Image based Indian Monument Recognition using Convolutional Neural Networks

Aradhya Saini, Tanu Gupta, Akshay Kumar Gupta, Ankush Mittal

IEEE Conference on Big Data,IoT and Data Science 2017

Content

Introduction

1. Aim of paper is to classify 100 different monuments .
2. The proposed paper mainly aims at analyzing the archaeological monuments for its visual features to help in automating the process of identifying the monuments and to retrieve the similar images for studying art forms in greater details.
3. Paper has been implemented using classification algorithm and Convolution Neural Network.

Dataset Overview

100 classes of 100 Indian Monuments

Proposed Methods

1. Histogram of Oriented Gradients

- HOG are feature descriptors used for object detection.
- It represents objects as single feature vector as opposed to feature vector where each represents a segment of image.

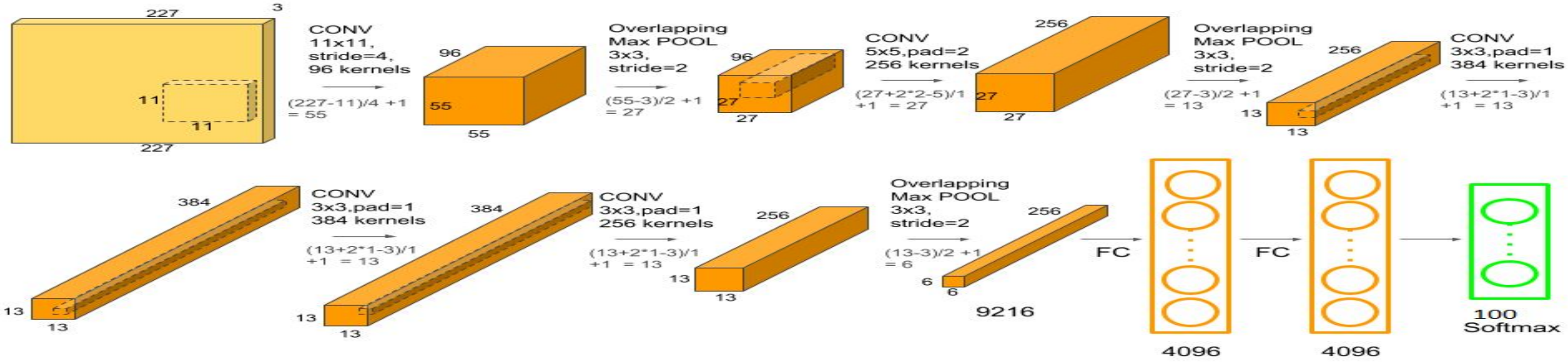
Proposed Methods (Continued)

2. Local Binary Patterns(LBP)

- Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.
- It represents objects as single feature vector as opposed to feature vector where each represents a segment of image.

Proposed Methods (Continued)

3. Convolutional Neural Network



Implementation

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_36 (Conv2D)           (None, 224, 224, 32)      896
_________________________________________________________________
activation_48 (Activation)   (None, 224, 224, 32)      0
_________________________________________________________________
conv2d_37 (Conv2D)           (None, 222, 222, 32)      9248
_________________________________________________________________
activation_49 (Activation)   (None, 222, 222, 32)      0
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 111, 111, 32)      0
_________________________________________________________________
dropout_20 (Dropout)         (None, 111, 111, 32)      0
_________________________________________________________________
conv2d_38 (Conv2D)           (None, 111, 111, 64)      18496
_________________________________________________________________
activation_50 (Activation)   (None, 111, 111, 64)      0
_________________________________________________________________
conv2d_39 (Conv2D)           (None, 109, 109, 64)      36928
_________________________________________________________________
activation_51 (Activation)   (None, 109, 109, 64)      0
_________________________________________________________________
max_pooling2d_21 (MaxPooling (None, 54, 54, 64)        0
_________________________________________________________________
dropout_21 (Dropout)         (None, 54, 54, 64)        0
_________________________________________________________________
flatten_8 (Flatten)          (None, 186624)            0
_________________________________________________________________
dense_20 (Dense)             (None, 512)               95552000
_________________________________________________________________
activation_52 (Activation)   (None, 512)               0
_________________________________________________________________
dropout_22 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_21 (Dense)             (None, 30)                15390
_________________________________________________________________
activation_53 (Activation)   (None, 30)                0
=================================================================
Total params: 95,632,958
Trainable params: 95,632,958
Non-trainable params: 0
```

Softmax Function at last layer

1. Given a vector of real numbers, it converts each value to a probability,
2. It gives the P(y=class |x).
3. The output node with maximum probability is assigned the corresponding class.

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

Error Function- Categorical Cross Entropy Loss

P(A) - Probability of Happening of event A

I(A)= -log(P(A)), Information Content of A

$$\text{Entropy} = -\sum_{i} P_i \operatorname{Log}_2 P_i$$

Loss Function

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

O- Observed/predicted value

C- Class Label, M Classes

y - Binary value(0/1), 1 if correct classification

p- probability values after softmax

Result Of Paper

Technique	Accuracy %
HOG+SVM	1.47
HOG+Random Forest	1
HOG+KNN	1
LBP+SVM	7.23
LBP+Random Forest	14.27
LBP+KNN	20.09
GIST+SVM	1
GIST+Random Forest	1
GIST+KNN	1
CNN fc6	**92.7**
CNN fc7	**90.60**
CNN fc6+fc7	**91.82**

Result Of Our Implementation

