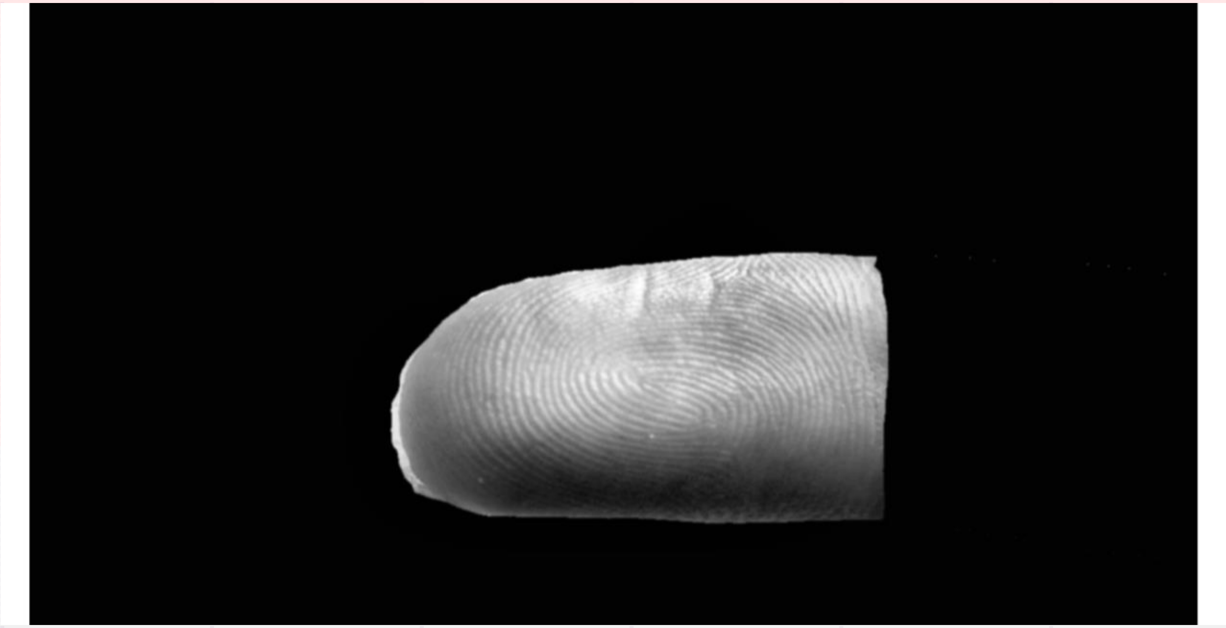
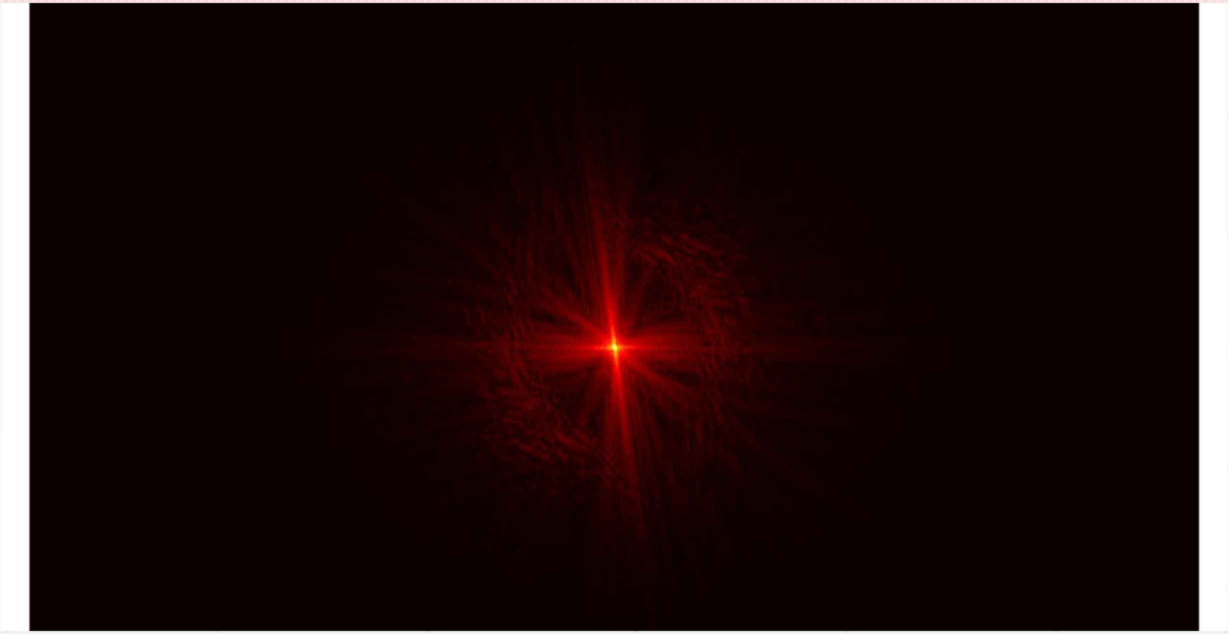
Finger Photo Quality Assessment

6310501933 Wiwitthawin Charoenngam





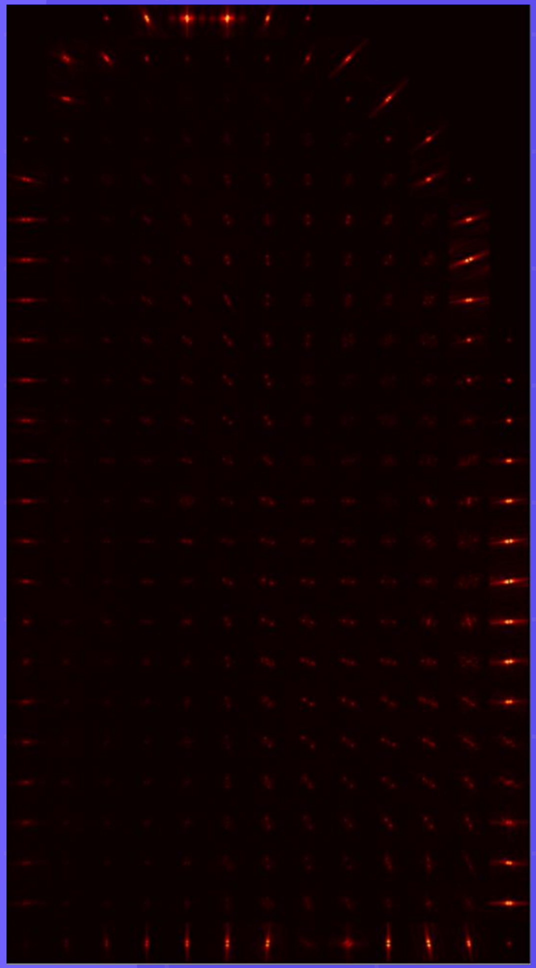
Agenda

- Problem
- Data
- Problem Solving
- Result
- Analysis
- Summary

Problem

How to classify Good Quality Fingerprint Image from Bad Quality?

