CS5014 Learning Visual Attributes

Preprocessing

Understanding the Data

The GQA dataset provided a large amount of annotated images. It provided the following attributes:

- **The Image ID**

A unique identifier number for the image

- **The object ID**

A unique identifying number for an object within an image

- **The position (x,y)**

describing the location of the object's bounding box (an object's bounding box is a square enclosing its location on an image).

- **Width and Height**

Describing the width and height of the bounding box

- **Colour**

A categorical description of the object's colour

- **Texture**

A single word description of the texture of the object

- **Colour histograms from the bounding box, in the CIELAB colour space**

The human eye doesn't always perceive colours accurately. The same colour (i.e. light of the same wavelength) could be perceived differently according to lighting, or viewing angle. The CIELAB colour space is designed to approximate human vision (Wikipedia Contributors, 2019a), more accurate than traditional colour spaces such as RGB.

Each bounding box is split into 9 regions. Each region has three values corresponding to the components of CIELAB. Giving 27 values in total.

- **Histogram of oriented gradients (HOG)**

This is a feature descriptor. It counts occurrences of gradient orientation in parts of an image and identifies the main edge/gradient in each of those portions of the image (Wikipedia Contributors, 2019b)(e.g. Appendix: Example HOG). To over-simplify, it detects edges in an image.

Once again the bounding box is split into 9 parts, and each of these parts is split again into a 32 part HOG to give a total of 288 values for each image.

- **A set of Complex Cell responses based on Gabor filters**

A Gabor filter is a filter used in edge and texture analysis. If a Gabor filter is oriented in a direction, it will have a strong response for locations of the target image that have structures in this direction (Joshi, 2014). They are useful because they resemble the human visual system in how they distinguish objects

Each bounding box is split into 9 sections with 14 values for each section, to give a total of 126 values for each object.

Splitting, Scaling and Selection

First, the Image ID and the Object ID can be discarded. This is because this data does not describe or have a correlation to information in the images in any fashion, it is an arbitrary number used to identify a particular object in a particular image.

Similarly, the position of the bounding box can also be discarded. The position of an object within an image should have no bearing on its colour or texture, or its significance when using that particular data point to make a model. The same is true for the width and height of the bounding box.

Then all the rows that have missing values are removed, and Label Encoding is used to change the categorical colour and texture into discrete numerical values. A MinMaxScaler was used to normalize the values. Normalization was chosen because the CIELAB colour space, HOG and BIMP values all have well-defined limits with no reason for the values to be grouped, so the values are likely spaced out.

There were also missing values in the target data. These values were guessed through iterative imputation. Each column with missing values was treated as an output y, which was a function of the other values.

Model Design and Implementation

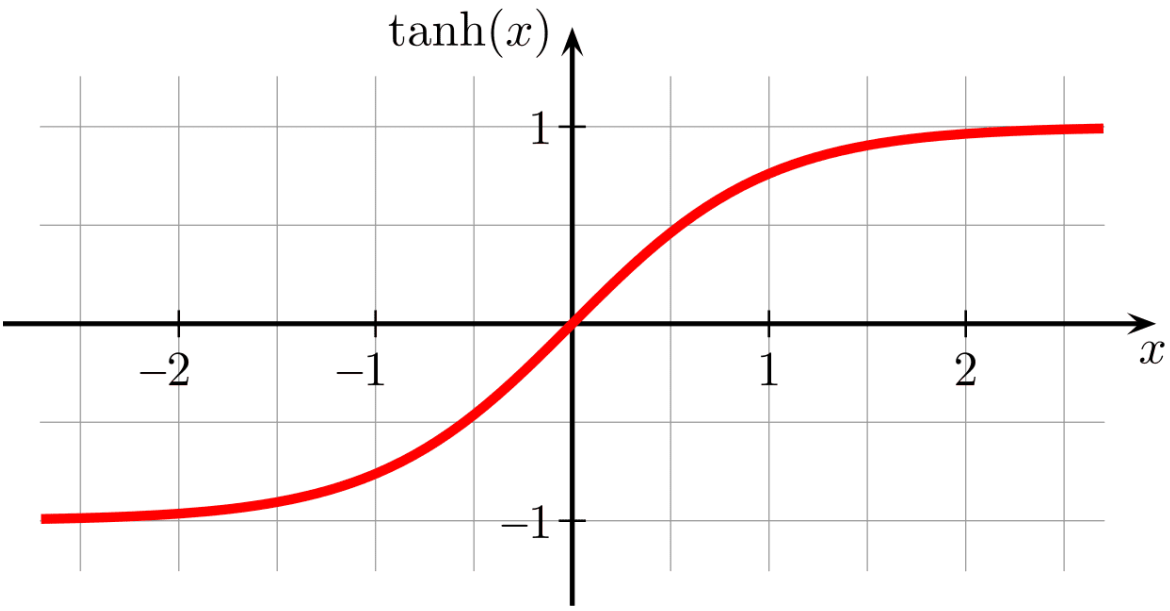
The MLPClassifier, an implementation of a neural network in sklearn was used for this project.

