ML for ENG: Classification and Clustering Term Project Report

Soo Min Kwon*, Manish Kewalramani*, Brian Cheng*
*Rutgers, The State University of New Jersey
E-Mail: smk330@scarletmail.rutgers.edu

***Abstract*—In this paper, we will describe the methodology, results, and conclusions of our final project, in which we execute machine learning algorithms on three different types of datasets: an image dataset, a time series dataset, and a text dataset. We will compare their performance in terms of correctness (where applicable) and runtime. For each given dataset, we will provide recommendations to whether which algorithms and feature learning steps should go into production.**

***Index Terms*—machine learning, classification, clustering, image dataset, audio dataset, text dataset**

I. Introduction

In our project, we attempt to solve two machine learning problems: clustering and classification. The first problem was to take a dataset of ten different classes of images and cluster them into their respective classes without knowing the labels (unsupervised learning task). We used the CIFAR-10 dataset, which consists of $60,000$ RGB images with a resolution of 32×32 pixels [1]. The images were separated into ten classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. We discarded the labels and attempted cluster the images into ten clusters to see if the ten classes would each emerge as one of the ten clusters. The second machine learning problem was a binary classification problem - to identify whether a sound was made by a human or something else (i.e. not human sounds such as ringing of bells). Some examples of what we considered as "human sounds" were speech, whistling, breathing, heartbeat, clapping. The dataset we used was AudioSet by Google, a collection of ten-second sound clips drawn from YouTube videos [2]. The entire dataset contains 2.1 million clips, but for our project, we used $22,160$ samples for training and $20,371$ samples for testing. The next machine learning problem was also a binary classification problem - given a review for a movie, determine if the review had a positive or a negative sentiment. The dataset we used was a collection of movie reviews from the website IMDb [3]. A movie review on the IMDb website consists of a rating that ranges from one to ten stars, and text for the reviewer's thoughts. In the dataset, reviews with one to four stars were labeled as "negative" and seven to ten stars were labeled as "positive." Reviews with five or six stars were considered "neutral" and were not included in the dataset.

II. Methodology

In this section, we will be discussing the pre-processing and feature learning/extraction steps taken for each dataset that we took before using the machine learning algorithms. All algorithms for both tasks were implemented using the built-in package, Scikit-Learn [4]-[8].

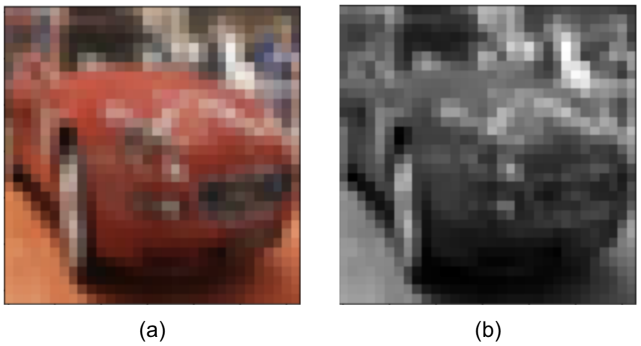

Fig. 1. Pre-processing steps for the CIFAR-10 Image Dataset (a) RGB Image (b) Grayscale Image After Pre-processing

A. Image Dataset - CIFAR 10

For the CIFAR-10 image dataset, the pre-processing steps that we took were converting the RGB images to grayscale and "vectorizing" our images before feature extraction. "Vectorizing" is a term that more or less arbitrarily stacks the elements in a high-dimensional array into a single vector. We converted the images to grayscale because working with RGB images are difficult in the sense that we have to make sure everything is reshaped correctly, as doing so otherwise would result in incorrect images and high-dimensionality. Reshaping one of our RGB images into a vector would result in dimensions of (3072×1), whereas grayscale images would have dimensions (1024×1). Using grayscale images would make computation more feasible and result in similar results in comparison to RGB images. We can observe the differences in an RGB images with a grayscale image in Figure 1.

The goal for the image dataset was to cluster the images into its respective classes. Instead of using three different distance metrics for K-means clustering, we decided to use three different feature learning steps for clustering with Euclidean distance. The three different feature learning steps that we took were Principal Component Analysis (PCA), Scale Invariant Feature Transform (SIFT), and both PCA and SIFT. The results are displayed in the following section.