-> Quality Assessment is crucial for Fingerprint Recognition System for it is to rejected low-quality image that provide unwanted data resulting in less processing time of the overall system.



Data

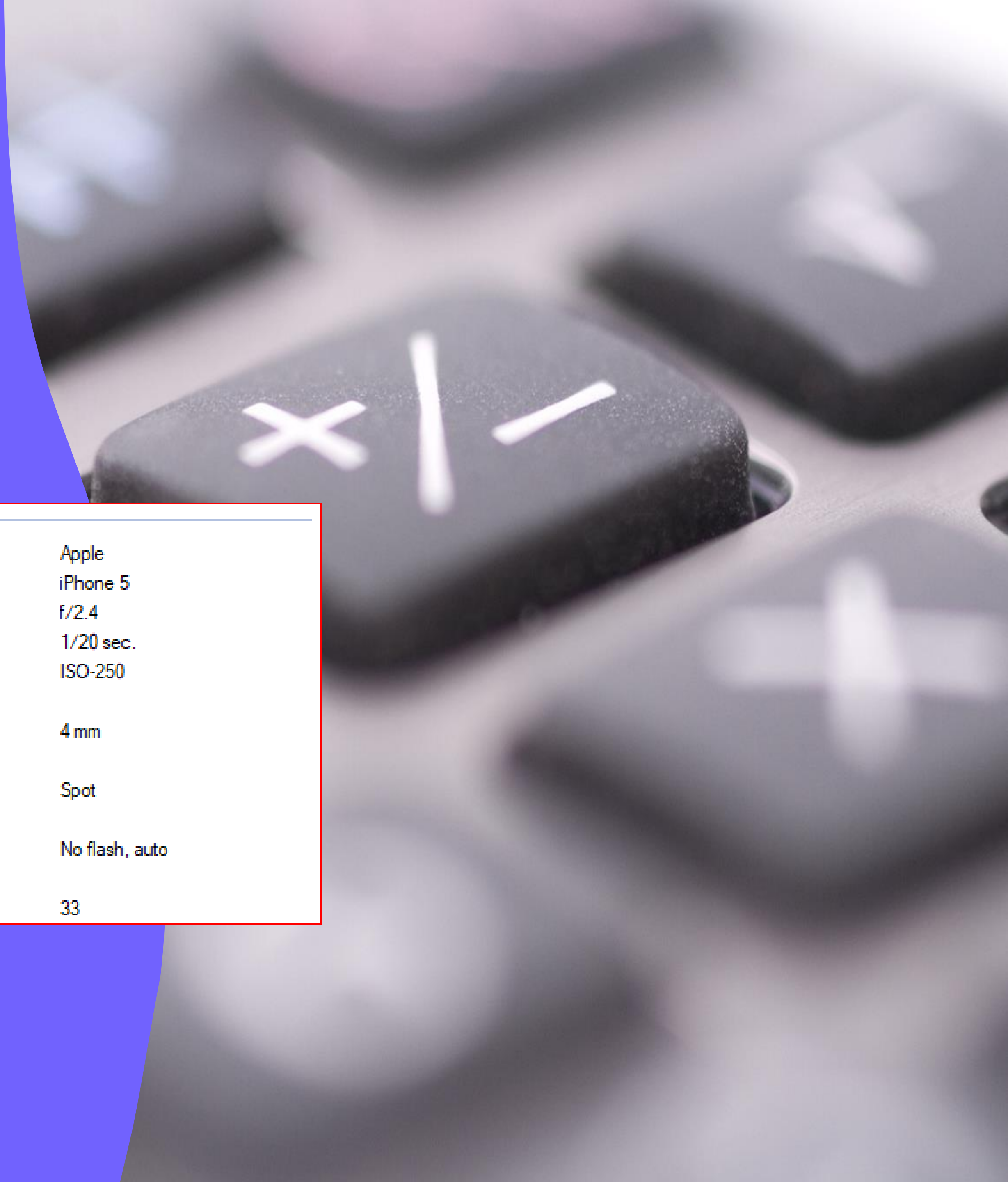
2 Classes: Good quality and Bad quality

Database: 126 images in ISPFDv1 with labelled quality

ISPFD = IIITD Smartphone Finger Photo Database

Camera

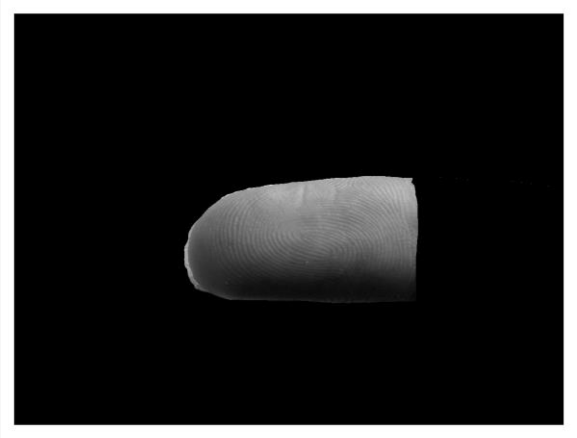
Camera maker	Apple
Camera model	iPhone 5
F-stop	f/2.4
Exposure time	1/20 sec.
ISO speed	ISO-250
Exposure bias	
Focal length	4 mm
Max aperture	
Metering mode	Spot
Subject distance	
Flash mode	No flash, auto
Flash energy	
35mm focal length	33

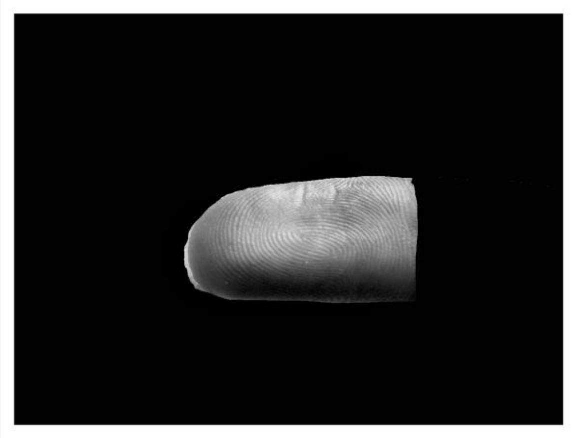


Problem Solving

Preprocessing

Resize by 0.4 → Linear Normalization → Masked with Segmentation Ground Truth → CLAHE (4, (6,6))





