Kaggle Project

By: Ryan Bui

Overview of the Project

In this Kaggle challenge I had to use TPU's to speed up the training of deep learning models on images of flowers and classifying them as the correct type from the "Petals to the Metal" Kaggle challenge (https://www.kaggle.com/competitions/tpu-getting-started/overview).

However, since this challenge is more of a demonstration of how to use TPUs for any Kaggle challenge, I decided to conduct some of my own studies.

- The first of which is to see look at 3 different models and compare the time it took to train them.
- The second, is to see the effects of 2 different types of Learning Rate Schedulers and see if the have any impact of the performance of our models.

Performance is going to be determined by the F1 score metric:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

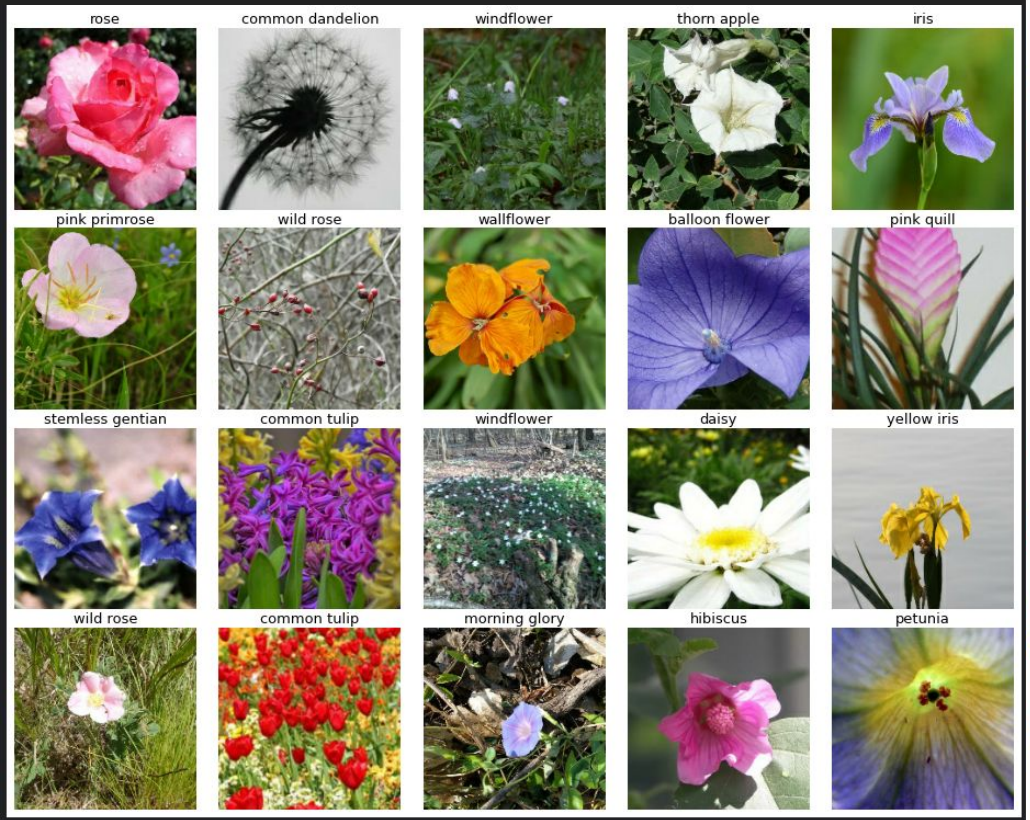
Data

Input: images of flowers ranging from (192x192 pixel jpegs) to (512x512 pixel jpegs) each stored in TFRecord format

Size: The size of the data is 5.15GB

Instances (Train, Test, Validation Split): 2753 training images, 7382 for testing, 3712 for validation

How the Data Looks



Problem Formulation

Input / Output: Our inputs were the images of the flowers and our output is going to be the labeled image of the flowers.

Models used:

- VGG16
- Xception
- ResNet

Types of Learning Rate Schedulers:

