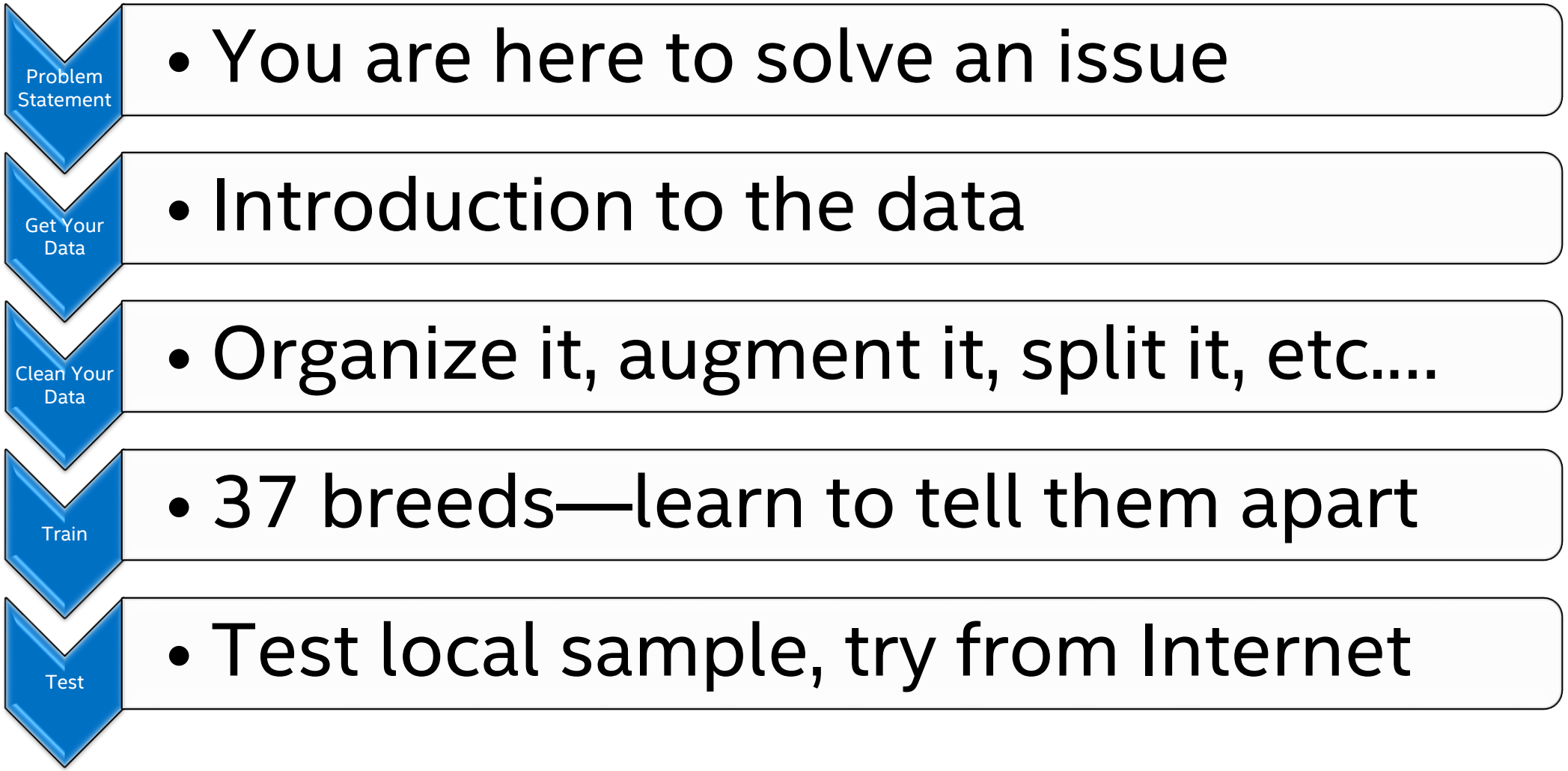
END-TO-END TENSORFLOW MODEL DEVELOPMENT

Outline: What Students Will Learn

- Part 1: Data Wrangling with Breeds on CPU
  - Find a Data set and Fetch Data
  - Clean and Normalize the Data
  - Organize and Optimize Data for Consumption by TensorFlow*
  - After All of This Data Wrangling We Can Actually Begin the Training Process
- Part 2: Training Cat v. Dog with TensorFlow* and GoogLeNet Inception* v1 on CPU
  - Understand the stages of preparing for training using the TensorFlow* framework and an GoogLeNet Inception v1 topology
- Part 3: Evaluate, Freeze and Test Your Training Results
  - Evaluate Your Latest Training Checkpoint
  - Export Your Inference Graph of Inception v1
  - Freeze Your Graph
  - Run a Test Inference on an Image (use as jump off point to OpenVINO™ inference)

