

Shape detection

by methods of Feature Extraction and CNN

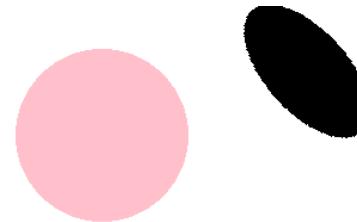
Sparsha Ray

Rishi Vora

1	Dataset	1
1.1	<i>Samples</i>	2
1.2	<i>Stats</i>	3
2	Feature Extraction	4
3	Convolutional Neural Networks	20
4	Conclusion	30

1.1 Samples

1 Dataset



circle



oval



parallelogram



pentagon



rectangle



rhombus



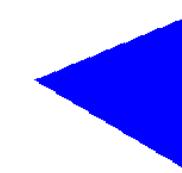
semicircle



square



trapezoid



triangle

1.2 Stats

- 10 labels
- 12000 images for each label
- All images are 224x224 pixels.
- Each class contains shapes that are rotated, scaled and differently coloured.

1	Dataset	1
2	Feature Extraction	4
2.1	<i>Overview</i>	5
2.2	<i>Preprocessing</i>	6
2.3	<i>Finding contour</i>	7
2.4	<i>Feature extraction</i>	8
2.5	<i>Training</i>	16
2.6	<i>Evaluation</i>	17
3	Convolutional Neural Networks	20
4	Conclusion	30

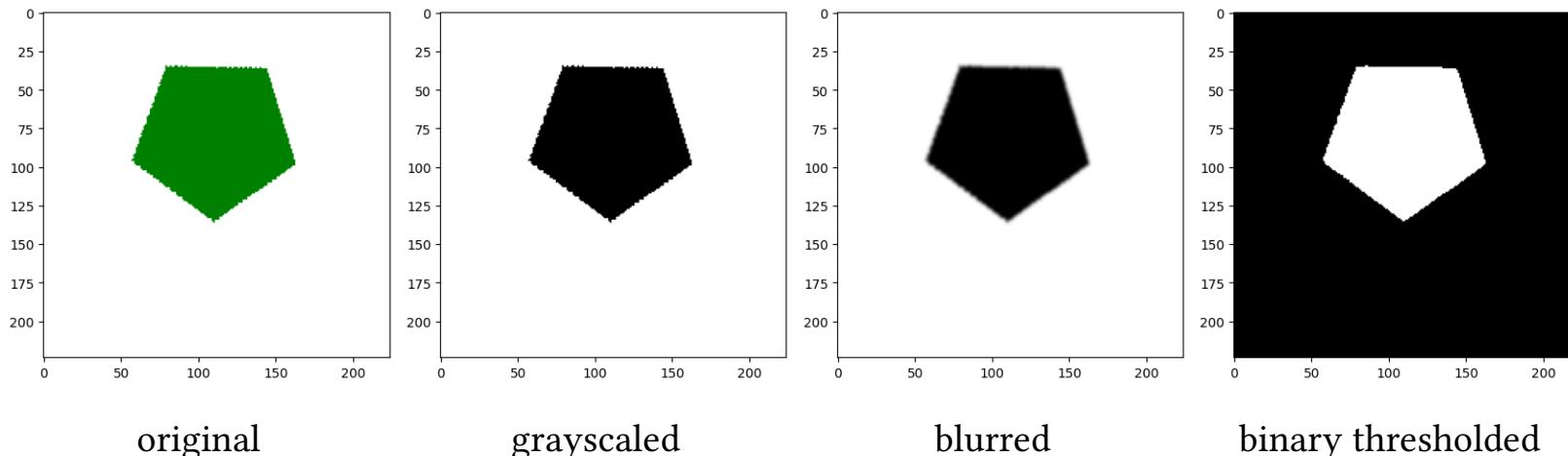
2.1 Overview

- The core idea is to extract the feature vector manually and then feed it to a classifier.
- The features that we chose are:
 - ▶ Number of corners
 - ▶ Solidity
 - ▶ Aspect ratio
 - ▶ Circularity
 - ▶ Hu Moments
- For classification, we chose Random Forests Classifier.

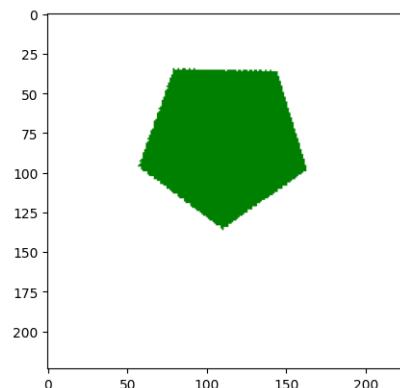
We use **OpenCV** for all image manipulation.

2.2 Preprocessing

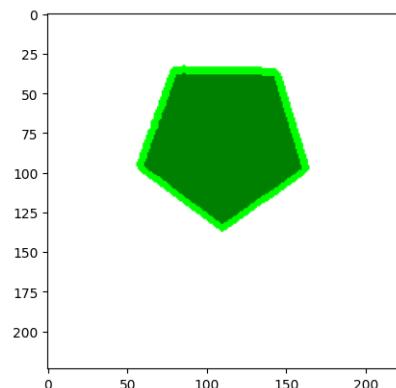
- Read the image
- Make it gray-scale
- Blur it to reduce noise
- Threshold it to make it binary



