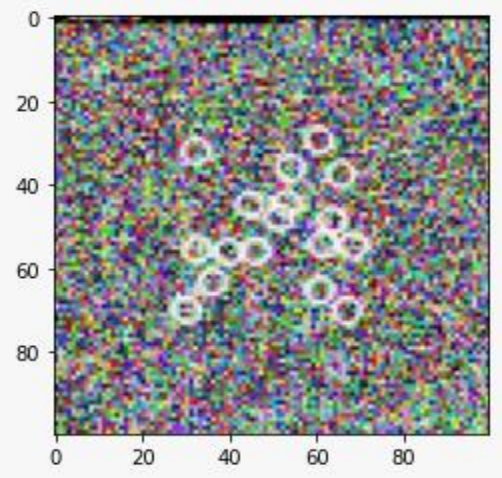
This tool is not limited to only extracting features from raw data, it can also be used to generate new features in order to improve the performance of machine learning models.



Original Image



KAZE Image



AKAZE Image

Analysis of data

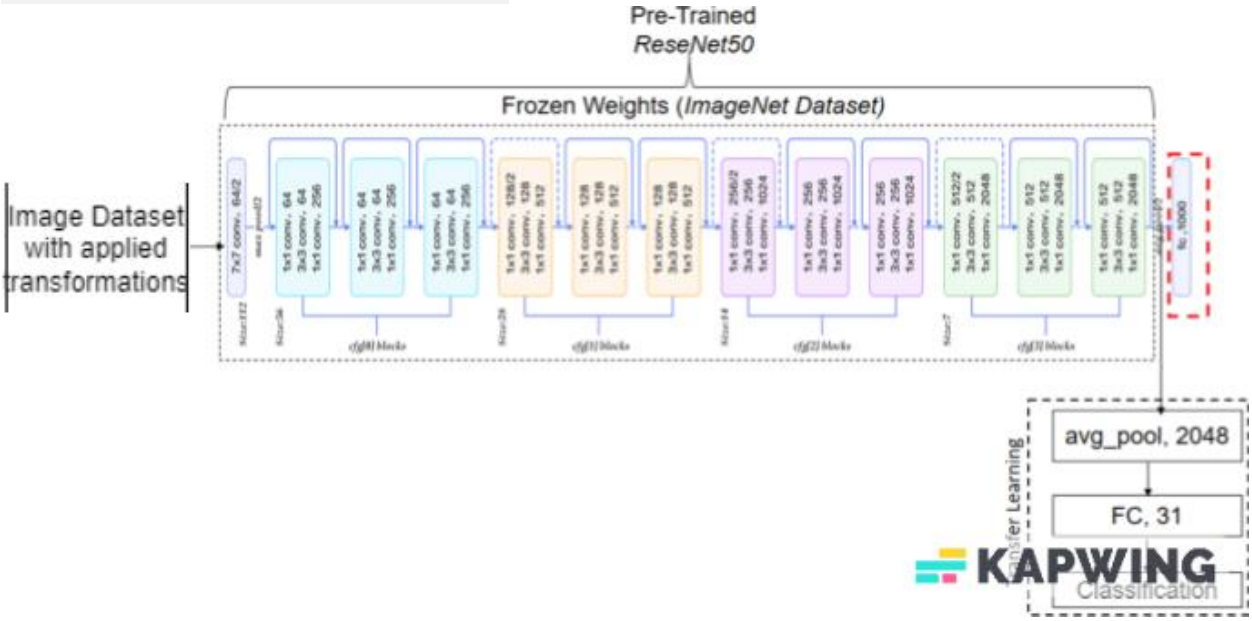
- ## Data Pre-processing

  In the Data preprocessing we are using AKAZE and converting the 3D image to 2D image.

  - **Converting to 2D** – 3D images are converted to 2D images by summation of the RGB. single channel image benefits are using a single channel image helps the model to learn the features better and the total number of parameters is reduced to 23.5M from 25.6M in the original model and majority of the parameters are in the dense layers which are not affected by the change in the input layer of the model and while coming to the convolutional part the model is able to learn the features better and the accuracy is also increased by 2% and there is a lot less filters to be trained in the convolutional part of the model

  - **AKAZE** – It is used to extract the main key features of the image. The key feature will be the point we need to concentrate on. Benefits of using AKAZE feature detection in the preprocessing step are that it is able to detect and match features in the images and the model is able to learn the features of the images and classify them better than the model without AKAZE feature detection.