Hyper Parameters

To choose hyperparameters cross-validation was used to test which parameters gave the best results. GridSearchCV was used to split and test the different parameters with the training data.

The activation function is the mathematical equation that describes the output of a neural network's nodes. The 'tanh' and 'relu' functions were tested as they are the most widely used in neural networks (of the options available). The tanh function is the hyperbolic tan function with the following values.

Illustration of Tanh (Maladkar, 2018)

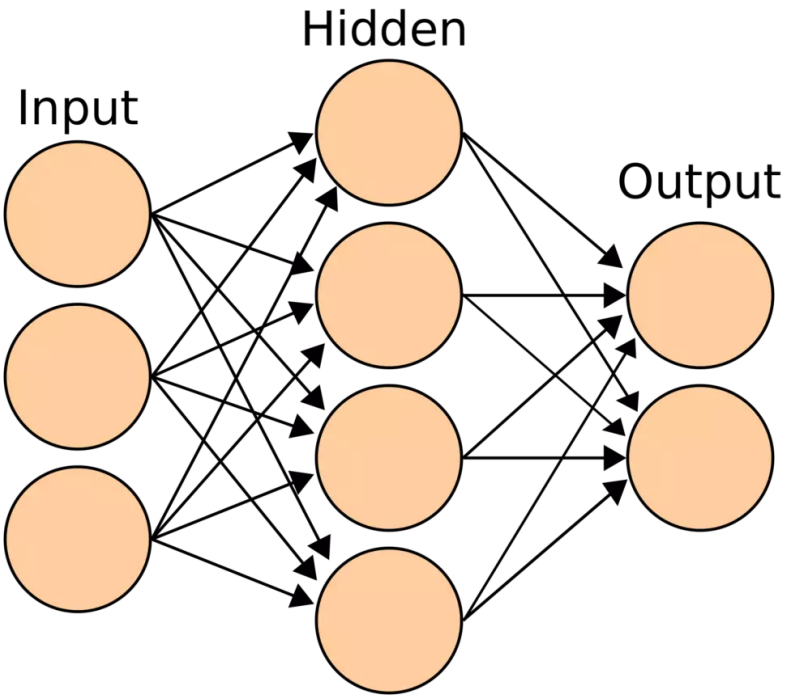


The reLU function is also known as rectified linear unit and has the definition: f(x)=max(0,x) (Wikipedia Contributors, 2019c). These functions are applied to the inputs X to get the output of neural network. The reLU function was used for the model.

The solver parameter is the optimization method used. The model made uses the ‘adam’ method, a variation of stochastic gradient descent that has an adaptive learning rate for the different parameters of the neural network (Ranjan Rath, 2020).

The ‘alpha’ value represents the regularization used by the model i.e. how it penalizes more complex models. Both the model for colour and the model for texture use ridge regression (L2 regularization). The model for colour has an alpha of 0.00001, and the texture model has an alpha of 0.001.

The hidden layer refers to the layers of a neural network that take in data. The models I used had only 1 hidden layer with 100 nodes.



Example Neural Network(Daityari, 2019)

Optimization Strategy

The models I employed use ‘Adam’, a variation of stochastic gradient descent. Stochastic Gradient Descent tries to minimize a loss function. It does this by stepwise taking in values (at random positions) of the training set and updating the current model. It has a constant learning rate. The ‘Adam’ method is an extension that uses a learning rate that is an exponential moving average of the gradient and the square and has additional parameters to control how these moving averages decay, so early learning steps are larger (to increase the speed of the process) and later learning steps are smaller (to avoid missing the minimum) (Jason Brownlee, 2017). This ultimately results in an optimization strategy faster than normal stochastic gradient descent. The disadvantage of Adam is that research suggests it has a

weight decay problem that causes models using Adam to not generalize as well as other models (Alabdullatef, 2020)(Graetz, 2020).

