NEA Investigation Into Facial Recognition

Luca Djehiche

candidate Number: 0611
—
Centre Name: Barton Peveril College
Centre Number: 58231
—
GitHub Page:
https://github.com/LucaDjV/NEA-Investigation-into-facial-recognition

Contents

1 Analysis

1.1 Statement Of Investigation

Can I write a facial recognition algorithm that can be run on video at 15 frames per second?

I plan to investigate Machine Learning techniques and dynamic programming by developing a facial recognition algorithm in which a program will receive an input of an image and decide whether it is a cat or not.

To do this I will need to consider what constitutes a successful algorithm, do research into optimisation techniques and understand the mathematics behind the models.

The program will be tested in the end with a set of test data where I will measure the speed and accuracy of the solution

1.2 Background

1.3 Expert

1.4 Interview

1.4.1 *Interview Questions*

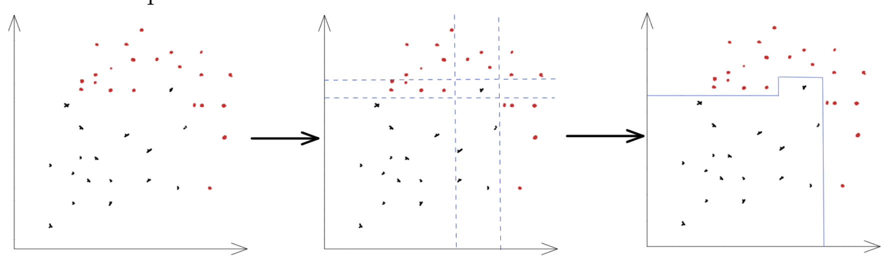
1. **What approaches are there to facial recognition?**
2. **What optimisation techniques have you seen?**
3. **Should I attempt to support multiple bit depths and picture formats?**
4. **What is the best way of storing and reading from a dataset?**
5. **What aspects of a program like this do you think make it successful?**
6. **What should I research before I start programming?**
7. **What would you say is the easiest way of tuning my model?**
8. **What should I focus on to have a successful project?**

1.4.2 *Interview Evaluation*

1.5 Initial Research

1.5.1 *Viola Jones Algorithm*

This approach considers the texture and shading of the object you are trying to identify. It relies on the designer identifying the key features of the object in terms of the areas that are on average darker or lighter than their surroundings in what's called a haar cascade[1]. Viola Jones uses **AdaBoost** to generate a forest of stumps that classify the item through each stump's vote. AdaBoost is a useful algorithm for decision trees as it can turn an array of linear classifications into a non linear form which is useful as data in reality is not always linearly split. For example:



1.5.2 *AdaBoost*

AdaBoost is a popular method for training a model based off of decision trees. The steps for AdaBoost are:

1. Add a new value to each record which is the record's record weight. This is calculated as
$$\frac{1}{Amount\ Of\ Records}$$

2. Generate stumps for each field in the dataset that may or may not be indicative of the correct classification. 'A stump is a tree with just 1 node and 2 leaves'[2]

3. Then use the stumps to get the result of each stump's classification

4. The stumps will produce a range of values that may or may not be correct for each record. Sum the correct and incorrect classifications for each datapoint in each field and then work out the Gini index which is:
$$1 - \left[\left(\frac{AmtIdentifiedCorrectly}{TotalNumberConsidered}\right)^2 - \left(\frac{AmtIdentifiedIncorrectly}{TotalNumberConsidered}\right)^2\right]$$

5. The stump with the lowest Gini index is then the most successful stump.

6. If the stump has a Gini index of 0, that is your correct classifier and the algorithm does not need to run any further.

7. Work out the vote for each stump with formula below where the total error is the sum of all of the record weights of the records that the

$$Vote = \frac{1}{2} log \left(\frac{1 - TotalError}{TotalError} \right)$$

   (equation from [2])

8. The vote indicates how accurate the stump is where an vote close to 0 says that the field has a 50 percent chance of outputting a correct classification and a positive vote tells us that there is some link between the stump and giving us a correct classification where the larger the number is, the better at classifying the data. A negative vote would therefore indicate that the stump mostly gets it wrong and therefore there may be an inverse relationship between the field and the correct classification.

9. Then, work out the new record weights of each record. If the record was incorrectly classified by the most successful stump, increase the record weight, if it was correctly classified by the stump, decrease the record weight.