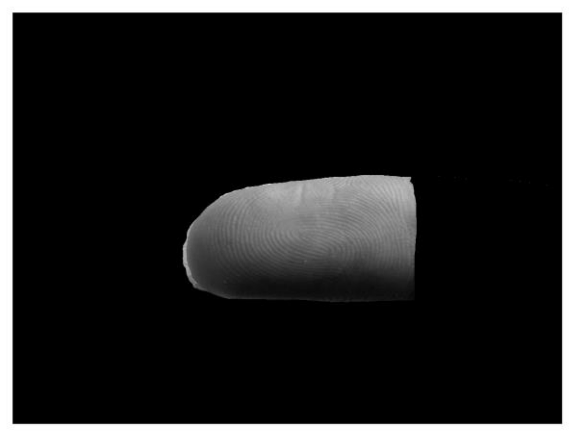
Problem Solving

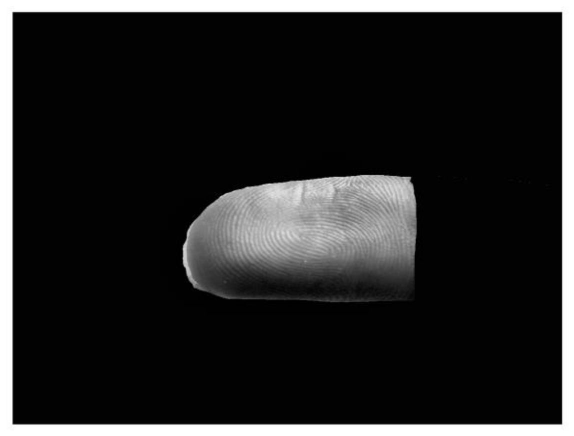
Preprocessing

Linear Normalization: $$X\_norm = \frac{X - Xmin}{Xmax - Xmiin}$$

CLAHE:

```
    # CLAHE

clip_limit = 4
grid_shape = (6,6)
CLAHE = cv.createCLAHE(clipLimit = clip_limit, tileGridSize = grid_shape)
output_CLAHE = CLAHE.apply(mask_img)
output_CLAHE = np.uint8(output_CLAHE)
```

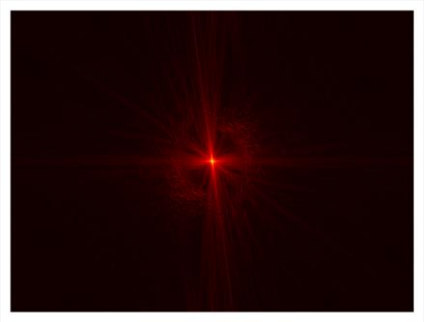


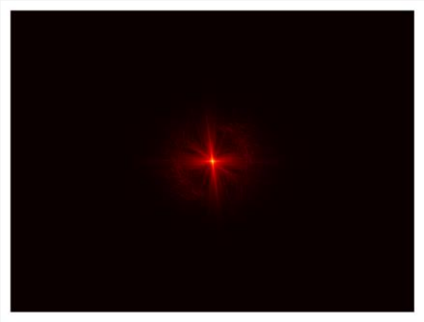


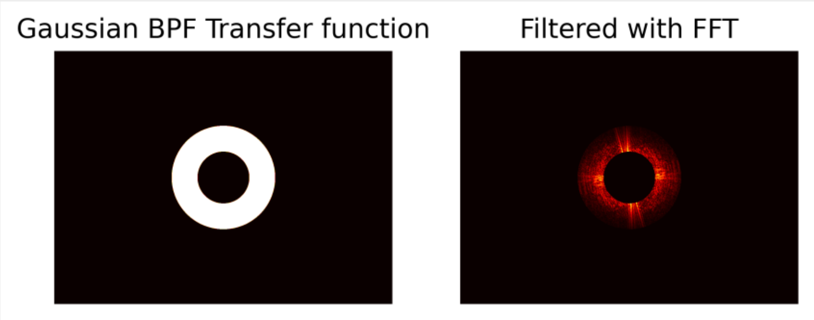
Problem Solving

Feature Extraction: Global Analysis

Input → Gaussian Blur (9,9) → Gaussian BPF (80,220) → Magnitude FFT → Variance, Mean, RMS

