Big Data Analytics Plankton Classification

asutosh satapathy

Initial Setup

- Started with Java
  - Limited Libraries for image processing.
  - Extremely Slow.
- Developed the application with Python
  - Fast Image processing Libraries
    - scikit-image
    - mahotas
    - OpenCV
  - Extremely fast Numerical Processing
    - numpy

Image Pre-processing

- Pre-processing actions
  - Thresholding images by mean value
    - This reduces noise.
  - Segmenting
    - Dilate image to connect to neighboring pixels
    - calculate labels for connected regions
    - apply threshold to unlabeled regions
  - Extracting regions
- Choose the largest segment based on area
- Rescale the images to the same size of 25X25
  - Large images ranging from 70X100 to 200X300

Feature One: Width to Height Ratio

After viewing the mages, I came to the conclusion that, a lot of the images can be can be easily classified to their correct classification based on their width to height ratio. A lot of images have specific shapes and they are in constant proportion in all images.

Efficiency: 43%

Feature Two: Geometrical properties

As we can see from the features, these are all geometrical properties of the image. We have had a lot of success from classifying the images just from the width to height ratio before. These all additions should improve the performance by a lot. This makes a lot of sense as the images vary a lot and can be easily distinguished by their geometrical shapes. A few of the classes have very similar shapes and can't be distinguished. We can identify these classes later and try to address them by some other method.

The efficiency increased by 2% from 43% to 46%

Efficiency: 43%

Feature Three: Hu Moments

The non-orthogonal centralised moments are translation invariant and can be normalised with respect to changes in scale. However, to enable invariance to rotation they require reformulation. Hu described two different methods for producing rotation invariant moments. The first used a method called principal axes, however it was noted that this method can break down when images do not have unique principal axes. Such images are described as being rotationally symmetric. The second method Hu described is the method of absolute moment invariants and is discussed here. Hu derived these expressions from algebraic invariants applied to the moment generating function under a rotation transformation. They consist of groups of nonlinear centralised moment expressions. The result is a set of absolute orthogonal (i.e. rotation) moment invariants, which can be used for scale, position, and rotation invariant pattern identification. These were used in a simple pattern recognition experiment to successfully identify various typed characters.

We can use this to see if the performance of our application improves.

Hu Invariants effectively increase **efficiency by 2%** with a very slight tradeoff on performance/speed.

Feature Four: Linear Binary Patterns

Local binary patterns (LBP) is a type of feature used for classification in computer vision. LBP is the particular case of the Texture Spectrum model.

The LBP feature vector, in its simplest form, is created in the following manner:

- Divide the examined window into cells (e.g. 16x16 pixels for each cell).
- For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
- Where the center pixel's value is greater than the neighbor's value, write "1". Otherwise, write "0". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
- Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center).
- Optionally normalize the histogram.
- Concatenate (normalized) histograms of all cells. This gives the feature vector for the window

This has increased the **efficiency by 1%.**

End

THANK YOU

References

1. https://www.kaggle.com/c/datasciencebowl/details/tutorial
2. https://en.wikipedia.org/wiki/Local_binary_patterns
3. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/SHUTLER3/node8.html
4.