F21DL PROJECT[12]

1. ANALYSING THE DATA SET

The data used is a subset of 12660 unique images from Stallkamp et al's *German Street Sign Recognition Benchmark*, down from 51839 images. Prior to analysing the data, the order of instances was randomised to prevent any a priori knowledge being introduced as bias to the models.

Figure 1 shows that the frequency of instances equal per class. For example, there are 2160 instances of the "Give Way" sign but there are only 240 instances of the "No speed limit general" sign.

Figure 2 shows that, across the data set, any given instance appears to cover the range of grey values as the distribution is negatively-skewed. However, using an assumption can be made that the images may contains some random noise, it is possible that the lightness values within a given image might cause the distribution of values for any given image to appear skewed. Using the interquartile range to evaluate the spread of values, as in Figure 2, indicates that the middle 50% of values are not as dispersed as the range might suggest. This implies that the detail which exists in each image might not be inherently obvious to the different training models as it made assume adjacent attributes are more likely to be noise.

2. NAÏVE BAYES NETWORKS

Data was prepared by first transforming output array to categorical variables and randomizing order of input and output data to exclude ordering bias. As Gaussian Naïve Bayes assumes data in each class is normally distributed input vectors were scaled individually to unit norm. Data was then split for train (70%) and test (30%) sets in each class to ensure split was uniform. Finally, a trained model was used to predict test set outcomes so the classification report and performance statistics could be displayed. As a result, randomized and normalized dataset containing full set of 2304 attributes correctly predicts the classification for 2067 of 3798 signs, with an F1-score of 0.562 as shown in Table 1.

Using Pearson's chi-squared test, each of the attributes out of the full set of 2304 was scored based on its correlation to the categorical output variable. As a result, the best 2, 5 and 10 attributes were selected for each class and combined into new train datasets overlapping into 14, 28 and 46 features respectively. An algorithm maximising the F1-score was run to select the best number of features per class resulting in top 3 pixels in each class overlapping into a new train dataset of 18 attributes correctly predicting 2324 of 3798 signs and increasing the F1-score to 0.618. The resulting model was validated using a 5-fold cross-validation showing no significant performance deviations. As shown in Figure 5, the top attributes were consistently located in the centre of the grid and around the edges of signs.

If the data was not randomized the results would promote bias relating to order of appearance of signs in dataset, for example this could be linked to time of day images were taken and would therefore impact the test/train split distribution. More importantly, as demonstrated, a non-randomized data-set yields poorer results than a randomized one. A non-normalized input data produces poor results as the distributions of values in each class appear skewed with a significant concentration of values in two separate sections of the grayscale (as shown in Figure 6).

---

[1] All contributors to the project contributed equally.
[2] Code and figures available at https://github.com/amitkparekh/hw-dmml-cw1

Interestingly, there appear to be little correlation between number of observations in each class and resulting respective F1-scores. Using produced confusion matrix (Figure 7), the easiest signs to recognise, in order, were *“Turn left down”*, *“Stop”* and *“Speed limit 80 lifter”*. Most difficult signs to recognise, in order, were *“Speed limit 60”*, *“Speed limit 80”* and *“Right of way in general*”. The model often confused *“Speed limit 60”* as *“No speed limit”*, *“Speed limit 60”* as *“Speed limit 80”* and *“Speed limit 80”* as *“Speed limit 60”*.

As in Figure 8, plotting the chi-square feature correlation values visualises which of the classes have more informative feature set in terms of absolute correlation. Remarkably, the aforementioned signs that the model has least problems identifying can be recognised in the visualised correlation values, while the categories which Naïve Bayes has most problems with are similar to each other. As expected the most informative features are located in the centre of images while the least – in the corners.

As demonstrated, selecting a small subset of attributes increases performance of Naïve Bayes both in resulting F1-scores as well as in processing time. In this example using 18 attributes proved more informative than the full set of 2304 which rightfully indicates high dependency of attributes in given dataset. Results of the optimisation show high F1-scores in low 2 or 3 best features per class, then sinking significantly to increase to a local maximum around the best 900 features mark and decreasing again as number of features approaches maximum (as in Figure 9).

3. COMPLEX BAYESIAN NETWORKS

In the context of image recognition, one may ask if the independence assumption is realistic. Logically, for a given pixel, if the surrounding pixel values are all close to 255, then the probability of that pixel being 0 should be lower than it being 255. Under Naïve Bayes, the independence assumption forced these probabilities to ignore any correlated attributes. As such, the probability parameters were solely estimated based on the class. The main purpose of building more complex Bayesian networks is to capture these significant dependencies among attributes and use more accurate conditional probability tables.

Compared with the Naïve Bayes metrics in Table 2, Table 3 shows that all the search algorithms improved each model’s F1 score by roughly 0.2, suggesting that establishing non-class dependencies is a critical aid.