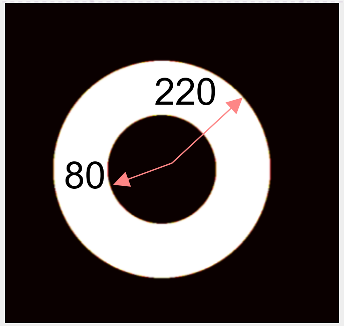




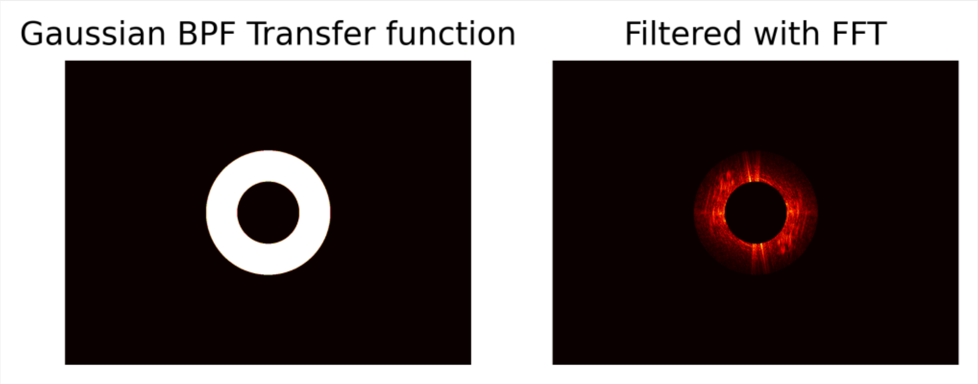


Problem Solving

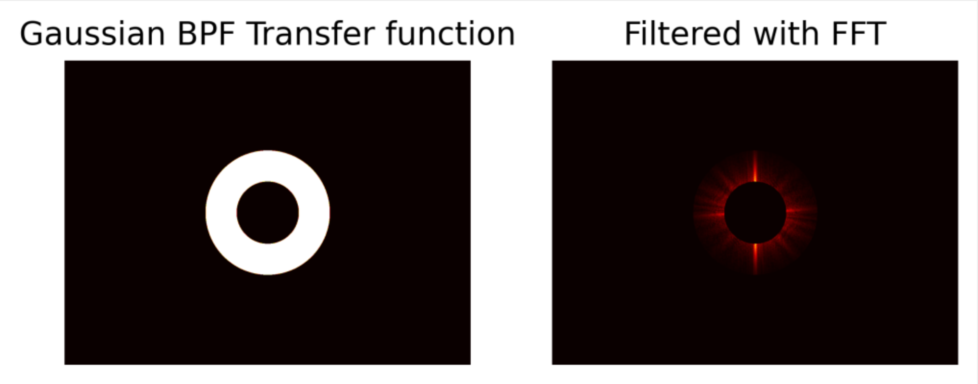
Feature Extraction: Global Analysis



1. Gaussian Band Pass Filter: D0 = 80, D1 = 220



Good Quality in Frequency Domain

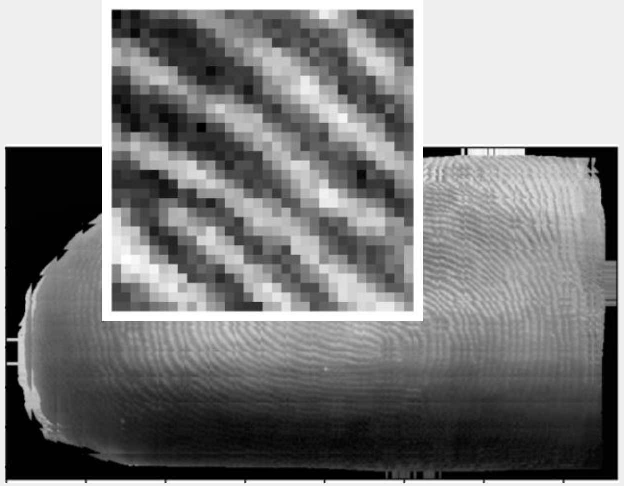


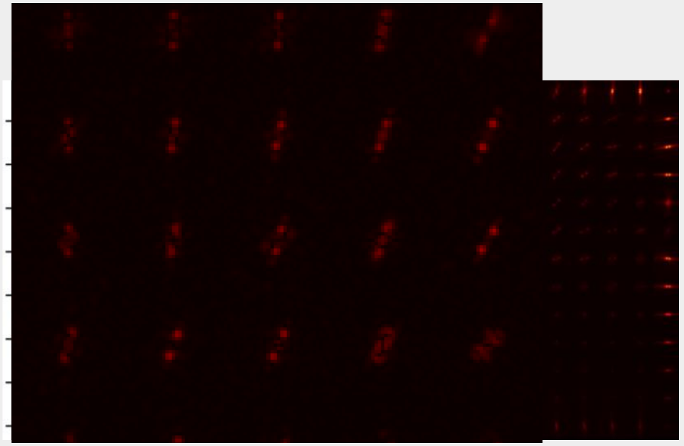
Bad Quality in Frequency Domain

Problem Solving

Feature Extraction: Local Analysis

Find Fingerprint Boundary Box → Cropped Fingerprint → STFT(kernel = 32, stride = 24) → Get STFT Magnitude → Variance, Mean, RMS