2.3 Finding contour



original

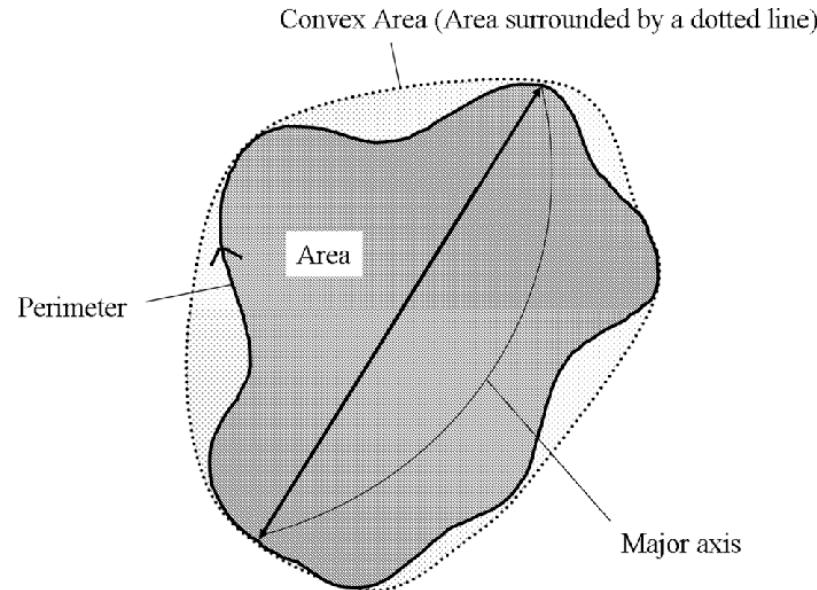


with contour

1. Number of corners

- great first filter
- can categorize shapes into broad groups like:
 - ▶ triangles (3 corners)
 - ▶ squares + rectangles + parallelogram + rhombus (4 corners)
 - ▶ semicircle (2 corners)
 - ▶ circle + oval (no corners)

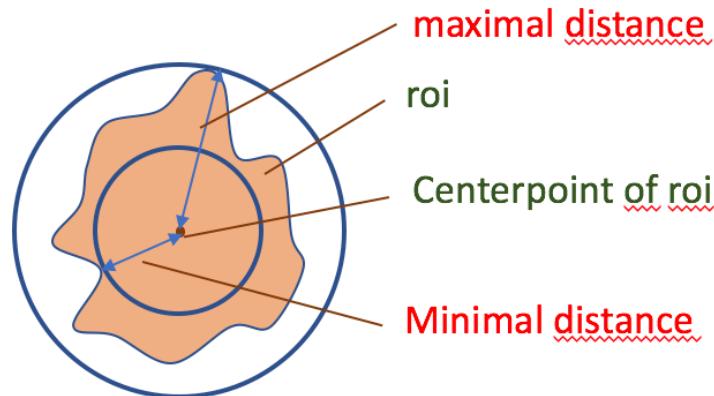
2. Solidity



$$\text{Solidity} = \frac{\text{Convex hull area}}{\text{Area}}$$

- All the shapes we chose have solidity 1.0 but it is a great filter for concave shapes like stars.

3. Circularity (Roundness)



$$\text{Circularity} = \frac{4\pi * \text{Area}}{\text{Perimeter}^2}$$

- A perfect circle will have a score of exactly 1.0.

4. Aspect Ratio

- We draw the tightest upright box around the shape and divides its width by its height.

4. Aspect Ratio

- We draw the tightest upright box around the shape and divides its width by its height.
- square and circle will have an aspect ratio of 1.0.
- rectangle and oval will have an aspect ratio not equal to 1.0.

4. Aspect Ratio

- We draw the tightest upright box around the shape and divides its width by its height.
- square and circle will have an aspect ratio of 1.0.
- rectangle and oval will have an aspect ratio not equal to 1.0.
- But this can be easily fooled by rotation.

5. Hu Moments

- Set of 7 numbers that can be considered unique “mathematical fingerprint” for a shape’s geometry.
- Derived from the field of Statistics

The central moments are defined as

$$\mu_{ij} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j I(x, y)$$

where (x, y) are pixel coordinates, (\bar{x}, \bar{y}) is the centroid of the shape, and $I(x, y)$ are the intensity values.

Central moments are translation invariant.

Using these, we get normalized central moments

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\left(1+\frac{i+j}{2}\right)}}$$

These are invariant to scaling and translation both.

Using these, M. K. Hu, derived a set of 7 moments that are invariant to translation, scaling, and rotation.

$$I_1 = \eta_{20} + \eta_{02}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$I_3 = (\eta_{20} - 3\eta_{12})^2 + (3\eta_{11} - \eta_{02})^2$$

$$I_4 = (\eta_{20} - \eta_{12})^2 + (\eta_{21} + \eta_{13})^2$$

$$I_5 = (\eta_{20} - 3\eta_{12})(\eta_{20} + \eta_{12})[(\eta_{20} + \eta_{12})^2 - 3(\eta_{21} + \eta_{13})^2] \\ + (3\eta_{21} - \eta_{13})(\eta_{21} + \eta_{13})[3(\eta_{20} + \eta_{12})^2 - (\eta_{21} + \eta_{13})^2]$$

$$I_6 = (\eta_{20} - \eta_{12})(\eta_{20} + \eta_{12})^2 - (\eta_{21} + \eta_{33})^2 \\ + 4\eta_{11}(\eta_{20} + \eta_{12})(\eta_{21} + \eta_{13})$$

$$I_7 = (8\eta_{21} - \eta_{30})(\eta_{30} + \eta_{22})[(\eta_{30} + \eta_{22})^2 - 3(\eta_{21} + \eta_{33})^2] \\ - (\eta_{30} - 3\eta_{22})(\eta_{21} + \eta_{33})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{33})^2]$$