- ## Graph model with explanation



Implementation

- ## Algorithms / Pseudocode

1) Reading the image and Converting into 2D and Normalizatio

```
def single_channel_image(img):
    (B, G, R) = cv2.split(img)
    merged_channel = B+G+R
    normalized_channel_image = cv2.normalize(merged_channel, None, alpha=0, beta=255, norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8U)
    return normalized_channel_image
```

B. Appling AKAZE

```
def apply_keypoints(img):
akaze=cv2.AKAZE_create()
queryKeypoints, queryDescriptors = akaze.detectAndCompute(img,None)
empty_img = np.empty(img.shape, dtype=np.uint8)
keypoints_img = cv2.drawKeypoints(empty_img, queryKeypoints, outImage = None, color=(255, 255, 255), flags=0)
return keypoints_img
```

C. Combining 2D and AKAZE

```
def apply_image_transormations(img):
return single_channel_image(img) + single_channel_image(apply_keypoints(img))
```

2) Applying RESTNET model on the dataset. resnet50 = keras.applications.resnet50
conv_model = resnet50.ResNet50(weights='imagenet', include_top=False)conv_model.summary()

3) Printing the accuracy of the filter and adjusting the epochs accordingly
Full_model.fit_generator(train,validation,workers,epochs)
Full_model.predict()

4) Appling the KAZE filter on the whole dataset.

```
def preprocess_input(img): #appling on the dataset
img = np.array(img, dtype=np.uint8)
query_img = single_channel_image(img)
query_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)akaze=cv2.AKAZE_create()
queryKeypoints, queryDescriptors = akaze.detectAndCompute(query_img,None) img = cv2.drawKeypoints(img,queryKeypoints,outImage = None,color=(255,0,0),
```

```
flags=0)
    return resnet50.preprocess_input(img)
```

- ## Explanation of implementation

  This project involves two phases as mentioned below:

  Phase 1: This phase involves building resnet 50 model. Now, I am going to explain the workflow for building this model.

  To begin with, we import the resnet50 model with ImageNet weights and use this model to pre-process the input function which is an image. Further, we generate the train and test data with custom pre-processing function. To move on, we change the input layer of resnet to work with a single channel image and create the full model. Finally, we compile this model and fit it.

  Phase 2: This phase involves building the resnet50 model with AKAZE feature detection and changing the image to single channel. Hereafter, I am going to explain the workflow of this process.

  Firstly, we import the resnet50 model with ImageNet weights and define the custom pre-processing function for the resnet50 model by applying AKAZE feature detection and matching and changing the image to single channel. Secondly, we apply transformations to the images.
  Single channel image: Here we convert the image to single channel by taking the matrix addition of the three channels of the image and then normalizing the image for better results.

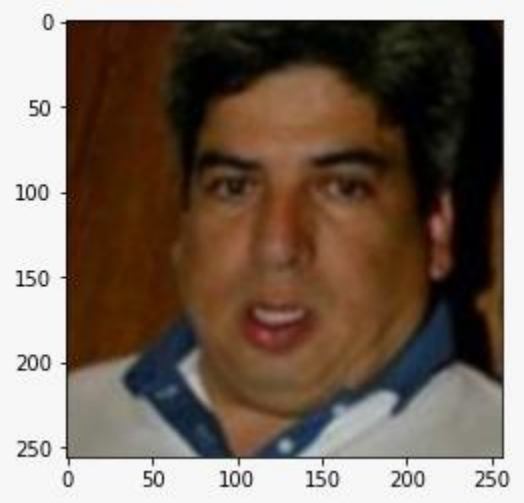
  Thirdly, applying AKAZE feature detection: Here we apply AKAZE feature detection and match the image and then draw the key points on the image. These images are further converted to a single channel. In addition, we generate the train and test data with custom pre-processing function and change the input layer of resnet to work with a single channel image. In the end, we create, compile and fit the full model.

  Here in the following step we are comparing the performance of the above two models on the training and test data.