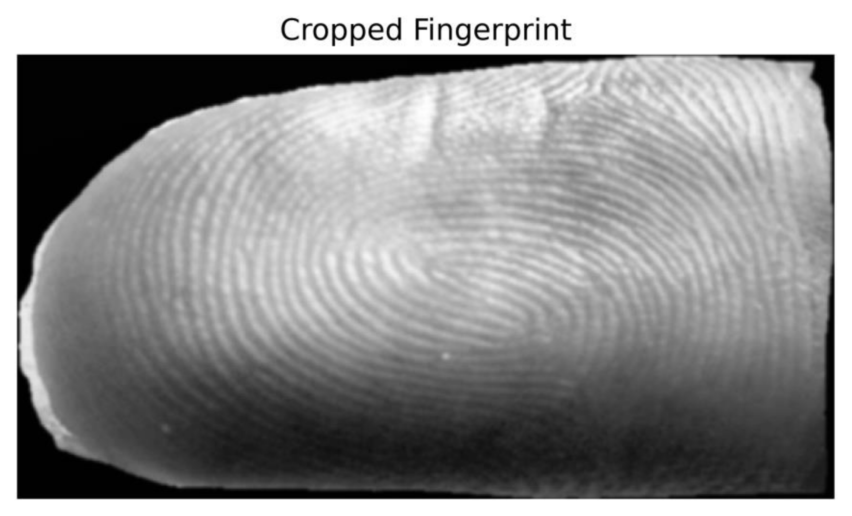


Problem Solving

Feature Extraction: Local Analysis

1. Boundary Box

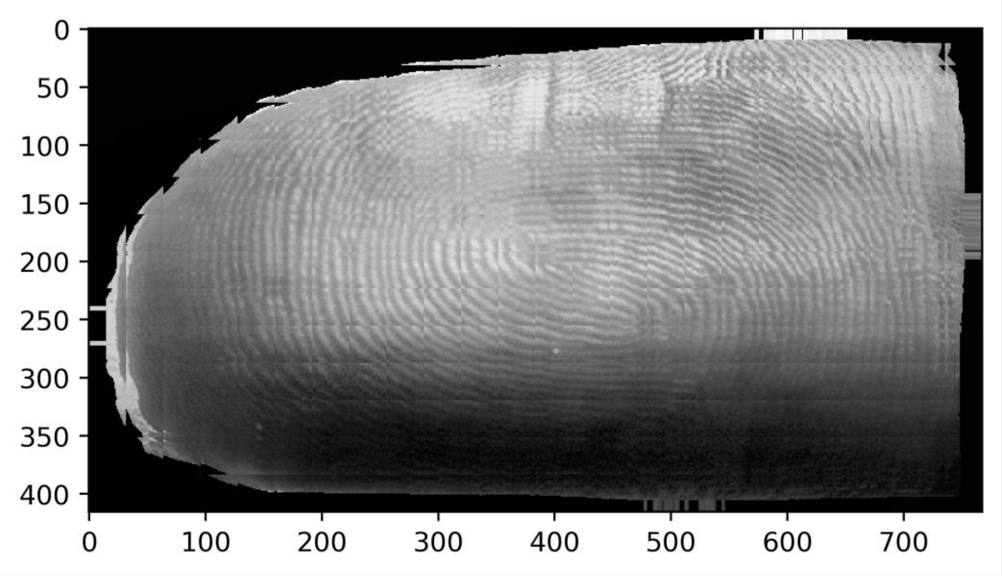
```
ret,thresh = cv.threshold(output_CLAHE,30,255,cv.THRESH_BINARY)
contours,_ = cv.findContours(thresh,cv.RETR_LIST,cv.CHAIN_APPROX_SIMPLE)

draw = output_CLAHE.copy();  i = 0;  box_list = [];  ROI_b = [];    j=0;

for cntl in contours:
    area = cv.contourArea(cntl)
    area_norm = area / (output_CLAHE.shape[0] * output_CLAHE.shape[1])

    (cx, cy), (w, h), angle = cv.minAreaRect(cntl)
    x0,y0,w0,h0 = cv.boundingRect(cntl)

    if (area_norm > 0.05 ):

        rect = cv.minAreaRect(cntl)
        box = cv.boxPoints(rect)
        box = np.int0(box)
        cv.drawContours(draw,[box],0,(255,255,0),12)
        x,y,w,h = cv.boundingRect(cntl)
        box_list.append(box);       i = i+1;
        roi_data = [y, y+h, x, x+w]
        ROI_b.append(roi_data)


output_CLAHE_ = output_CLAHE.copy()
output_CLAHE_crop = output_CLAHE_[ROI_b[j][0]:ROI_b[j][1], ROI_b[j][2]:ROI_b[j][3]]
```