What are the new structures?

For each search, the network was initialised as Naïve Bayes Directed Acyclic Graph (DAG). The local search algorithms, without detailing nuance differences, look at each node and consider adding a parent node beside the class. Assuming two parents, the resulting structure is an expanded Naïve Bayes DAG with an additional parent for each node — discounting exceptions such as K2 not adding a parent for the first node in order, or when the Hill Climber removes the class. When observing the complex networks, an interesting observation is that most dependencies are established between nearby pixels. For instance, looking at the results on the best 10 data set, feature 1322’s parent is 1370. On a 48x48 dimension, these pixels are vertically adjacent. We can observe many similar dependencies such as 888 to 840 or 1751 to 1703.

Goodness of Fit vs. Complexity

Although increasing the number of parents boosts the model’s accuracy, it can also overfit to the training data. The more parents each node has, the more complex the network is, implying a risk of

learning irrelevant details. Theoretically, increasing the number of parents will reach a threshold at which the model's predictive capacity to unseen instances saturates. This is what we see when we increase the maximum parents using the Entropy score type with the best 2 data — F1 scores going from 0.806 to 0.832 and 0.839. The importance of choosing a scoring method with regards to striking a balance between goodness of fit and complexity should be noted. Some scoring methods are designed to penalise for the complexity by negating the log probability of the data given a structure, as Witten, Frank and Hall (2011) describe. In our tests, AIC seems to penalise so heavily that increasing the number of parents to 3 or above does not expand the network. However, when using Entropy score, the algorithm always prefers to add more parents. Finding the adequate number of parents should therefore be carefully considered.

4. CLUSTERING ALGORITHMS

K-means

In the first iteration of k-means, the class attribute was not included, and the numbers of clusters were kept as 10, since the number of actual classes was also 10. Instances assigned the same label are expected to be the same. Therefore, their distribution of pixels should also match and be in the vicinity of each other. As in Table 4, F1-score observed was very low at 0.223. Homogeneity and completeness scores are also considerably low, indicating that all the clusters contain instances of multiple actual types in them.

The inclusion of the actual labels to the input data before training the model should theoretically boost the clustering significantly. Since these values clearly indicate if two instances are of the same type or not, homogeneity and completeness should also improve drastically. However, although there is an increase in the F1 score, it is still not a good result with the F1 score at 0.250.

Gaussian Mixture Model (GMM)

K-means focuses solely on the mean distance of the centroid of each cluster to each of its corresponding instances. This distance-based decision-making results in k-means enforcing a circular boundary on the clusters. A Gaussian Mixture Model (GMM) on the other hand considers the overall distribution of instances as a mixture of gaussian distributions each with its own mean and variance. Based on the nature of the dataset and the way GMM operates, it should have an improvement over k-means. However, contrary to the expectation, the GMM performed slightly worse than the k-means algorithm for the same situations. As in Table 4, the homogeneity of the GMM, without inclusion of labels is 0.094. When including labels, the homogeneity is 0.102, which is still below the homogeneity of k-means when labels are not included.

Applying Principal Component Analysis (PCA)

Principal Component Analysis (PCA) can be used to reduce the dimensionality to the top n-dimensions, in the order of decreasing variance. In other words, after applying PCA on the data, the first dimension (or attribute) will be the one with the most variance, the second will be the one with second largest amount of variance, and so on. This is useful in the sense that we can filter out the dimensions that do not have much variance as they would not be contributing much to classification. Using PCA would enable the model to focus on attributes that contributed the most variance, and other columns which did not and would only add complexity would be filtered away.

As shown in Table 4, using PCA provided little to no increase in accuracy scores in most cases. There was an increase in accuracy by 3% when PCA was applied on enhanced data as compared to running k-means directly on enhanced data.

Overall we can see that K-means and other clustering algorithms perform poorly compared to Naïve Bayes and other supervised learning techniques. However, this is expected because k-means does not consider the distortions and noise in the individual images which aids in poor clustering.

The minor increases in accuracy when adding the output field is expected. Instances belonging to different classes will have an additional value added when considering the Euclidean distance between them. However, the Euclidean distance between two instances of the same class will not have that term (since the difference between the instances is 0 in that dimension). This way objects of the same class will appear slightly closer to each other.

In the case of the GMM, the presence of noise and distortion cause very small deviations in the values seen for each pixel. These minor deviations ultimately result in the algorithm misclassifying the instance as belonging to the wrong distribution.

PCA was used to filter out the dimensions with less variance with the expectation that it would enable the algorithms to focus on a smaller number of dimensions and thereby reduce the complexity of the task. This did not help much because of the same reason as above, the presence of noise which caused even the dimensions which were less significant in reality to appear as having high variance and get a higher ‘rank’ in terms of significance.