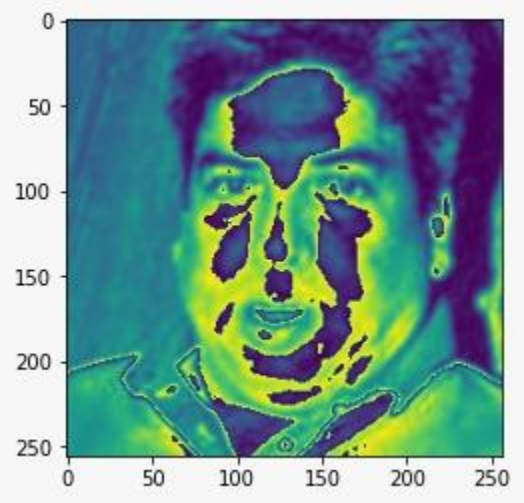
Results

- Diagrams for results with detailed explanation

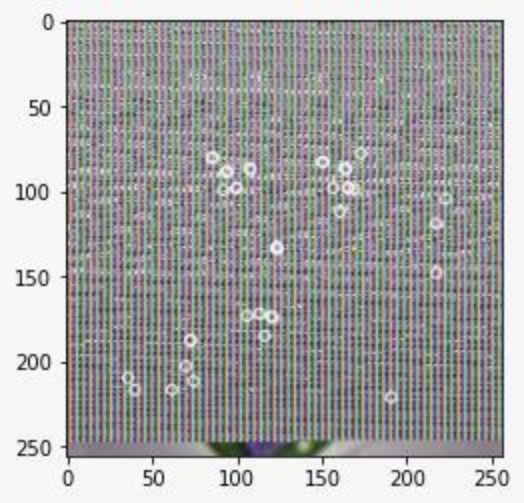
Comparing the two models, we find that the modified resnet50 model outperforms the original resnet50 model with custom preprocessing function, AKAZE feature detection and matching applied to the images, changing the image to single channel, applying the preprocessing function of the resnet50 model to the image, and then training the model with the new images. Here Observing the epochs, loss, and accuracy, we can see that the model is converging more quickly and accurately than the model with the original resnet50 model. Based on the graphs below, we can support the aforementioned assertion and draw the conclusion that the model performs better when the suggested modifications are applied to the resnet50 model.