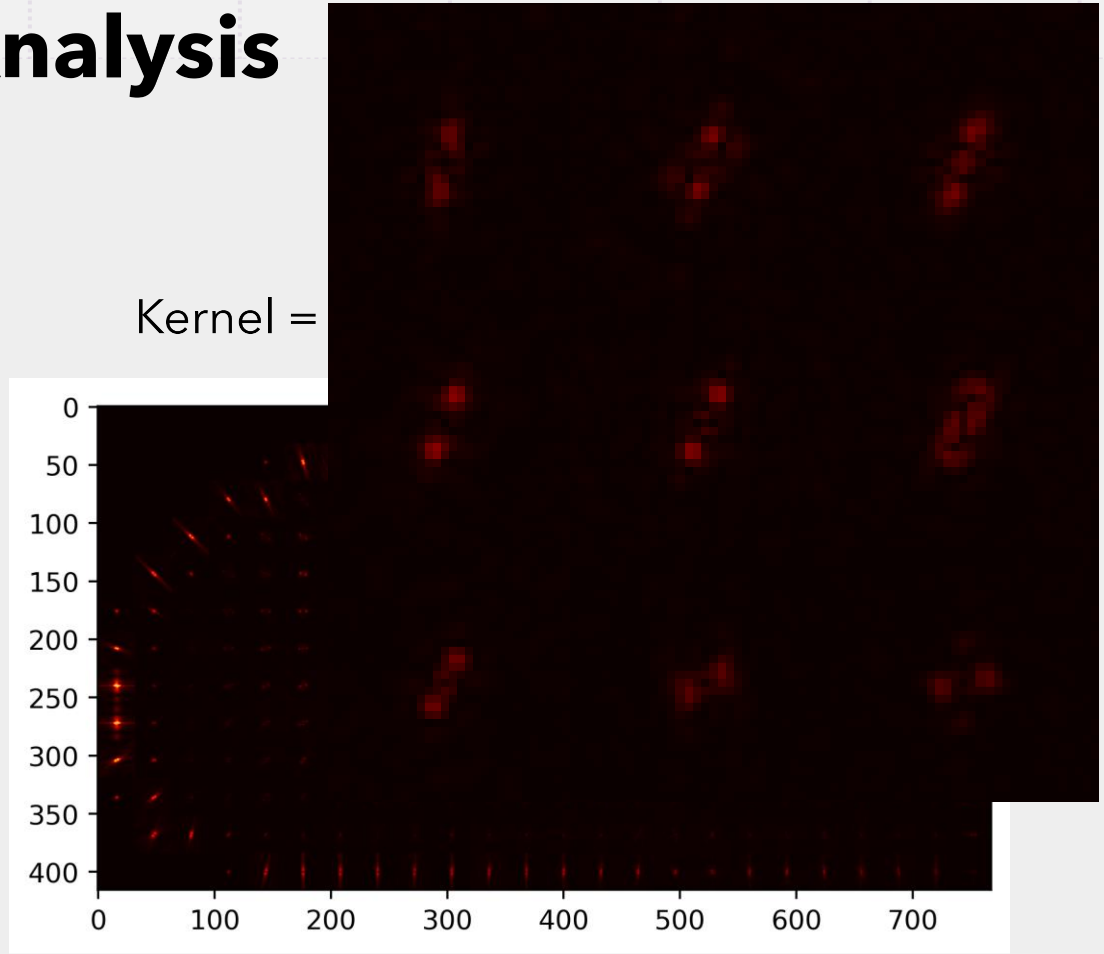
Problem Solving

Feature Extraction: Local Analysis

1. Short-time Fourier Transform

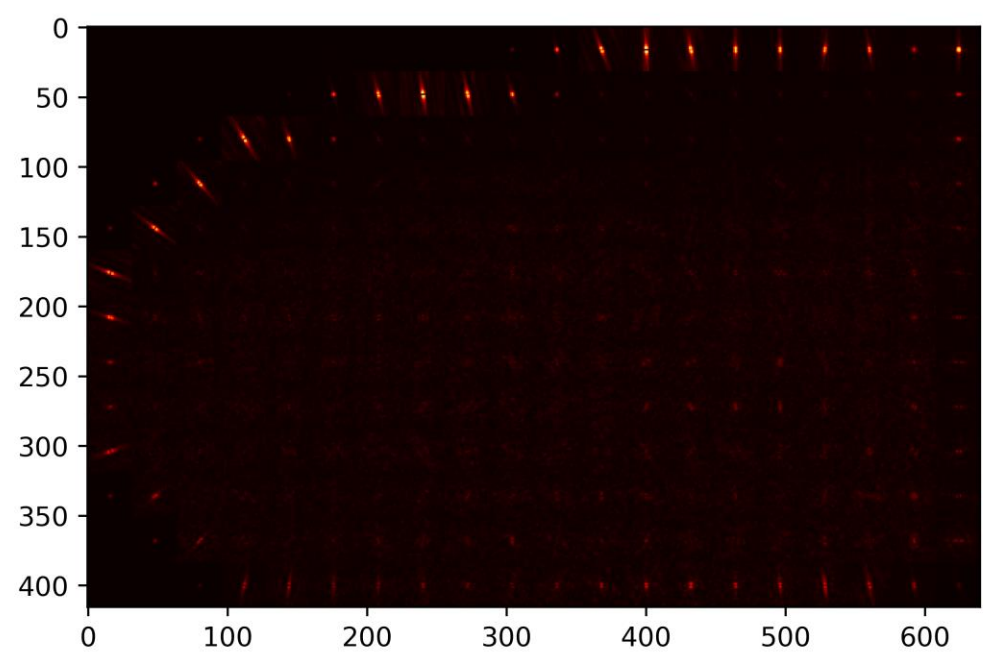


Kernel =

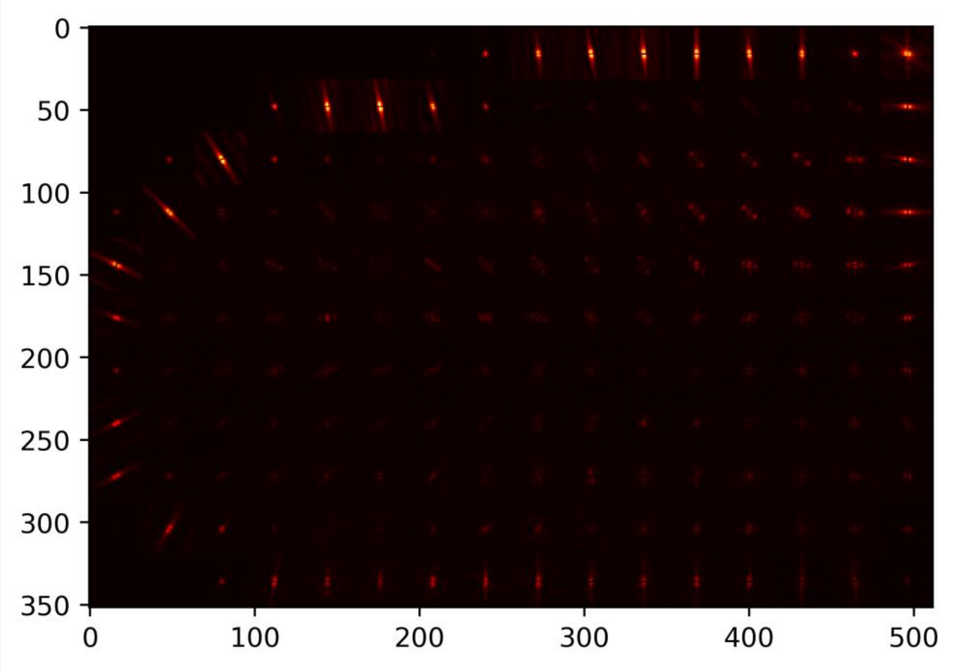


Problem Solving

Feature Extraction: Local Analysis



Bad Quality in STFT

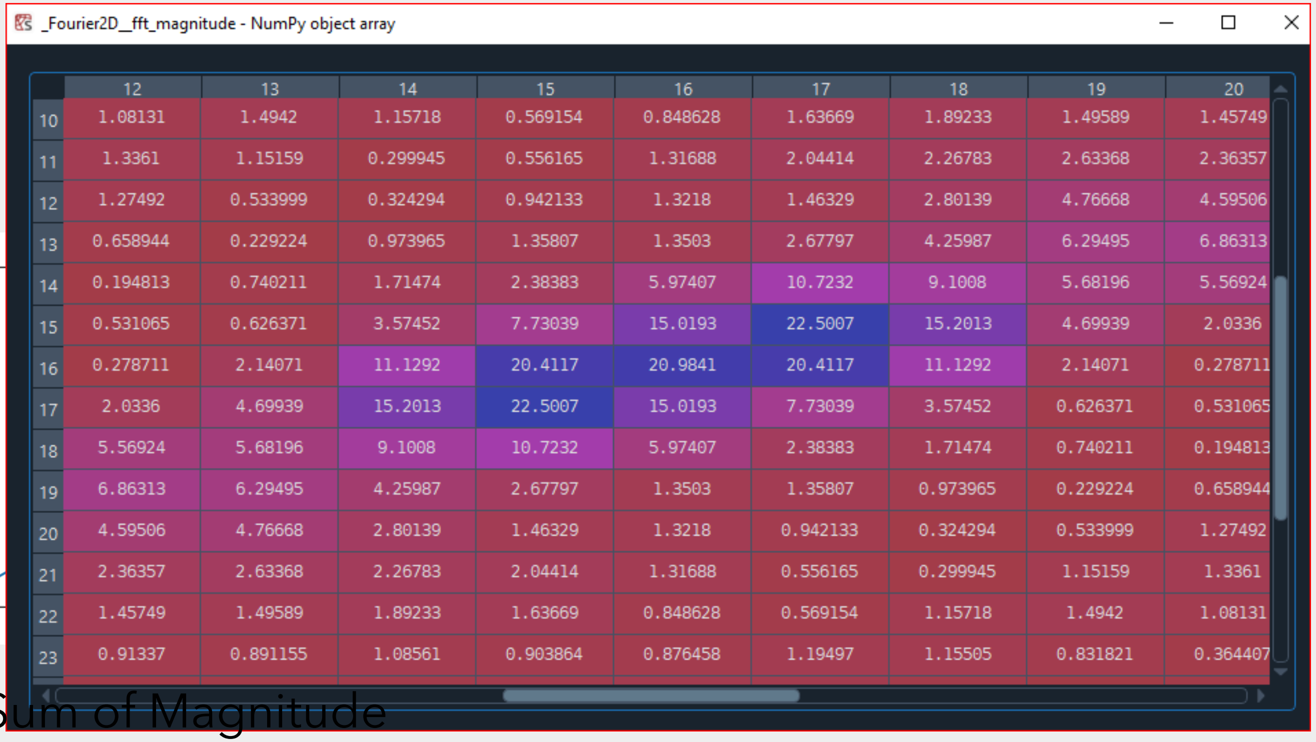


Good Quality in STFT

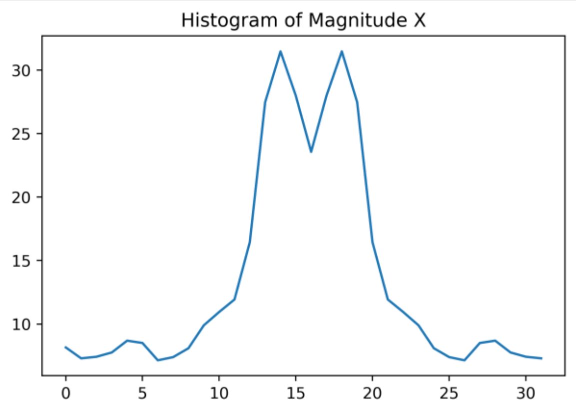
Problem Solving

Feature Extraction: Local Analysis

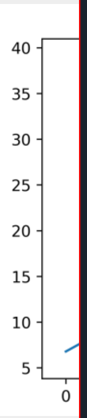
1. 32 x 32 Magnitude Matrix


_Fourier2D__fft_magnitude - NumPy object array

	12	13	14	15	16	17	18	19	20
10	1.08131	1.4942	1.15718	0.569154	0.848628	1.63669	1.89233	1.49589	1.45749
11	1.3361	1.15159	0.299945	0.556165	1.31688	2.04414	2.26783	2.63368	2.36357
12	1.27492	0.533999	0.324294	0.942133	1.3218	1.46329	2.80139	4.76668	4.59506
13	0.658944	0.229224	0.973965	1.35807	1.3503	2.67797	4.25987	6.29495	6.86313
14	0.194813	0.740211	1.71474	2.38383	5.97407	10.7232	9.1008	5.68196	5.56924
15	0.531065	0.626371	3.57452	7.73039	15.0193	22.5007	15.2013	4.69939	2.0336
16	0.278711	2.14071	11.1292	20.4117	20.9841	20.4117	11.1292	2.14071	0.278711
17	2.0336	4.69939	15.2013	22.5007	15.0193	7.73039	3.57452	0.626371	0.531065
18	5.56924	5.68196	9.1008	10.7232	5.97407	2.38383	1.71474	0.740211	0.194813
19	6.86313	6.29495	4.25987	2.67797	1.3503	1.35807	0.973965	0.229224	0.658944
20	4.59506	4.76668	2.80139	1.46329	1.3218	0.942133	0.324294	0.533999	1.27492
21	2.36357	2.63368	2.26783	2.04414	1.31688	0.556165	0.299945	1.15159	1.3361
22	1.45749	1.49589	1.89233	1.63669	0.848628	0.569154	1.15718	1.4942	1.08131