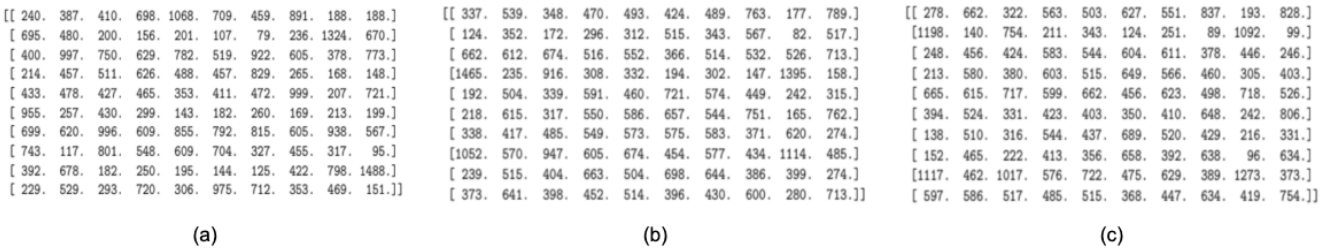

Fig. 2. Results for each feature learning step for clustering, where the rows represent the predicted cluster labels and the columns represent the true class labels (a) Confusion matrix with PCA as features (b) Confusion matrix with SIFT as features (c) Confusion matrix with SIFT and PCA as features

B. Time-Series Dataset - AudioSet by Google

For the AudioSet time-series dataset, the pre-processing steps that we took were converting the one-hot encoded labels to binary label encoding. The one-hot labels had 527 classes, each being the type of noise recorded in the audio clip. The label encoding pre-processing that we did changed the labels to "1" if the recording had human sounds, and to "0" if the recording had anything else (not human sound).

The goal for the time-series dataset was binary classification of either human-sound or not human-sound. The feature learning step was already completed before downloading the dataset, and more information for feature extraction can be found here [9]. In summary, the features were the output of a deep convolutional neural network, which resulted in 128-dimensional feature vectors. These vectors were the input to our machine learning algorithms discussing in the furthering sections.

C. Text Dataset - IMDb Movie Reviews

For the IMDb Movie Reviews text dataset, the pre-processing and feature learning steps that we took was to create a "bag of words" model and to change the "one-hot" encoded labels to binary encoded labels. As for the binary label encoding, we changed the labels to either be "1" for positive movie review, and "0" for negative movie review. The bag of words model tries to capture all the distinct words that appeared in each movie review. However, since we were working with a sizeable dataset, the bag of words model contained too many features (i.e. too many distinct words in each review) and slowed down computation signficantly. To combat this, we used Truncated Singular Value Decomposition (SVD) [10], which works similar to Principal Component Analysis, but for sparse matrices. Our "bag of words" model was sparse because many of the words that appeared distinctly only appeared in specific movie reviews, which resulted in having many zeros for words that did not appear. Once Truncated SVD was performed, we used three machine learning algorithms to classify whether the movie review was positive or negative.

III. Results & Discussion

In this section, we introduce the machine learning algorithms used for clustering and classification on the respective datasets and results.

For the CIFAR-10 dataset, we performed K-Means clustering, with K=10. The K-value was chosen to be 10, as there were a total of 10 different classes in our dataset. Once the algorithm was ran, we obtained a vector consisting of cluster labels for each image. Since we did not input any of the labels in the training process, the algorithm would have no idea what the true class labels were. For example, the 'bird' class in the true label vector can have value '5', but that does not necessarily mean that the cluster number for birds would be '5'. Rather, if the clustering was completed accurately, then there would be similarities in the patterns of the labels (i.e. if the true labels were $(1, 1, 2, 2)$, the predicted cluster labels could have been $(4, 4, 5, 5)$). Since this is rather difficult to identify, we instead made a confusion matrix where we iterated through each vector of cluster results and incremented the matrix by the respective entries. The results can be observed in Figure 2. In theory, the confusion matrix we would have hoped to see can be visualized in Figure 3. Figure 3 shows an ideal simulated confusion matrix, where one row would have a significantly higher value than the rest. However, since our matrices were not similar to the ideal matrix, we can conclude that our feature learning steps were not efficient, as the inherent features did not lend any useful information to be clustered correctly.

For the AudioSet by Google dataset, the three machine learning algorithms that we implemented for classification were Support Vector Machines (SVM), Linear Discriminant Analysis (LDA), and Quadratic Discriminant Analysis (QDA). For each algorithm, we recorded the run-time (training + cross validation), accuracy (for testing), recall, precision, and F1-score. Tne results can be observed in (a) of Figure 4. Amongst