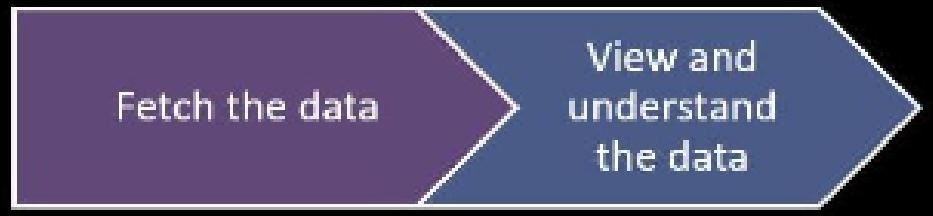


Part 1: fetch the data

[The Oxford Pets Database]

- 37 categories
- ~200 images of each class
- 25 dogs
- 12 cats
- [Paper talks about data and their techniques]

Part 1: view the baseline data



Part 1: clean and normalize the data

- **Extract, Transform and Load (ETL)**
  - **Data cleaning** – Eliminates noise and resolves inconsistencies in the data.
  - **Data integration** – Migrates data from various different sources into one coherent source, such as a data warehouse.
  - **Data transformation** – Standardizes or normalizes any form of data.
  - **Data reduction** – Reduces the size of the data by aggregating it.
- **Prepare data as expected by topology.**
- **Ensure you have enough processing and storage capacity.**

Part 1: organize data for consumption by Tensorflow*

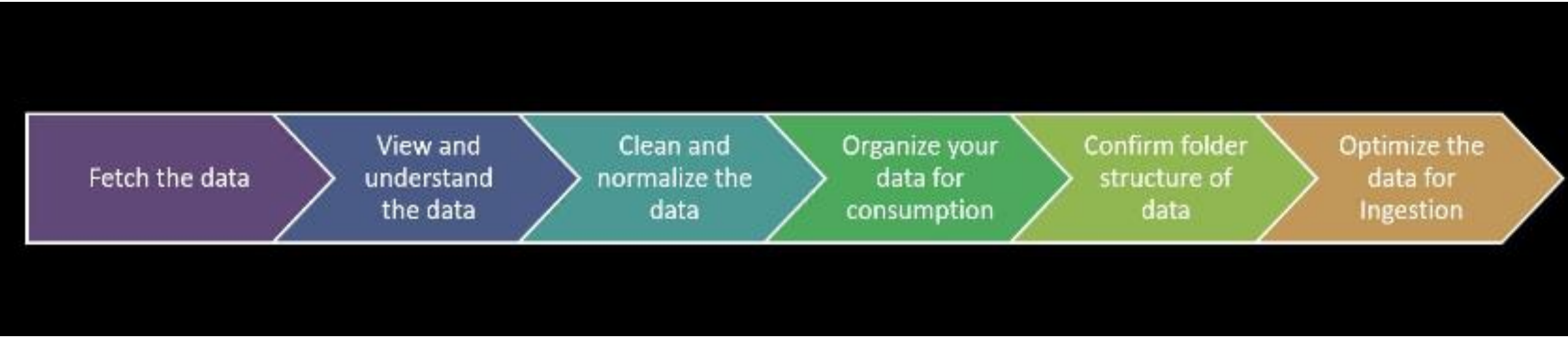
Part 1: Organize data for consumption - categorize

- TensorFlow* expects images to be organized into categories.
- Once complete, each category would look something like this (there are 37 categories).

```
breeds/
    sorted/
        british_shorthair/
            British_Shorthair_184.jpg
            British_Shorthair_269.jpg
            British_Shorthair_37.jpg
            British_Shorthair_71.jpg
            British_Shorthair_167.jpg
        japanese_chin/
            japanese_chin_167.jpg
            japanese_chin_182.jpg
            japanese_chin_191.jpg
            japanese_chin_38.jpg
            japanese_chin_17.jpg
        wheaten_terrier/
            wheaten_terrier_74.jpg
            wheaten_terrier_128.jpg
            wheaten_terrier_137.jpg
            wheaten_terrier_4.jpg
            wheaten_terrier_9.jpg
```