23	0.91337	0.891155	1.08561	0.903864	0.876458	1.19497	1.15505	0.831821	0.364407



Sum of Magnitude
in X-axis



Sum of Magnitude
in Y-axis

Problem Solving

Classification

6 features: (Variance, Mean, RMS) of Global and Local Analysis → Standard Scaler → 70-30 Train-Test Split → KNN(5), SVM, Decision Tree, ANN

Problem Solving

Classification

```
scaler = StandardScaler()

#X_ = [X[0:3] for X in X]
X_ = scaler.fit_transform(X)
#X_norm = normalize(X, norm='l2')
X_train, X_test, y_train, y_true = train_test_split(X_, y_, test_size = 0.3, random_state=99)

# KNN

KNN = KNeighborsClassifier(n_neighbors=5)
KNN.fit(X_train, y_train)
y_test = KNN.predict(X_test)

# SVM

SVM = make_pipeline(StandardScaler(), SVC(gamma='auto'))
SVM.fit(X_train, y_train)
y_test = SVM.predict(X_test)

# Decision Tree

DT = tree.DecisionTreeClassifier()
DT = DT.fit(X_train, y_train)
y_test = DT.predict(X_test)

# Neural Network

NN = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1)
NN.fit(X_train, y_train)
y_test = NN.predict(X_test)
```

Result

All Features: Variance, Mean, RMS of Global Analysis and Local Analysis

	KNN	SVM	DT	ANN
Acc	0.71	0.68	0.66	0.79
Error	0.29	0.32	0.34	0.21
Precision	0.67	0.64	0.58	0.75
Recall	0.63	0.56	0.69	0.75

Result

Global Analysis Features: Variance, Mean, RMS

	KNN	SVM	DT	ANN
Acc	0.55	0.71	0.68	0.66
Error	0.45	0.29	0.32	0.34
Precision	0.45	0.86	0.62	0.80
Recall	0.31	0.37	0.62	0.25

Result

Local Analysis Features: Variance, Mean, RMS

	KNN	SVM	DT	ANN
Acc	0.50	0.55	0.53	0.53
Error	0.50	0.45	0.47	0.47
Precision	0.33	0.40	0.42	0.40
Recall	0.19	0.12	0.31	0.25