5. RESEARCH QUESTION

The topic in question is whether mutating the entire dataset can improve the model accuracy. When performed on the entire data set, the benefit is that it is computationally more efficient to transform images than building a more complex network.

Mutating the images

As shown in Figure 2, the interquartile range shows the spread of values per instance is generally lower. Using this metric, increasing the spread of values per instance could improve model accuracy. One way to increase the spread of value is to increase the contrast, that is increasing the distance between the greatest and lowest value.

One way of improving the contrast of an image is to equalise the histogram of the image, thereby rescaling all the values within the image so that the cumulative distribution function (CDF) of the images is uniform across the entire range (from 0 to 255) (see Figure 3). This method can be naïve as it simply alters the global contrast of the image without accounting for more local changes. A more reasonable approach to improve the contrast per image would be to locally increase the contrast for some given radius of pixels. Although other algorithms exist, such as Adaptive Histogram Equalisation (AHE), it increases the contrast of the histogram to a limit while also increasing the noise. Ultimately, as equalising the histogram primarily redistributes the data to follow a uniform distribution, it does not fully stretch every value to cover the entire spectrum.

Another way of increasing the contrast for an image is to apply thresholding, which creates a binary image — an image consisting of only 0 and 1 (or black and white). After choosing some fixed constant, thresholding the image would replace each pixel with a black pixel if it is below the constant, or with a white pixel if it is above the constant. Figure 3 shows that by thresholding the image, you lose a lot of the detail within the image.

One ideal method for this data set would be to combine the thresholding with histogram equalisation, accounting for local variations within the image without drastically increasing the noise. Using thresholding as inspiration, the contrast of each image was enhanced locally — using a radius of 2 pixels in order to restrict it to a small range. This method replaces the value of the pixels similarly to thresholding. However, it is not thresholding the values as it is replacing the values with either the local minimum or local maximum, and not either black or white. The purpose is to increase the contrast between the more distinct differences within the image; for example, on the thick lines on the signs. By using local contrast enhancement, it merged similar values together, reducing some of the noise within the image, As shown in Figure 3, locally contrasting the original image, the amount of random noise in the image has been reduced, and the black letters of the sign is more visible.

After locally enhancing the contrast, the histogram equalisation was applied locally by stretching the values between black and white within a given neighbourhood of pixels. As the local contrast of the image used a radius of 2 pixels, the radius was doubled to 4 pixels implying that every neighbourhood included at least two neighbourhoods of 2 pixels. Figure 3 shows this results in a brighter image but implying more noise in the areas of more solid colour (for example, the white area of the sign).

Combining the two approaches results in the final image within Figure 3, which combines the best of both thresholding and histogram equalisation. As shown, there still appears to be a lot of noise, but the noise does not appear to mask the more relevant attributed related to the sign itself in the middle of the image. Therefore, it is reasonable to expect that this mutation would improve the model accuracy.

Effects of image mutations on the machine learning models

Figure 4 shows that each model has improved after applying the mutations. The F1 score for Naïve Bayes has increased to be in line with both K2 and TAN without enhancement. When more constrained for time or resources, running Naïve Bayes with an enhancement can be as good as building a complex Bayesian network. Additionally, both K2 and TAN have F1 scores greater than 0.9. Although clustering algorithms had a much lower accuracy when compared to Naïve Bayes, applying the mutations improved their F1 scores to be greater than Naïve Bayes before mutation.

Applying the image mutations to the original data has improved the results of every model, therefore showing that the mutations applied are both valid and incredibly useful to the field of computer vision.

6. REFERENCES

- Witten I., Frank E. and Hall A., 2011 Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition, Amsterdam, Elsevier
- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans, K-means documentation
- https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation Evaluating K means
- https://jakevdp.github.io/PythonDataScienceHandbook/05.12-gaussian-mixtures.html, https://www.geeksforgeeks.org/gaussian-mixture-model/ Gaussian Mixture Model
- https://www.kaggle.com/alexisbcook/bar-charts-and-heatmaps Data visualization using seaborn.
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html Train-test split
- https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html Naïve Bayes
- https://en.wikipedia.org/wiki/Pearson%27s_chi-squared_test Chi-Squared test.
- https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.chi2.html Feature Selection
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html F1-Score
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html K fold cross validation
- https://en.wikipedia.org/wiki/Histogram_equalization Histogram equalization
- https://en.wikipedia.org/wiki/Adaptive_histogram_equalization
- https://en.wikipedia.org/wiki/Thresholding_(image_processing) Thresholding
- https://scikit-image.org/docs/dev/api/skimage.filters.rank.html#skimage.filters.rank.enhance_contrast Introducing contrast.