2.4 Feature extraction

2 Feature Extraction

Images	# H[1]	# H[2]	# H[3]	# H[4]	# H[5]	# H[6]	# H[7]	images
triangle	0.711	3.114	2.334	4.563	6.96	6.068	7	
pentagon	0.791	5.32	6.332	7	-7	-7	-7	
rectangle	0.744	2.329	7	7	-7	7	7	
circle	0.798	7	7	7	-7	7	7	
trapezoid	0.753	2.831	2.999	6.745	7	6.987	7	
oval	0.747	2.203	4.935	6.275	-7	-6.873	7	
semicircle	0.691	1.882	2.905	4.12	-6.909	-5.058	-6.993	

- Now we have 11 features (corners, solidity, aspect ratio, circularity, and 7 Hu moments.)
- We train a Random Forests Classifier with
 - ▶ 100 estimators
 - ▶ 120k samples
 - ▶ 80% training, 20% testing samples

2.6 Evaluation

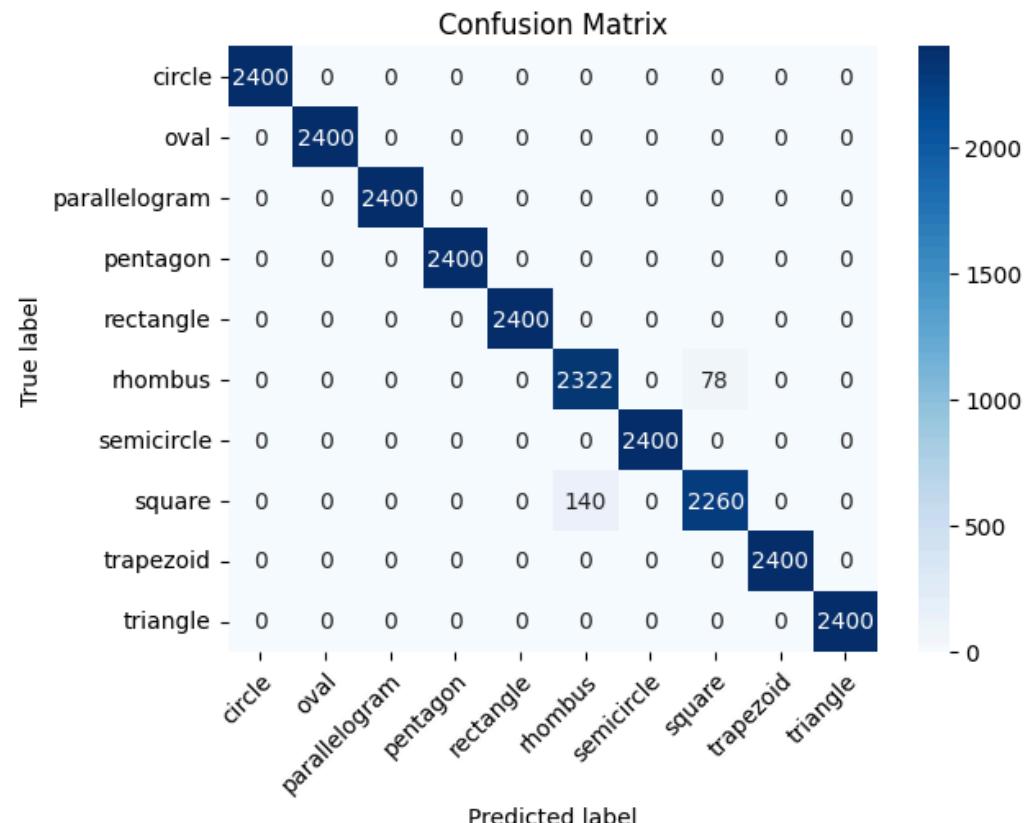
2.6 Evaluation

- Model Accuracy: **99.11%**

- Model Accuracy: **99.11%**

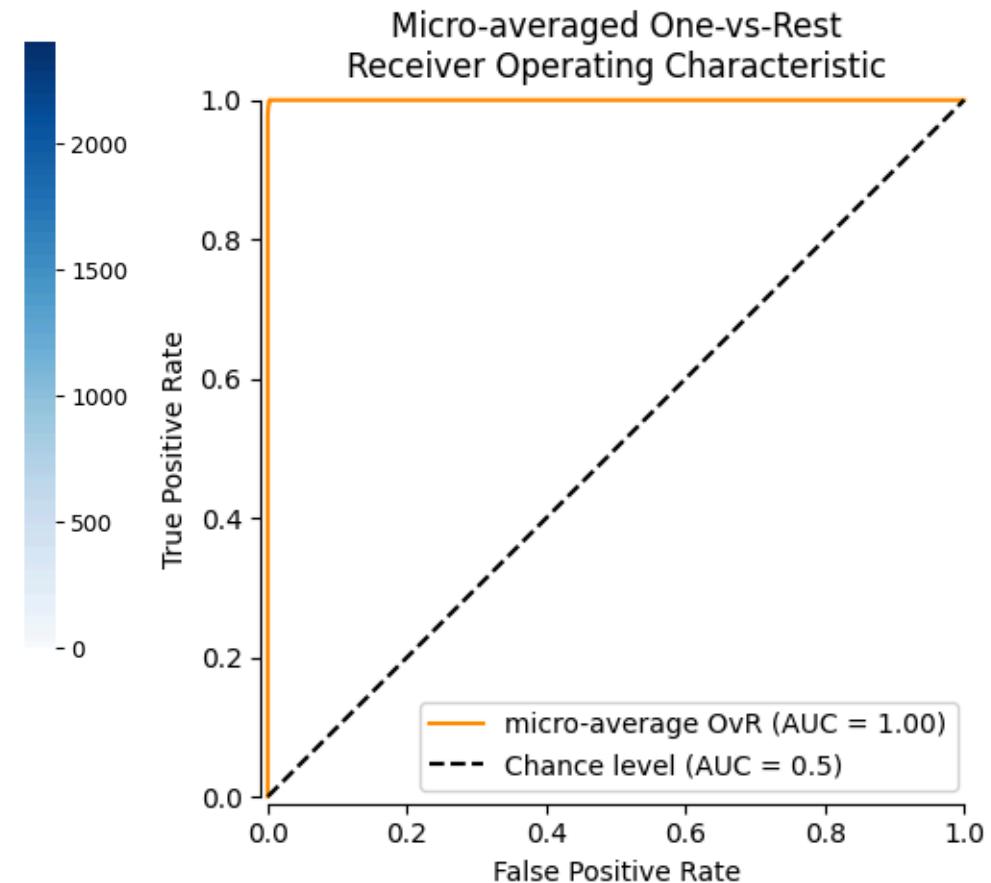
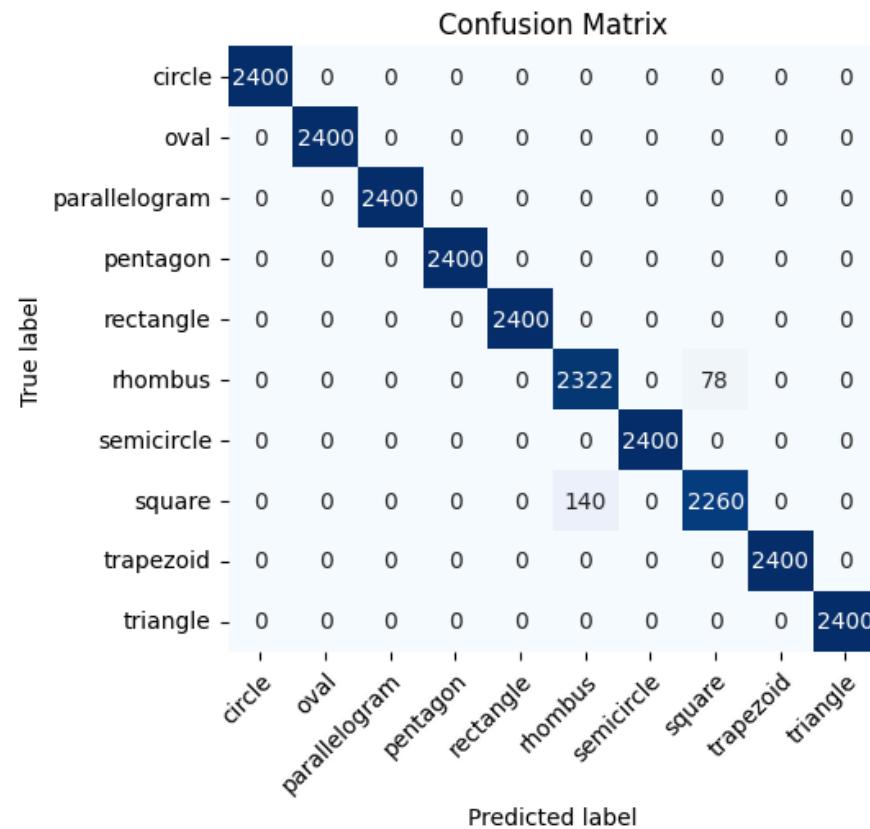
	precision	recall	f1-score	support
parallelogram	1.00	1.00	1.00	2400