Part 1: Confirm folder structure

Part 1: Optimize data for ingestion



Part 1: Optimize data for ingestion - Create TFRecords

- TFRecord is the TensorFlow* recommended format for ingestion.
- It is a sequence of binary strings.
- If the dataset is too large, we could create multiple shards of the TFRecords to make it more manageable.
- We create two TFRecords, one for training and another for validation.

https://en.wikipedia.org/wiki/Lightning_Memory-Mapped_Database

Part 2: Training

Step 1: Choose the right topology.

Step 2: Set up a pre-trained model to use breeds dataset.

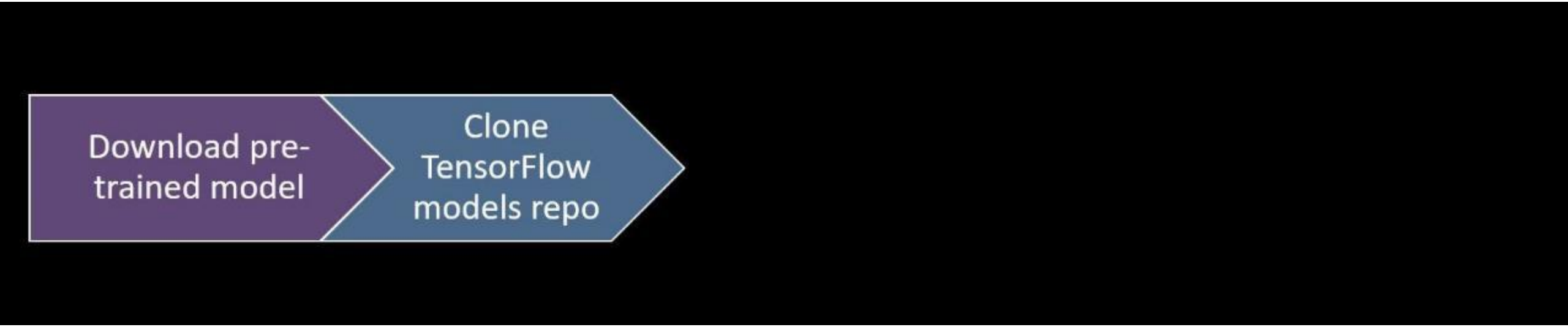
Step 3: Evaluate, freeze, and test results.

Part 2: Step 1 - Select the right topology

- **Criteria:**
  - Time to train: Depends on number of layers and computation required.
  - Size: Keep in mind the edge device you want to deploy to, networks it supports and resources like memory.
  - Inference speed: Tradeoff between accuracy and latency.
  - **GoogLeNet (Inception V1) was our topology of choice**

Part 2: Download pre-trained model

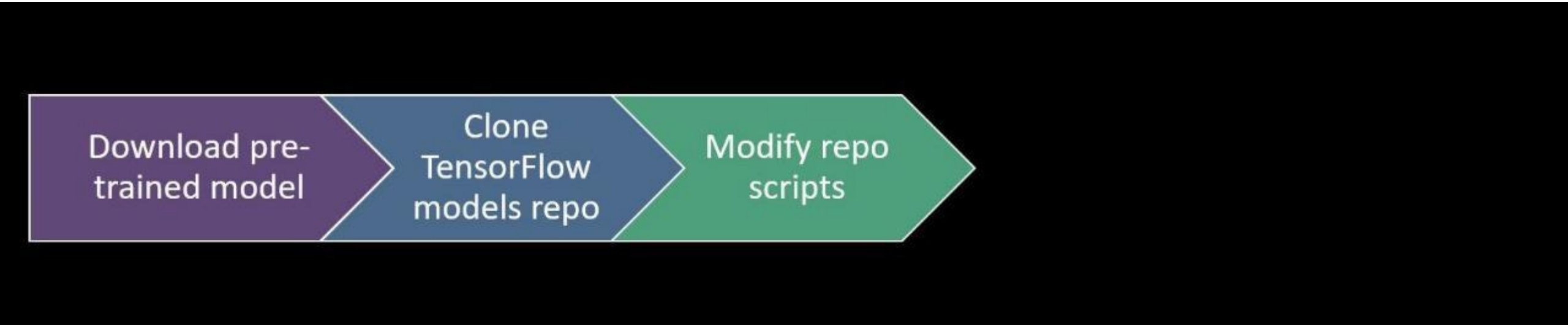
Part 2: Clone Tensorflow*/models github repo