- None
- Exponential decay
- Step decay

Table of Results

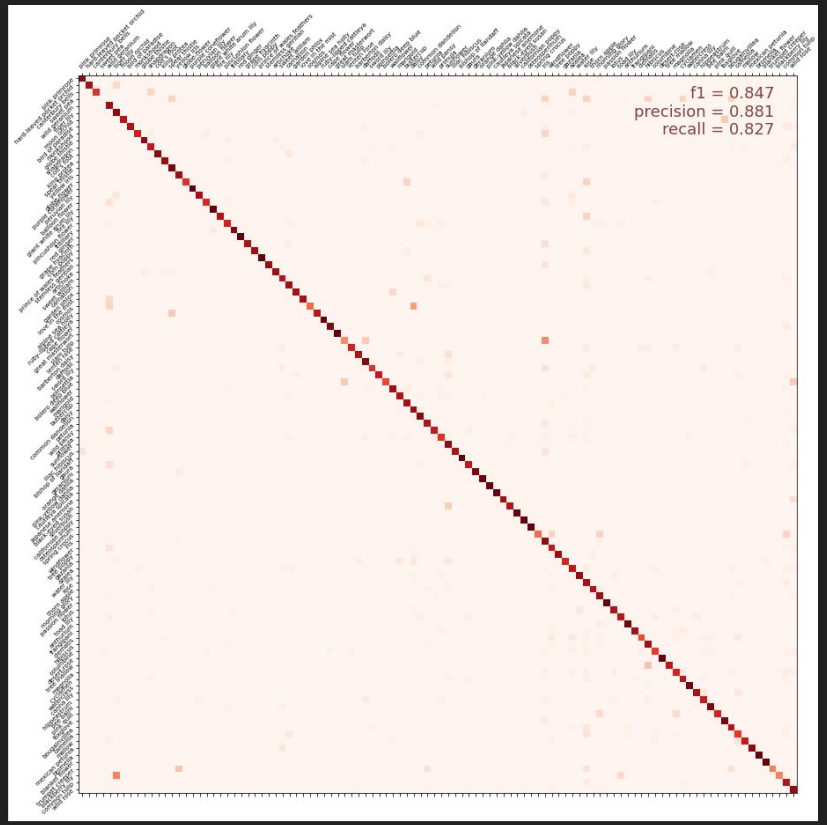
No LRS			
Model	VGG16	Xception	ResNet50
Time to Train	6min 7s	5min 6s	4min 56
F1 Score	0.302	0.834	0.035

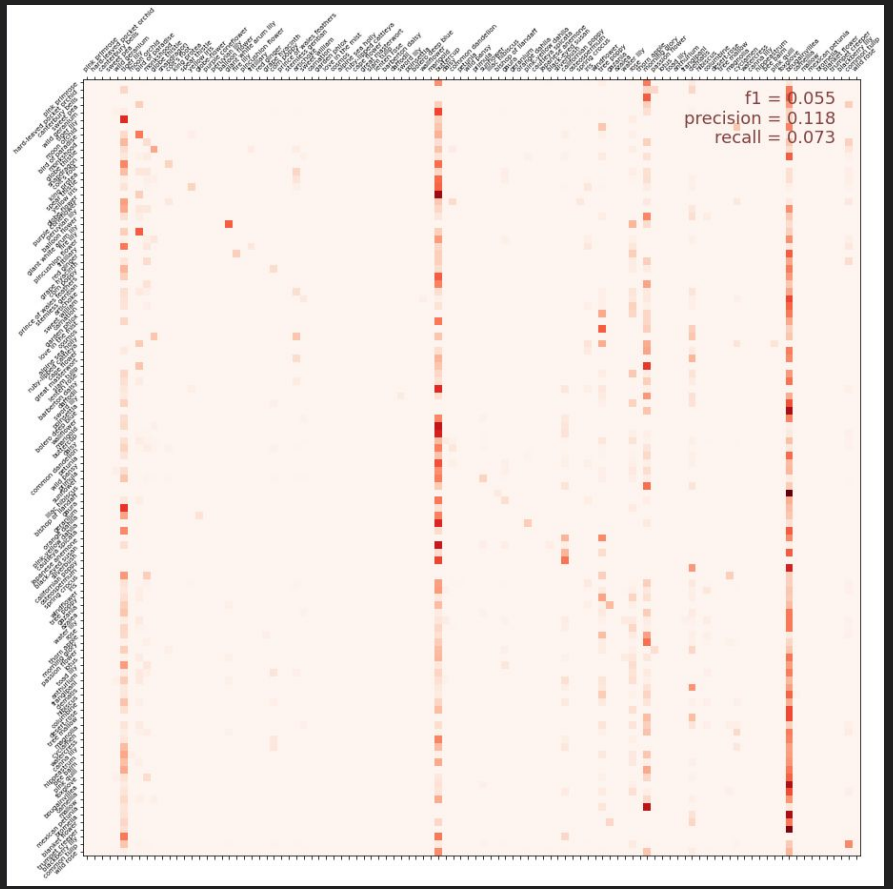
Step Decay LRS			
Model	VGG16	Xception	ResNet50
Time to Train	5min 51s	4min 49s	4min 33s
F1 Score	0.705	0.836	0.055

Exponential Decay LRS			
Model	VGG16	Xception	ResNet50
Time to Train	5min 50s	4min 47s	4min 34s
F1 Score	0.747	0.847	0.246

- We see that Xception had the better scores overall while only being a little bit slower than ResNet50 who had overall low scores
- We can also see a noticeable effect of that having a Learning Rate Scheduler has on the time it takes to train a model.

Visualization of Results





Conclusion

It seems that Xception had the better F1 scores regardless of the different LRSs we used; however we did see that using a step decay LRS significantly improved the training F1 score for the VGG16 model. We also can conclude that while using TPUs the time to train the 3 different models was different with VGG16 being the slowest of the 3 while pretty accurate and ResNet being the fastest however not very accurate for this dataset. We also saw that implementing a LRS did help with the training time for all models.