triangle	1.00	1.00	1.00	2400
pentagon	1.00	1.00	1.00	2400
rectangle	1.00	1.00	1.00	2400
square	0.97	0.94	0.95	2400
circle	1.00	1.00	1.00	2400
trapezoid	1.00	1.00	1.00	2400
oval	1.00	1.00	1.00	2400
semicircle	1.00	1.00	1.00	2400
rhombus	0.94	0.97	0.96	2400
accuracy			0.99	24000
macro avg	0.99	0.99	0.99	24000
weighted avg	0.99	0.99	0.99	24000

2.6 Evaluation

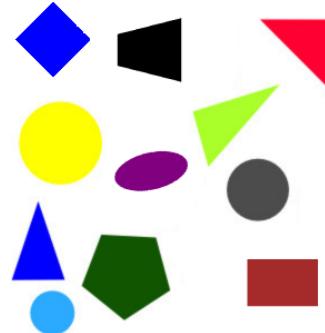


2.6 Evaluation

2 Feature Extraction



We also made a function recognize all present shapes from an image, by extracting all contours.



```
1 all_shapes_features = extract_features_from_all_shapes(  
2     multi_shape_image, threshold_value=250  
3 )  
4 predicted_shapes = model.predict(all_shapes_features)  
5 Counter(predicted_shapes)  
  
Counter({'circle': 3, 'triangle': 3, 'pentagon': 2, 'rectangle': 1, 'oval': 1, 'trapezoid': 1})
```

It mostly correctly identified all the shapes.