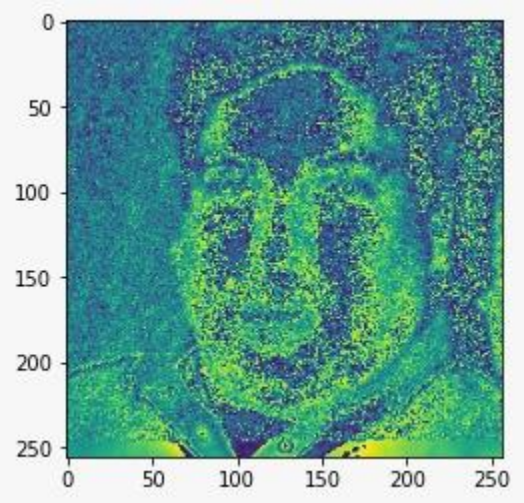
Original Data



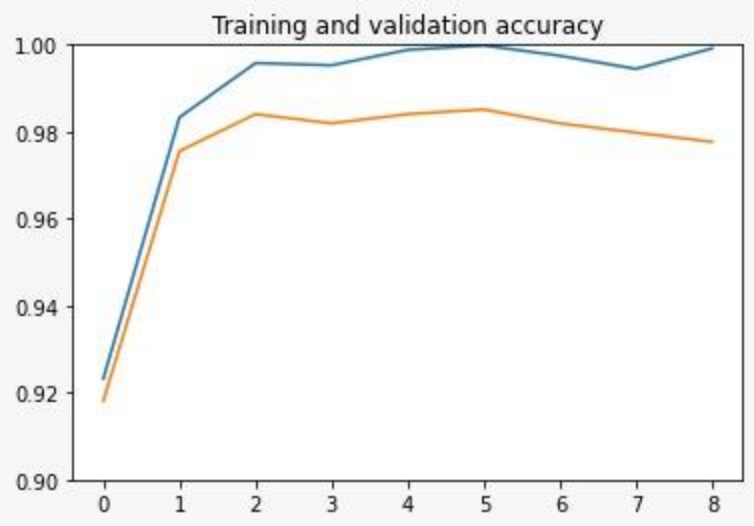
Converted 2D image



AKAZE

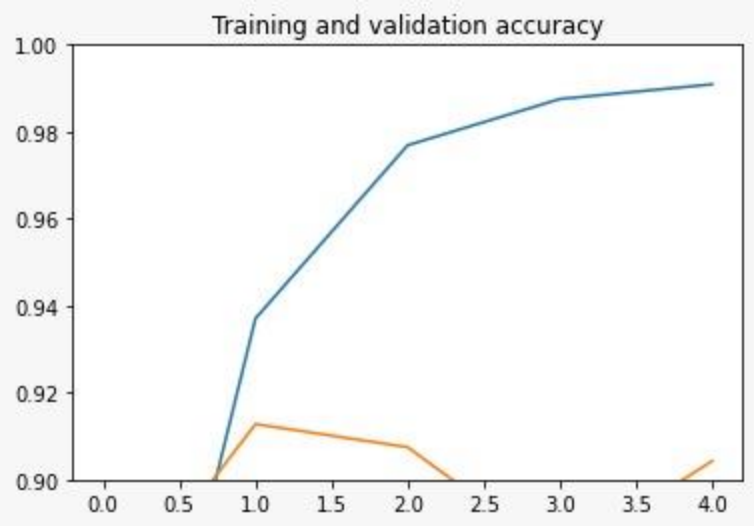


Merged AKAZE and 2D



ReccNet50 Accuracy without Feature Extraction

ReccNet50 validation loss without Feature Extraction



Model Accuracy with Feature Extraction

Model loss with Feature Extraction

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1
1	1.00	1.00	1.00	1
10	1.00	1.00	1.00	1
11	1.00	1.00	1.00	1
12	1.00	1.00	1.00	1
13	1.00	1.00	1.00	1
14	1.00	1.00	1.00	1
15	1.00	1.00	1.00	1
16	1.00	1.00	1.00	1
17	1.00	1.00	1.00	1
18	1.00	1.00	1.00	1
19	1.00	1.00	1.00	1
2	1.00	1.00	1.00	1
20	1.00	1.00	1.00	1
21	1.00	1.00	1.00	1
22	0.00	0.00	0.00	0
23	1.00	1.00	1.00	1
24	1.00	1.00	1.00	1
3	1.00	1.00	1.00	1
4	0.00	0.00	0.00	0
5	1.00	0.50	0.67	2
6	1.00	1.00	1.00	1
7	1.00	1.00	1.00	1
8	1.00	1.00	1.00	1
9	0.00	0.00	0.00	1
accuracy			0.92	24
macro avg	0.88	0.86	0.87	24
weighted avg	0.96	0.92	0.93	24

```
[[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]]
```

Project Management

- Implementation status report
  - Work completed
    - Description - At the end of the increment 2 we are successfully able to convert the image 3D image to 2D Image and apply the KAZE feature and then trained the CNN model to check the results. We have observed that the the accuracy of the model is increased by applying KAZE features.
    - Responsibility (Task, Person)
      - Abhijit Talluri: Worked on the data preproccesing and data cleaning techniques and worked on the KAZE feature and CNN post training
      - Kavya Sri: Worked on the feature extraction of the data and worked along with the abhijith by applying the KAZE feature and train the CNN model.
      - Venkata Naga & Prasanth : Both worked on the CNN model before applying the KAZE feature. And helped to extract features of the data and applied normalization
    - Issues/Concerns
      - The dataset is very huge. Working on the down sampling.

References/Bibliography

https://docs.opencv.org/3.4/dc/d16/tutorial_akaze_tracking.html

https://ieeexplore.ieee.org/document/9778928

https://medium.com/vitalify-asia/create-3d-model-from-a-single-2d-image-in-pytorch-917aca00bb07

Links:-

GitHub – https://github.com/Kavyaachanta0808/fe

Demo Video – https://csce5290.s3.amazonaws.com/fe_project_video.mp4

Presentation Video – https://docs.google.com/presentation/d/1tMAwALOl3dheH0iRDfA-QHN-A7xdiuPD/edit?usp=share_link&ouid=102830962755595745925&rtpof=true&sd=true

Presentation - https://docs.google.com/presentation/d/1tMAwALOl3dheH0iRDfA-QHN-A7xdiuPD/edit?usp=share_link&ouid=102830962755595745925&rtpof=true&sd=true