Adam was used for the models due to the large number of features in the input data(27 for colour, 400> for texture). Stochastic gradient derived methods typically converge faster than other optimization methods.

Evaluation Metric

The accuracy was used as an evaluation metric. This is measured (number of correct predictions)/(number of total predictions). Accuracy is best used in balanced datasets, these models are not balanced as several classes are more prevalent in the training data than others (e.g. red, blue and green for colours, fluffy for textures). As a result, it is a good idea to use a range of metrics, preferably a confusion matrix to capture factors such as the false positive/false negative rate for different classes (See Appendix for Confusion Matrices.)

The accuracy score for both models was only around 0.43. This is only slightly better than correctly guessing the most prevalent class in the training data for both models. The most prevalent colour was white, at 0.334 (3 s.f) proportion of the total classes, and the most prevalent texture was fluffy, at 0.382(s.f.) proportion of the total classes. The accuracy for the color model was more than 10% higher than just guessing the most prevalent class, and 5% higher for the texture model. Suggesting its not a statistical anomaly.

Some classes were easier to identify because there were more instances of those classes so there were more data points for the model to ‘understand’ the features associated with those classes.

Comparison to Logistic Regression

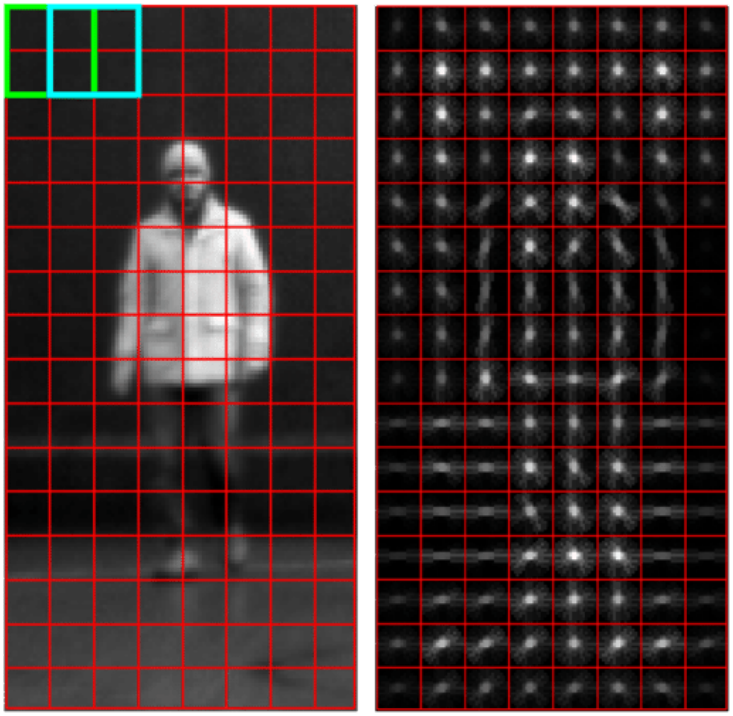
Accuracy	MLPClassifier	Logistic Regression
Colour	0.4550898203592814	0.4251497005988024
Texture	0.4311377245508982	0.4281437125748503

Accuracy values of Models for Colour and Texture, MLPClassifier vs Logistic Regression

The Logistic Regression model used the default settings, with the same data uses as an input for the MLP Classifier.