1	Dataset	1
2	Feature Extraction	4

3 Convolutional Neural Networks . . 20

3.1	<i>What is convolution ?</i>	21
3.2	<i>What are filters ?</i>	22
3.3	<i>How does CNNs work ?</i>	23
3.4	<i>Architecture</i>	24
3.5	<i>Training</i>	27
3.6	<i>Results</i>	28
3.7	<i>Interpretations</i>	29
4	Conclusion	30

3.1 What is convolution ?

3 Convolutional Neural Networks

3.1 What is convolution ?

3 Convolutional Neural Networks

$$\begin{matrix} & \begin{matrix} 3 & 0 & -1 \\ 1 & 5 & -1 \\ 2 & 7 & -1 \end{matrix} & \begin{matrix} 2 & 7 & 4 \\ 9 & 3 & 1 \\ 5 & 1 & 3 \end{matrix} \\ \begin{matrix} 3 & 0 & -1 \\ 1 & 5 & -1 \\ 2 & 7 & -1 \end{matrix} & * & \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix} \\ & & \begin{matrix} 6 \times 6 & & 3 \times 3 & & 4 \times 4 \end{matrix} \end{matrix}$$

The diagram illustrates a convolution operation. It shows a 6x6 input matrix and a 3x3 kernel matrix. The input matrix has its top-left 3x3 block highlighted in red. The kernel matrix is also highlighted in red. The result of the convolution is a 4x4 output matrix, where only the top-left element (-5) is highlighted in red.

$$3 \times 1 + 1 \times 1 + 2 \times 1 + 0 \times 0 + 5 \times 0 + 7 \times 0 + 1 \times -1 + 8 \times -1 + 2 \times -1 = -5$$

3.1 What is convolution ?

3 Convolutional Neural Networks

$$\begin{array}{|c|c|c|c|c|c|} \hline 3 & 0_1 & 1_0 & 2_{-1} & 7 & 4 \\ \hline 1 & 5_1 & 8_0 & 9_{-1} & 3 & 1 \\ \hline 2 & 7_1 & 2_0 & 5_{-1} & 1 & 3 \\ \hline 0 & 1 & 3 & 1 & 7 & 8 \\ \hline 4 & 2 & 1 & 6 & 2 & 8 \\ \hline 2 & 4 & 5 & 2 & 3 & 9 \\ \hline \end{array} \quad * \quad \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline -5 & -4 & & \\ \hline & & & \\ \hline \end{array}$$

6×6 3×3 4×4

$0 \times 1 + 5 \times 1 + 7 \times 1 + 1 \times 0 + 8 \times 0 + 2 \times 0 + 2 \times -1 + 9 \times -1 + 5 \times -1 = -4$

3.1 What is convolution ?

3 Convolutional Neural Networks

The diagram illustrates a convolution operation. On the left, a 6x6 input matrix is shown with values ranging from -1 to 9. A 3x3 kernel is applied to a 3x3 receptive field in the top-left corner of the input. The result of this convolution step is highlighted in red. This result is then multiplied by the corresponding value in the kernel (1) and summed. The resulting value is 0, which is highlighted in red in the 4x4 output matrix on the right. The output matrix has a blue background.

$\begin{matrix} 3 & 0 & \boxed{1_1} & 2_0 & 7_{-1} & 4 \\ 1 & 5 & 8_1 & 9_0 & 3_{-1} & 1 \\ 2 & 7 & 2_1 & 5_0 & 1_{-1} & 3 \\ 0 & 1 & 3 & 1 & 7 & 8 \\ 4 & 2 & 1 & 6 & 2 & 8 \\ 2 & 4 & 5 & 2 & 3 & 9 \end{matrix}$

$*$

$\begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}$

3×3

=

$\begin{matrix} -5 & -4 & \boxed{0} & \\ & & & \\ & & & \\ & & & \end{matrix}$

4×4

6×6

$$1 \times 1 + 8 \times 1 + 2 \times 1 + 2 \times 0 + 9 \times 0 + 5 \times 0 + 7 \times -1 + 3 \times -1 + 1 \times -1 = 0$$

3.1 What is convolution ?

3 Convolutional Neural Networks

$$\begin{array}{|c|c|c|c|c|c|} \hline 3 & 0 & 1 & 2 & 7 & 4 \\ \hline 1 & 1 & 5 & 0 & 8 & -1 \\ \hline 2 & 1 & 7 & 0 & 2 & -1 \\ \hline 0 & 1 & 1 & 0 & 3 & -1 \\ \hline 4 & 2 & 1 & 6 & 2 & 8 \\ \hline 2 & 4 & 5 & 2 & 3 & 9 \\ \hline \end{array} \quad * \quad \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline -5 & -4 & 0 & 8 \\ \hline -10 & & & \\ \hline \end{array}$$

6×6 3×3 4×4

$$1 \times 1 + 2 \times 1 + 0 \times 1 + 5 \times 0 + 7 \times 0 + 1 \times 0 + 8 \times -1 + 2 \times -1 + 3 \times -1 = -10$$

3.1 What is convolution ?

3 Convolutional Neural Networks

The diagram illustrates a convolution operation between two matrices. On the left is a 6×6 input matrix with values:

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	-1
4	2	1	6	2	-1
2	4	5	2	3	-1

The middle part shows a 3×3 kernel matrix with values:

1	0	-1
1	0	-1
1	0	-1

The result of the convolution is a 4×4 output matrix:

-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

The calculation for the bottom-right element of the output matrix is shown at the bottom:

$$1 \times 1 + 6 \times 1 + 2 \times 1 + 7 \times 0 + 2 \times 0 + 3 \times 0 + 8 \times -1 + 8 \times -1 + 9 \times -1 = -16$$

3.2 What are filters ?

3 Convolutional Neural Networks

Consider the filter (or kernel) $\begin{pmatrix} -3+3i & +10i & 3+3i \\ -10 & 0 & 10 \\ -3-3i & -10i & 3-3i \end{pmatrix}$



3.3 How does CNNs work ?

3 Convolutional Neural Networks

3.3 How does CNNs work ?

3 Convolutional Neural Networks

- In CNNs, a image is passed through a series of filters (which the learnt as the model is trained) ...