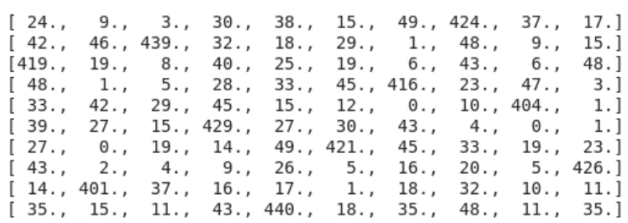

Fig. 3. Simulated Ideal Confusion Matrix

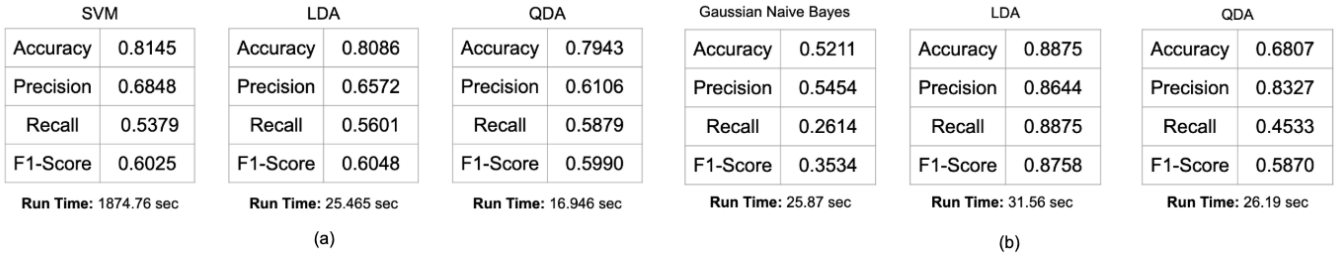

Fig. 4. (a) Results for each machine learning algorithm for time-series dataset (b) Results for each machine learning algorithm for text dataset

the three, we concluded that LDA was the best classification method given our pre-processing and feature extraction steps. All the algorithms had similar results, so the best algorithm was chosen based on the run-time. We can expect SVM to have a slower run-time, as it needs to use optimization methods (e.g. Gradient Descent) to solve the hyperplane parameters, whereas LDA and QDA have closed-form solutions. It had marginal better results than QDA, but as the computation time was similar, we concluded that LDA was the best for production.

For the IMDb Movie Reviews dataset, the three machine learning algorithms that we chose for classification were Gaussian Naive Bayes, LDA, and QDA. Similar to that of the AudioSet dataset, we reported scores that are displayed in (b) of Figure 4. For this classification task, we decided to use Gaussian Naive Bayes over SVM, as SVM took significantly longer to converge than the other two algorithms. However, the trade-off that we can expect is lower accuracy, as the assumption in Naive Bayes algorithms is that the conditional probabilities for each class are independent. From the results in Figure 4, we concluded that LDA was the best algorithm for production, as it exhibited a notable difference in scores compared to the other two algorithms.

IV. Conclusion & Future Works

From our experiments we conclude that clustering is not a good way to classify images. This indicates that there is nothing inherent about the classes of images used in the CIFAR-10 dataset that will lend them to become classified once they are grouped into clusters by their features. This is a bit counterintuitive, as one would expect any human given a subset of images from the CIFAR 10 dataset and asked to cluster them into 10 groups to cluster them based on their classes (since there are ten classes, it seems like a natural thing to do). We see from this however that we would need to teach the machine to think as a human before we can expect it to behave in human ways, hence why image classification is such a large problem to solve. For our timeseries dataset, the Audio clips, we conclude that the data may be linearly seperable but more rigor is required to accurately separate the data from our two classes. If we were to do further analysis, for example with a confusion matrix, our suspicion is that there would be a subset of human sounds that are relatively distinct and often correctly identified as human sounds, for example speech, and there would be a subset of human sounds that were more frequently incorrectly identified, like a human heartbeat, which we imagine is hard to distinguish from the heartbeat of a different animal. In the case of the text dataset, the movie reviews, we suspect that the dataset is linearly separable, and we could confirm this in a future work by verifying that we get similarly good results with another linear classifier like logistic regression. Our good results indicate that there is a strong correlation between usage of specific words and the overall sentiment of your review, which seems to make sense intuitively, as we would only use the words catastrophic or awful in a bad review. The poor performance of the Gaussian Naive Bayes classifier indicates that it is a very bad assumption to assume that the words are independent of each other.

References

[1] CIFAR-10 Image Dataset http://www.cs.utoronto.ca/kriz/cifar.html
[2] AudioSet by Google ttps://research.google.com/audioset/download.html
[3] IMDb Movie Reviews Dataset https://www.kaggle.com/iarunava/imdb-movie-reviews-dataset
[4] K-Means Clustering https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[5] Linear Discriminant Analysis https://scikit-learn.org/0.16/modules/generated/sklearn.lda.LDA.html
[6] Quadratic Discriminant Analysis https://scikit-learn.org/0.16/modules/generated/sklearn.qda.QDA.html
[7] Support Vector Machines https://scikit-learn.org/stable/modules/svm.html
[8] Gaussian Naive Bayes https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html
[9] VGGish Feature Extraction for Audioset by Google https://resources.wolframcloud.com/NeuralNetRepository/resources/VGGish-Feature-Extractor-Trained-on-YouTube-Data
[10] Truncated Singular Value Decomposition https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html