Appendix

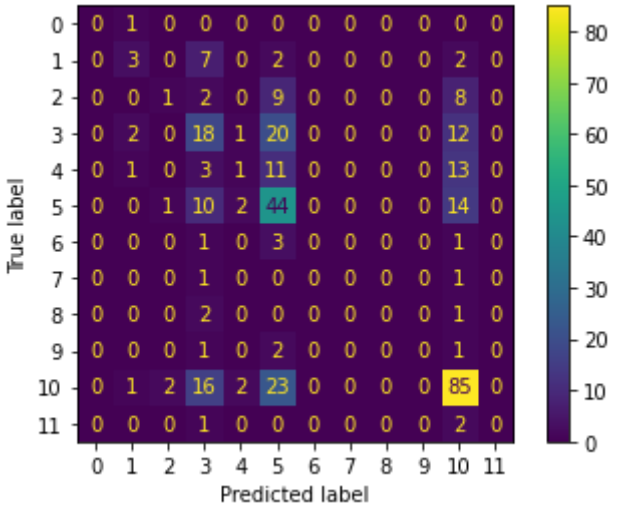
Example Histogram of Oriented Gradient



(a) **(b)**

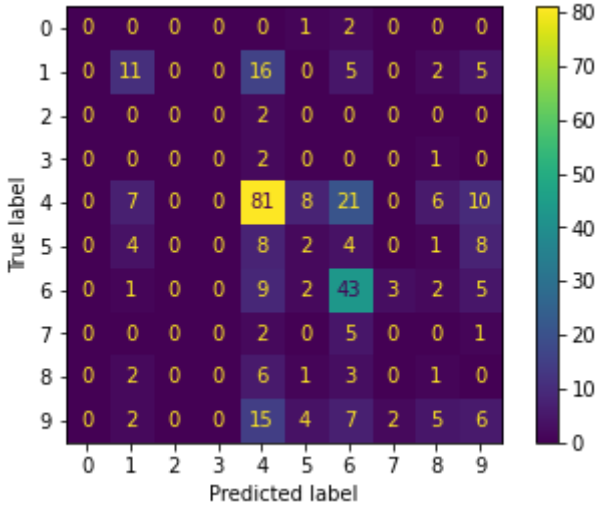
(Fotiadis, Garzón and Barrientos, 2013)

Confusion Matrices



Key:

1 Beige	2 Black	3 Blue
4 Brown	5 Gray	6 Green
7 Orange	8 Pink	9 Purple
10 Red	12 White	13 Yellow



Key:

1 Barbed	2 Blurry	3 Coarse
4 Crusty	5 Fluffy	6 Fuzzy
7 Grassy	8 Gravel	9 Rippled
10 Smooth		

Reference list

Alabdullatef, L. (2020). *Complete Guide to Adam Optimization*. [online] Medium. Available at: https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d [Accessed 16 Apr. 2021].

Daityari, S. (2019). *A Beginner's Guide to Keras: Digit Recognition in 30 Minutes - SitePoint*. [online] www.sitepoint.com. Available at: https://www.sitepoint.com/keras-digit-recognition-tutorial/.

Fotiadis, E., Garzón, M. and Barrientos, A. (2013). Human Detection from a Mobile Robot Using Fusion of Laser and Vision Information. *Sensors*, 13(9), pp.11603–11635.

Graetz, F.M. (2020). *Why AdamW matters*. [online] Medium. Available at: https://towardsdatascience.com/why-adamw-matters-736223f31b5d [Accessed 16 Apr. 2021].

Hansen, C. (2019). *Optimizers Explained - Adam, Momentum and Stochastic Gradient Descent*. [online] Machine Learning From Scratch. Available at: https://mlfromscratch.com/optimizers-explained/#/ [Accessed 16 Apr. 2021].

Jason Brownlee (2017). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/.

Joshi, P. (2014). *Understanding Gabor Filters*. [online] PERPETUAL ENIGMA. Available at: https://prateekvjoshi.com/2014/04/26/understanding-gabor-filters/ [Accessed 13 Apr. 2021].

Maladkar, K. (2018). *Types Of Activation Functions in Neural Networks and Rationale behind it*. [online] Analytics India Magazine. Available at: https://analyticsindiamag.com/most-common-activation-functions-in-neural-networks-and-rationale-behind-it/ [Accessed 16 Apr. 2021].

Ranjan Rath, S. (2020). *Adam Algorithm for Deep Learning Optimization*. [online] DebuggerCafe. Available at: https://debuggercafe.com/adam-algorithm-for-deep-learning-optimization/.

Wikipedia Contributors (2019a). *CIELAB color space*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/CIELAB_color_space.

Wikipedia Contributors (2019b). *Histogram of oriented gradients*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients.

Wikipedia Contributors (2019c). *Rectifier (neural networks)*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Rectifier_(neural_networks).