3.3 How does CNNs work ?

3 Convolutional Neural Networks

- In CNNs, a image is passed through a series of filters (which the learnt as the model is trained) ...
- Which extract features like edges corners etc (called feature maps) ...

3.3 How does CNNs work ?

3 Convolutional Neural Networks

- In CNNs, a image is passed through a series of filters (which the learnt as the model is trained) ...
- Which extract features like edges corners etc (called feature maps) ...
- As the feature maps are down-sampled and passed down through more and more layers of filters, specific filters starts to learn more complex higher order characteristics (like curvature, spacing between corners, parallel edges etc)

3.3 How does CNNs work ?

3 Convolutional Neural Networks

- In CNNs, a image is passed through a series of filters (which the learnt as the model is trained) ...
- Which extract features like edges corners etc (called feature maps) ...
- As the feature maps are down-sampled and passed down through more and more layers of filters, specific filters starts to learn more complex higher order characteristics (like curvature, spacing between corners, parallel edges etc)
- When these features are extracted, and the feature maps consists of only a few pixels, their values are passed down to a fully connected feed forward neural network (often with only 2 or 3 layers) which does the final classification task.

3.4 Architecture

3 Convolutional Neural Networks

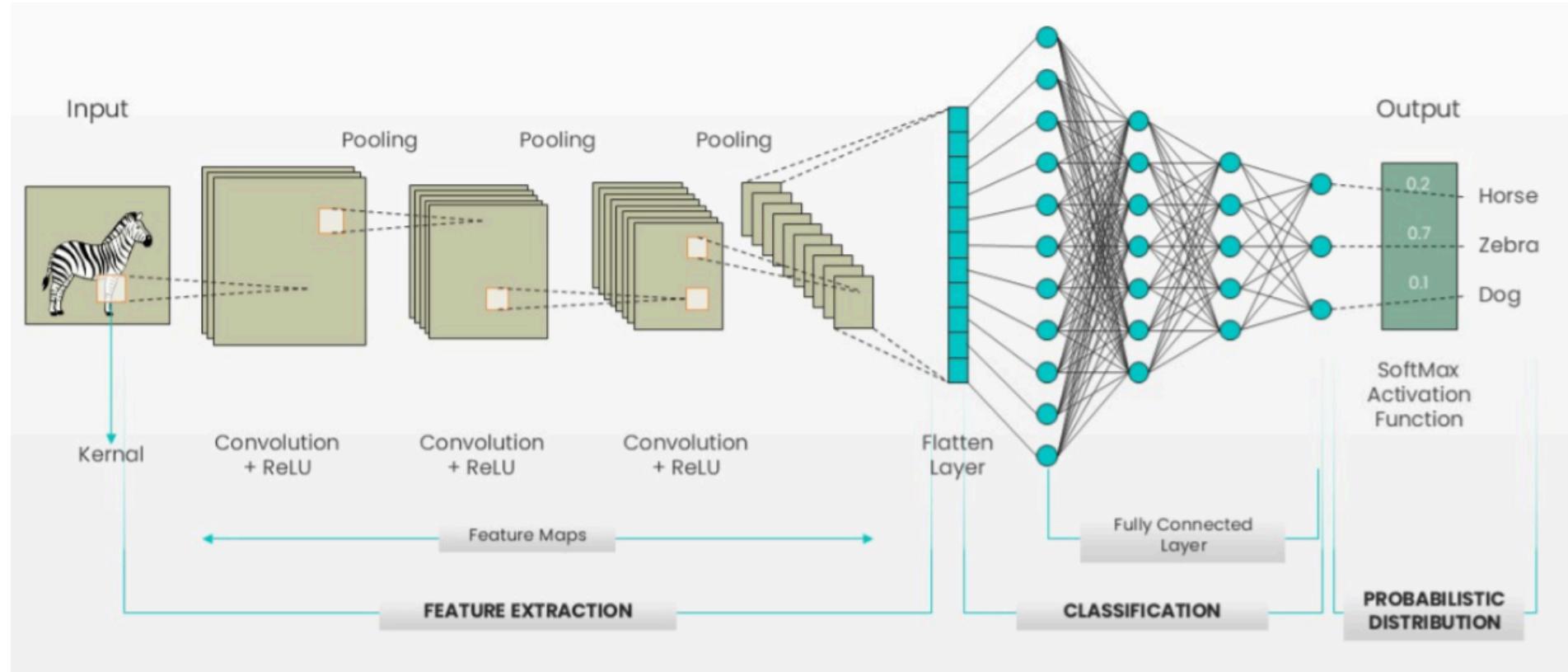


Image courtesy of Dwika Sudrajat's blog post

3.4 Architecture

3 Convolutional Neural Networks

Layer (type)	Output Shape	Parameters
conv2d_5 (Conv2D)	(None, 124, 124, 8)	80
max_pooling2d_5 (MaxPooling2D)	(None, 62, 62, 8)	0
conv2d_6 (Conv2D)	(None, 60, 60, 16)	1,168
max_pooling2d_6 (MaxPooling2D)	(None, 30, 30, 16)	0
conv2d_7 (Conv2D)	(None, 28, 28, 32)	4,640
max_pooling2d_7 (MaxPooling2D)	(None, 14, 14, 32)	0
conv2d_8 (Conv2D)	(None, 12, 12, 64)	18,496
max_pooling2d_8 (MaxPooling2D)	(None, 6, 6, 64)	0
conv2d_9 (Conv2D)	(None, 4, 4, 128)	73,856
max_pooling2d_9 (MaxPooling2D)	(None, 2, 2, 128)	0