   - to increase weight:

   $$NewWeight = RecordWeight \times e^{Vote}$$

   - to decrease weight:

   $$NewWeight = RecordWeight \times e^{-Vote}$$

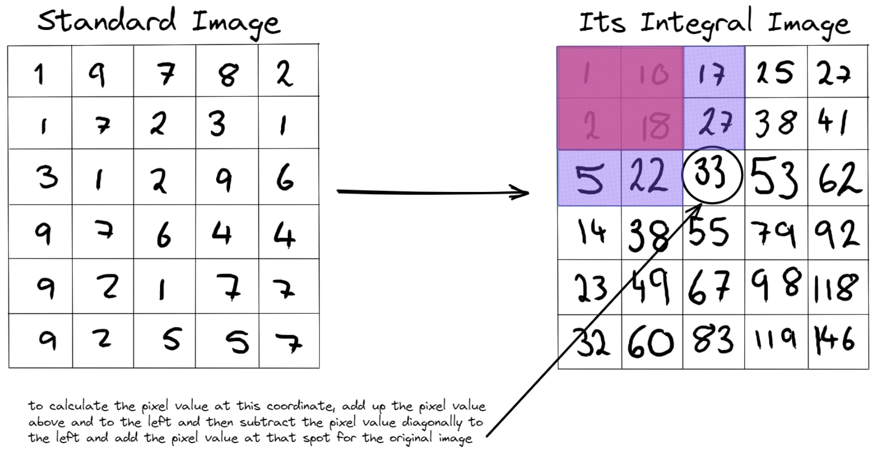
   (above equations from [2])

10. Then replace the record weights with the new record weights and normalise them by dividing by the sum of all the new record weights. Then start the algorithm from the beginning to get a forest of weak learners that will classify the data.

11. The forest can be used by running all trees to get those that give a true value as their answer and those that give false, summing the amounts of say of the stumps in each category and choosing the answer that is supported by the largest total vote.

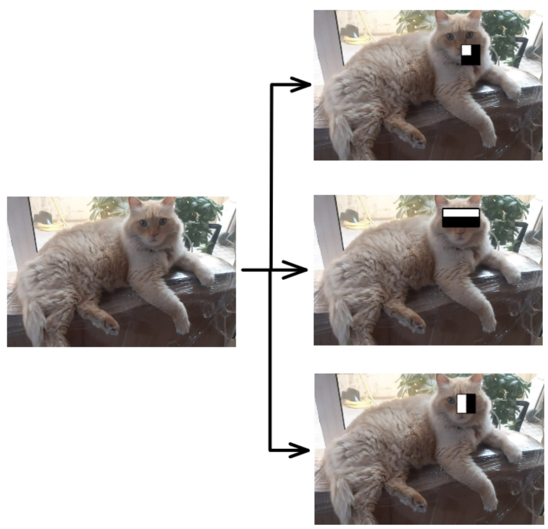
1.5.3 *Integral Image*

An integral image is a way of representing an image where each pixel is the sum of all of the pixels before it, this allows us to quickly and efficiently sum all of the values of the pixels in a block and subtract those that came before.



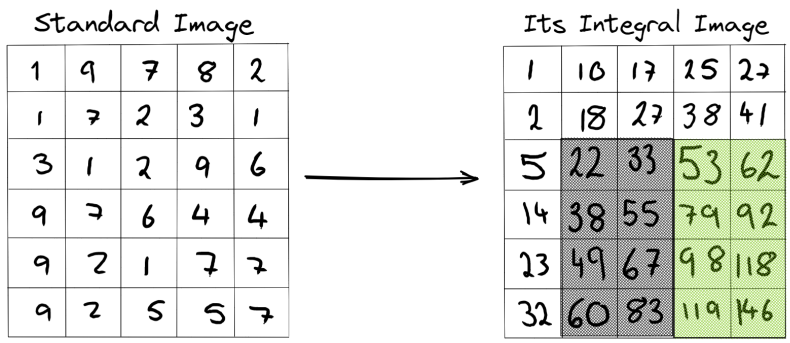
1.5.4 *Haar Features*

Haar features are a substitute to the edge detection kernels and consider the grading of the section compared to another for example a haar kernels could be:



Computing Haar Feature Extraction:

we can compute haar features in minimal calculations from an integral image rather than a standard image where you would have to perform many additions and subtractions.



to work out the intensity difference of the green section (high intensity section) and the black section (low intensity section) we do:

$$(83 - 32 - 27 + 2) - (146 - 83 - 41 + 27) = 23$$

so the difference between the sections is 23 and therefore this may represent a useful feature. We used 7 operations as opposed to 15 operations in the standard method without an integral image. The amount of operations for a comparison between 2 sections is constant but the amount of comparisons without an integral image scales with the box sizes

1.6 Prototype

1.7 Further Research

1.8 Objectives

1.9 Modelling Of Problem

2 Design

3 Testing

4 Evaluation

5 Technical Solution

6 References

References

[1] Paul Viola and Michael Jones' paper: Rapid Object Detection Using A Boosted Cascade Of Simple Features

[2] Definition of stump taken from Josh Starmer's video on AdaBoost, used as one of many sources to learn the steps for AdaBoost: https://www.youtube.com/watch?v=LsK-xG1cLYA