Analysis

J3 Score of Features

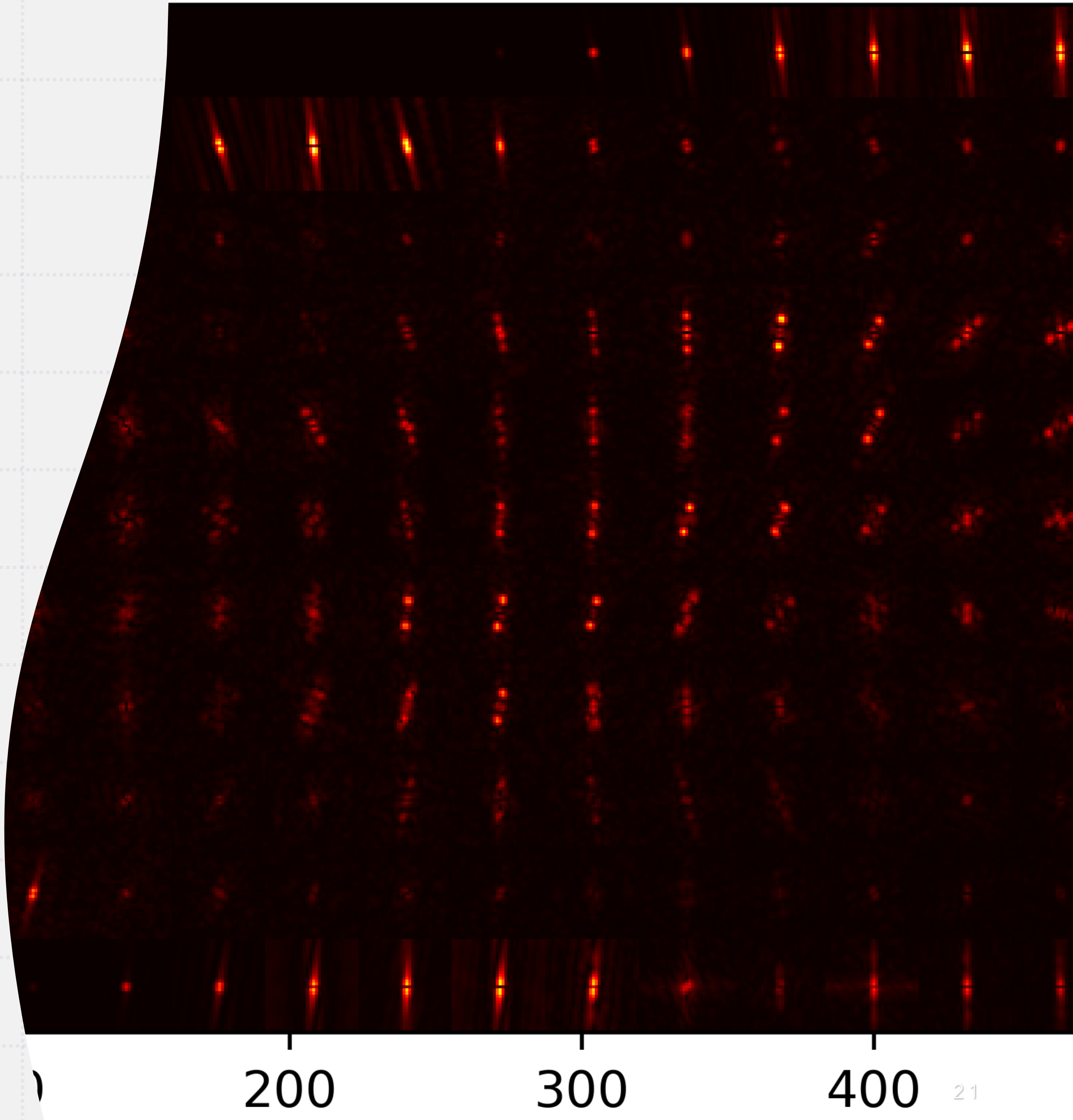
	1	2	3	4	5	6
1		2.46	2.21	2.16	2.17	2.16
2	2.46		2.29	2.01	2.00	2.00
3	2.21	2.29		2.12	2.13	2.12
4	2.16	2.01	2.12		2.01	2.01
5	2.17	2.00	2.13	2.01		2.01
6	2.16	2.00	2.12	2.01	2.01	
All Feature	6.64					

- All Features has the best J3 Score
- All Features Classification has 0.79% Accuracy, 0.75% precision and recall
- ANN is the best in All Features for Accuracy, Precision and Recall
- All Feature work the best which is the same for class separability.

Summary

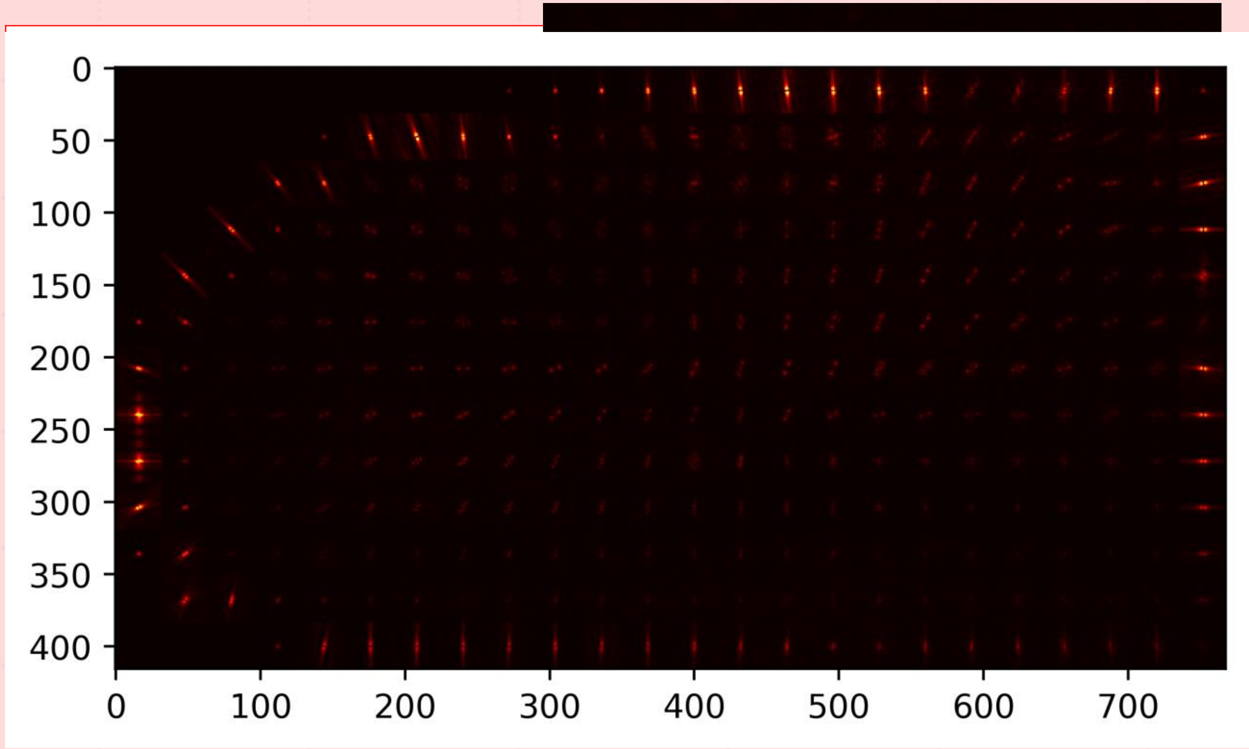
For Quality Fingerprint Assessment, Global and Local Analysis is used for finding features; Variance, Mean, and RMS of Both Analysis resulting in all features have the best accuracy, precision and recall for classification which is in the same way as class separability.

Fourier Transform and Short-time Fourier Transform is adopted for analysis in this project for finding features for classification



Improvement

- In Local Analysis:



1. Ban DC Magnitude of each Block
2. Apply Gaussian Bandpass Filter
3. Find PCA of each block
4. Affine Transform Matrix using PCA
5. Use Phase for detecting Peaking
6. More assuring criteria for boundary of fingerprint

Thank you

Wiwitthawin Charoenngam

Wiwitthawin.c@ku.th

6310501933 Kasetsart University