3.4 Architecture

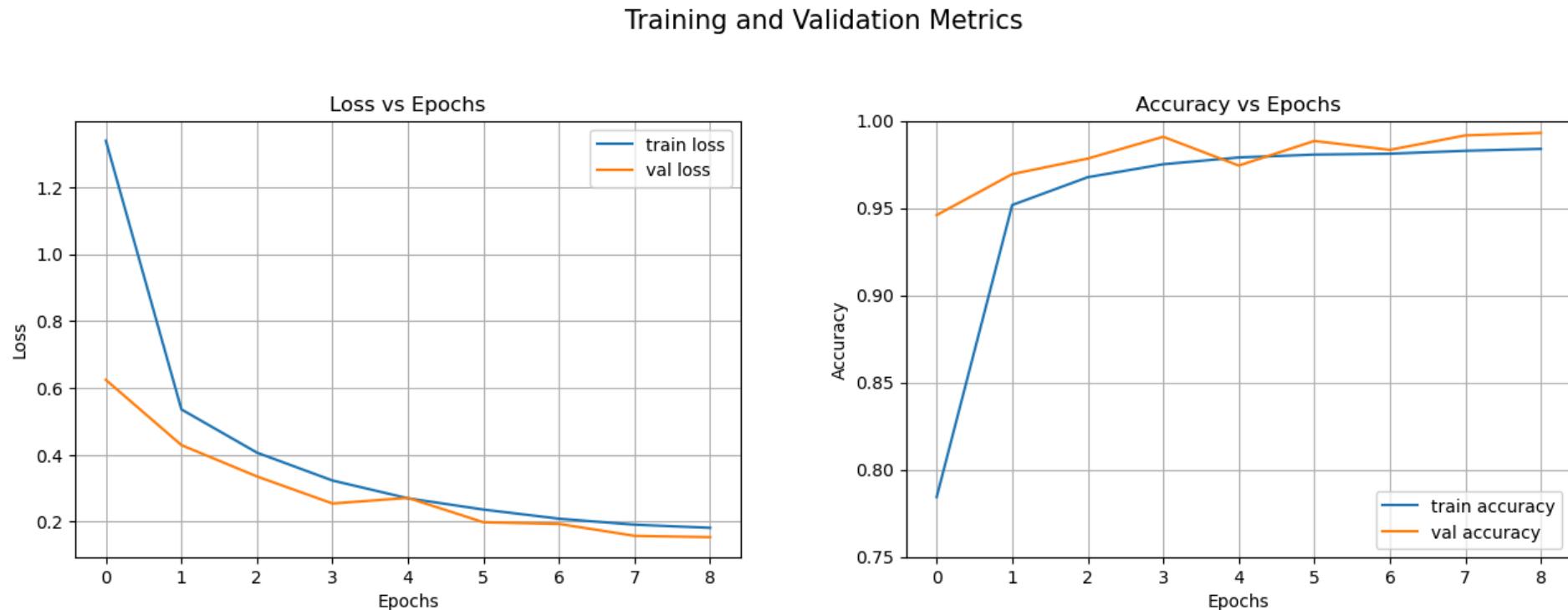
3 Convolutional Neural Networks

Layer (type)	Output Shape	Parameters
flatten_1 (Flatten)	(None, 512)	0
dense_3 (Dense)	(None, 128)	65,664
dense_4 (Dense)	(None, 64)	8,256
dense_5 (Dense)	(None, 10)	650