```
Train on 1393 samples, validate on 155 samples
Epoch 1/25
1393/1393 [==============================] - 9s 6ms/sample - loss: 4.1074 - acc: 0.0488 - val_loss: 3.2819 - val_acc: 0.1097
Epoch 2/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 3.1122 - acc: 0.1249 - val_loss: 2.9160 - val_acc: 0.2194
Epoch 3/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 2.7540 - acc: 0.2347 - val_loss: 2.5267 - val_acc: 0.2903
Epoch 4/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 2.3501 - acc: 0.3338 - val_loss: 2.3414 - val_acc: 0.3226
Epoch 5/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 1.9645 - acc: 0.4422 - val_loss: 2.0591 - val_acc: 0.4323
Epoch 6/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 1.3975 - acc: 0.5958 - val_loss: 2.1793 - val_acc: 0.4065
Epoch 7/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.8955 - acc: 0.7444 - val_loss: 2.0853 - val_acc: 0.4258
Epoch 8/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.5090 - acc: 0.8600 - val_loss: 2.0987 - val_acc: 0.5161
Epoch 9/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.3153 - acc: 0.9067 - val_loss: 2.4577 - val_acc: 0.4710
Epoch 10/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.2722 - acc: 0.9203 - val_loss: 2.2383 - val_acc: 0.4645
Epoch 11/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.1771 - acc: 0.9605 - val_loss: 2.3393 - val_acc: 0.4903
```

Result Of Our Implementation(Continued)

```
Epoch 13/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0964 - acc: 0.9742 - val_loss: 2.8564 - val_acc: 0.4581
Epoch 14/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.1436 - acc: 0.9605 - val_loss: 2.4593 - val_acc: 0.4903
Epoch 15/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0990 - acc: 0.9691 - val_loss: 2.6509 - val_acc: 0.4968
Epoch 16/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.1130 - acc: 0.9698 - val_loss: 2.9003 - val_acc: 0.4903
Epoch 17/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.1207 - acc: 0.9670 - val_loss: 2.5443 - val_acc: 0.5226
Epoch 18/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0965 - acc: 0.9698 - val_loss: 2.5682 - val_acc: 0.4903
Epoch 19/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0693 - acc: 0.9821 - val_loss: 2.5975 - val_acc: 0.5290
Epoch 20/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0934 - acc: 0.9734 - val_loss: 2.3170 - val_acc: 0.4839
Epoch 21/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0451 - acc: 0.9885 - val_loss: 2.9633 - val_acc: 0.4774
Epoch 22/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0893 - acc: 0.9734 - val_loss: 2.6635 - val_acc: 0.4839
Epoch 23/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0713 - acc: 0.9799 - val_loss: 2.8317 - val_acc: 0.5032
Epoch 24/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0522 - acc: 0.9835 - val_loss: 2.8881 - val_acc: 0.5032
Epoch 25/25
1393/1393 [==============================] - 8s 6ms/sample - loss: 0.0476 - acc: 0.9892 - val_loss: 2.8924 - val_acc: 0.5032
```

Challenges

- Main challenge was to collect dataset.
- It was very difficult how to implement paper on Deep Learning Framework.
- Hyperparameter Tuning in itself is a challenge, it was not mentioned by the author.

References

1. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks” in NIPS 2012
2. Kalliatakis, G. and Triantafyllidis, G., 2013. Image based Monument Recognition using Graph based Visual Saliency. ELCVIA, 12(2), pp.88-97.
3. Yaligar, S., Sannakki, S. and Yaligar, N., 2013. Identification and Retrieval of Archaeological Monuments Using Visual Features.