Objective: Closed-set Butterfly Classification using ResNet-101 Features (d = 2048)

Approaches:

Approach 1: Data is highly imbalanced. So, as my first approach I tried oversampling to generate synthetic data. I had one class which only had 1 sample. I had to duplicate that sample to be able to use SMOTE algorithm from imbalanced learning [1]. My k_neighbors=1 and sampling_strategy was 'minority'. I ran 60 iterations to generate 11292 samples.

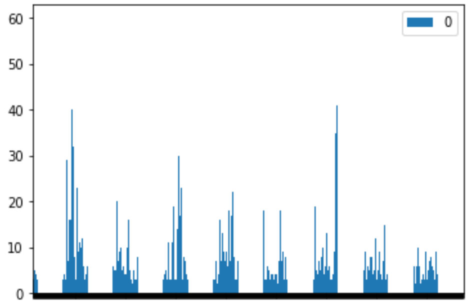

Figure 1: class frequency distribution of training images.

Approach 2: I have mostly used Python sklearn package models for training and prediction. Basic models like SVC, Logistic Regression gives an option to weight class samples. I used class_weight = 'balanced' to balance data. I have also tried different normalization technique to scale data. Minmax (0-1) normalization gave slightly better accuracy compared to Z-score normalization. I used minmax normalization for rest of my experiments. Following is a table which summarizes the hyper parameters I used for different approaches and best result I got on validation data for each approach.

No.	**Approaches**	**Hyper parameter tuning**	**Best Validation Mean Class Accuracy**

1	**SMOTE**	Solver = Liblinear
		PCA n = [250, 300,.., 1200]
		cost_param = [0.1, 1, 10, 100], penalty ='l2'
			0.71
			(Logistic Regression, n=1000, C=100)

2	**SVM (SVC, kernel='linear')**	PCA n = [150, 200, 250, 500, 750] C = [0.1, 1, 10, 100]	0.7490 (n = 250, C= 1)

3	**Logistic Regression**	Solver = Liblinear
		PCA n = [250, 300,.., 1950, 2000, 2048]
		cost_param = [0.1, 1, 10, 100], penalty ='l2'
			0.7357
			(n=1000, C=100)

4	**Ridge Classifier**	Works best with all (2048) features	**0.7829 (Best!)** And it is blazing fast!
5	**SGDClassifier**	PCA n = [500, 600, 700, 800, …., 1700, 1800]	**0.7790 (2nd best!)**

		alpha = [0.0001, 0.001, 0.01, 0.1]	(n= 1000, alpha = 0.0001)

6	**Student-t**	k = [0.1, 1.0, 10.0]
		m = [d+2, 2*d, 10*d, 100*d, 1e3*d, 1e5*d, 1e8*d, 1e10*d], here d = 500
			0.705
			(kappa=0.100, m=502)

Among my methods RidgeClassifier gave the best result. It uses Ridge Regression to perform multi-output regression in multi-class case [2]. SGD Classifier was working as a Linear SVM with Stochastic Gradient Learning. All classifiers seem to give better results compared to using oversampling method in the first approach.

Additional Approaches:

- **Cross-validation:** To get a more general model validated on different parts of the data, I have tried cross validation approach with k=5,6,7. My average accuracy was 0.78.
- **Voting Classifier:** I tried to combine three of my best models (RidgeClassifier, Logistic Regression and SGDClassifier) using Voting Classifier ensemble technique. I set the voting = ‘hard’ which uses predicted class labels for majority rule voting. Accuracy was 0.76 which is smaller than individual bests.
- **SVM (Support Vector counts):** I was trying to find an SVM model which gives decent validation accuracy but has lowest support vector counts. Within hyper parameter range PCA dim = [250, 300, .. , 950, 1000] and C = [1, 10, 25, 50, 100], I could not find an optimum model. Support vector total counts was within 7765 – 7791 and increased monotonically with PCA dim and C. Validation accuracy (0.73) was significantly lower than Logistic Regression.