3.5 Training

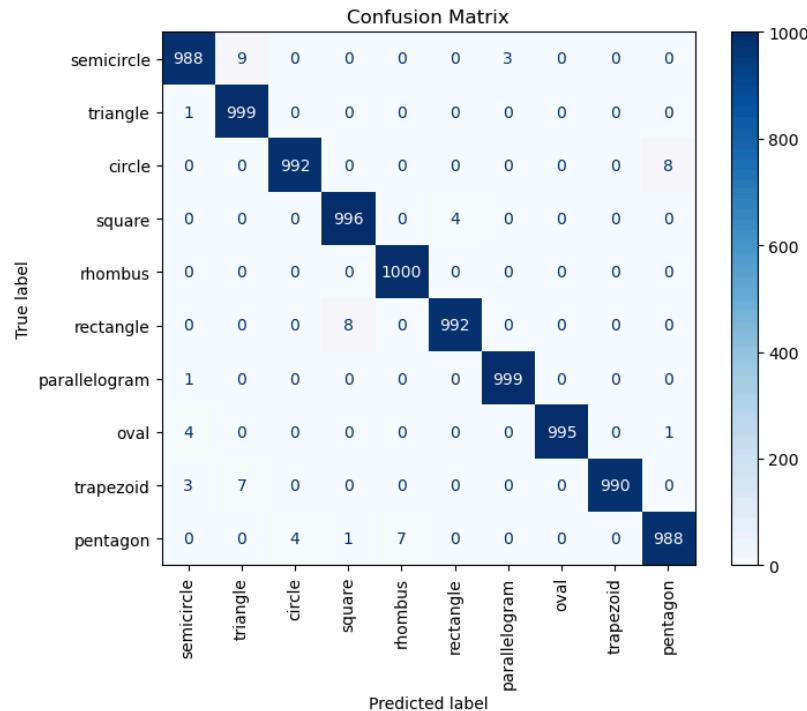
3 Convolutional Neural Networks

Training is done with the Adam optimizer for 9 epochs only



3.6 Results

3 Convolutional Neural Networks

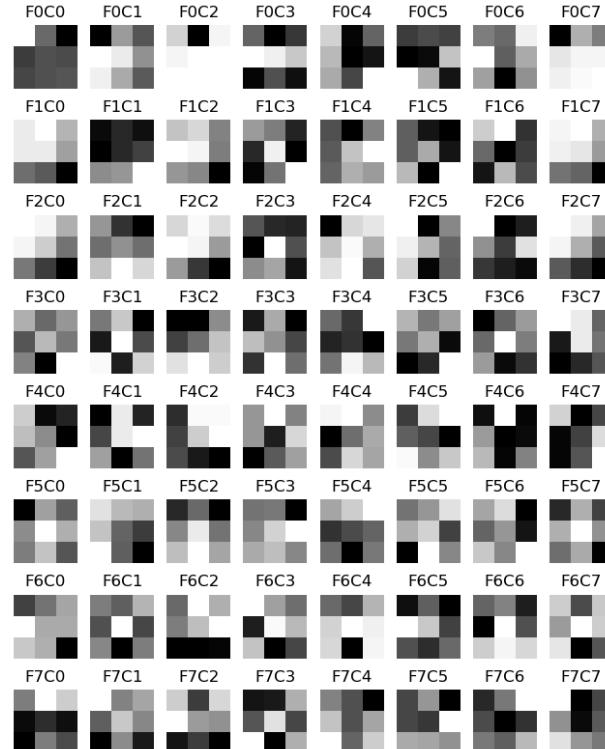


Class	Precision	Recall	f1-score
semicircle	0.99	0.99	0.99
triangle	0.98	1.00	0.99
circle	1.00	0.99	0.99
square	0.99	1.00	0.99
rhombus	0.99	1.00	1.00
rectangle	1.00	0.99	0.99
parallelogram	1.00	1.00	1.00
oval	1.00	0.99	1.00
trapezoid	1.00	0.99	0.99
pentagon	0.99	0.99	0.99
<i>Overall Accuracy</i>			0.99

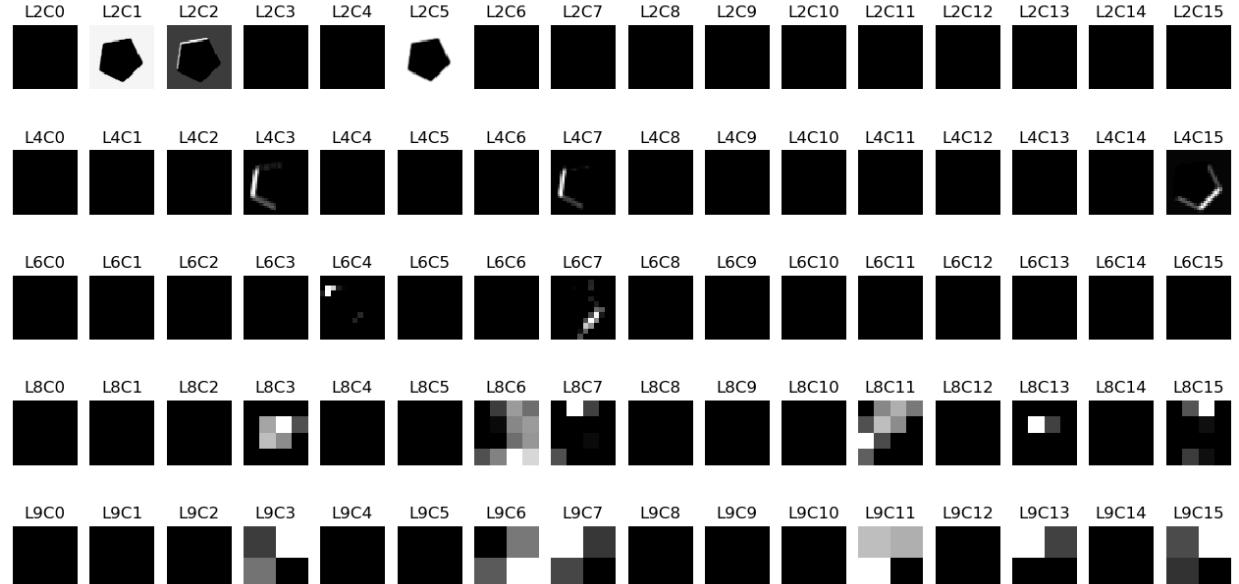
3.7 Interpretations

3 Convolutional Neural Networks

Some of the Second Layer Filters



Feature Maps of Different Layers for the Test Image



1	Dataset	1
2	Feature Extraction	4
3	Convolutional Neural Networks	20

4 Conclusion 30

<i>4.1 How does these two methods compare</i>	<i>31</i>
---	-----------

4.1 How does these two methods compare

4 Conclusion

- Both reaches similar accuracy, CNN is slower, Feature Extraction + Random Forests (RF) is much faster
- CNN learns the features by itself, FE+RF is given the features to classify
- FE+RF doesn't generalize well for complex shapes, CNN does

Thank you!

Questions?