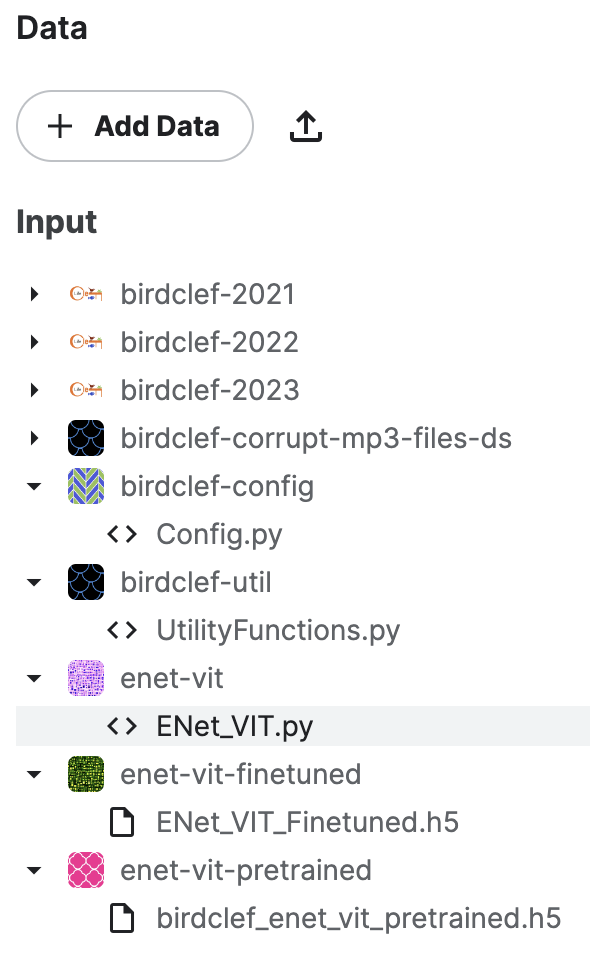
**READ ME**

1. The dataset used in all the notebooks and the .py files are imported from Kaggle competition.
2. All the notebooks and code will have to be executed on Kaggle. Use TPU-VM v3-8 accelerator for the pre-training and fine-tuning.CPU can be used to execute the inference module.
3. To test the models, import one of the model notebooks with extension .ipynb for example: Enet\_Transformer\_Hybrid.ipynb in Kaggle
4. Search and add the following kaggle datasets available through competition as inputs to the notebook.



1. Add the Code/UtilityFunctions.py as a dataset in kaggle inputs and give the title ‘birdclef-util’
2. Add the Code/Config.py as a dataset in kaggle inputs and give the title ‘birdclef-config’.
3. Add the model class file as a dataset on kaggle for example, the class file for ENet\_VIT model is in the Code/ENet\_VIT/ENet\_VIT.py file and give the title ‘enet\_vit’.
4. Add the pre-trained model .h5 file as a dataset on kaggle for example, the pretrained file for ENet\_VIT is in the Code/ENet\_VIT/birdclef\_enet\_vit\_pretrained.h5 file and give the title ‘enet-vit-pretrained’
5. Add the fine-tuned model .h5 file as a dataset on kaggle for example, the fine-tuned file for ENet\_VIT is in the Code/ENet\_VIT/ENet\_VIT\_Finetuned.h5 file and give the title ‘enet-vit-finetuned’
6. After uploading all files, the kaggle/inputs will look like the following



1. The BirdCLEF-EDA fie is also available as a notebook. This can also be imported with the above instructions and tested for pre-processing steps.
2. The code for pre-processing and data preparation pipeline is referenced and modified from <https://www.kaggle.com/code/awsaf49/birdclef23-pretraining-is-all-you-need-train>.
3. The main changes are the split in dataset for fine-tuning, validation and testing. Data is divided inti five folds and Fold 0 is used for validation and Fold 4 is used for testing. The other folds are used for training during fine-tuning.
4. Each model .py class contains the functions to build, pre-train, fine-tune and test the models. The files are ENet\_VIT.py, VGG16\_RCNN.py, Mobilenet\_CNN.py, ENet\_CNN.py
5. Config.py file contains configuration settings for the project. However, some of the variables are overridden by .ipynb files or the parameters sent to the model functions.
6. Utility.py has all the common pre-processing functions used by all the team members.
7. The predictions.csv and missed\_predictions.csv contain the testing predictions of each models.
8. The link to the project code and .h5 files are at <https://drive.google.com/drive/folders/1fPB2NLce-BuDaUycfpGUDuDE9z3WkdZu?usp=share_link>

**Team member contribution**

Ahaz Bhatti

Literature review, dataset finalization, VGG16-RCNN model development, project report parts including motivation, background, part of literature review, and the VGG16-RCNN model architecture and results

Nilisha Makam Prashantha

Literature review, ENet-CNN model development, project report parts including Data Collection, part of literature review, and the ENet-CNN model architecture and experiment results, slides editing, references

Ha V Phan

Literature review, Mobilenet-CNN model development, project report parts including data pre-processing, part of literature review, and the Mobilenet model architecture and results, slides editing, references

Shruthi Sathish

Literature review, ENet-VIT model development, project report parts including latex-editing, experiment results discussion, future scope, part of literature review, and the ENet-VIT model architecture, object oriented organization of the code.

The team met once every week to discuss the details of the project and MOM was maintained in ClickUp.