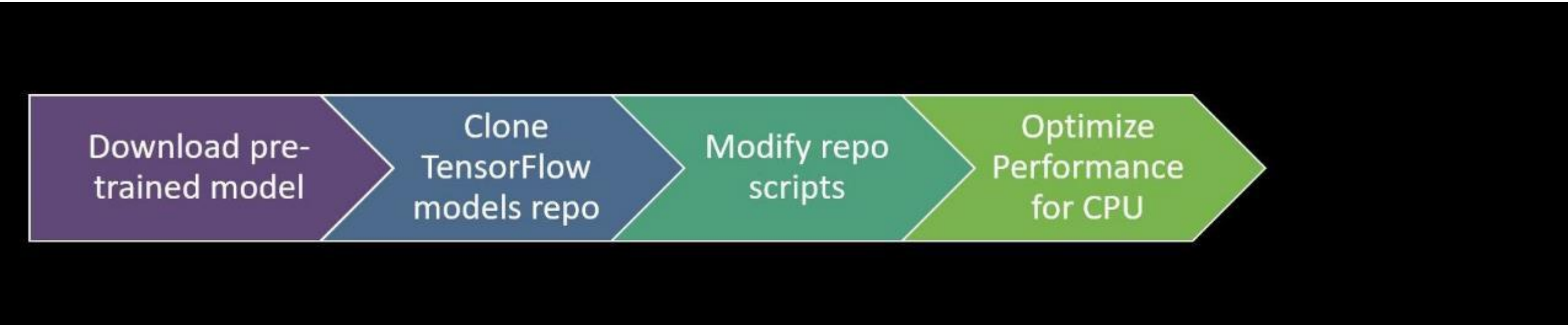
Clone TensorFlow*/models GitHub* repo

We use transfer learning using a Convolutional Neural Network pre-trained on ISLVRC-2012-CLS image classification dataset (https://github.com/tensorflow/models)

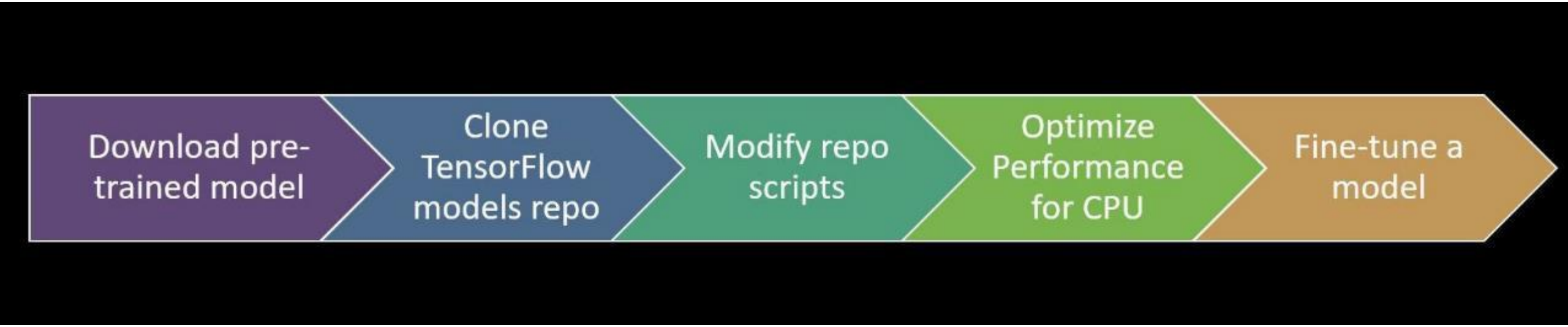
Part 2: Modify/Add files to slim repo to work with breeds dataset



Part 2: Optimize Performance for cpu



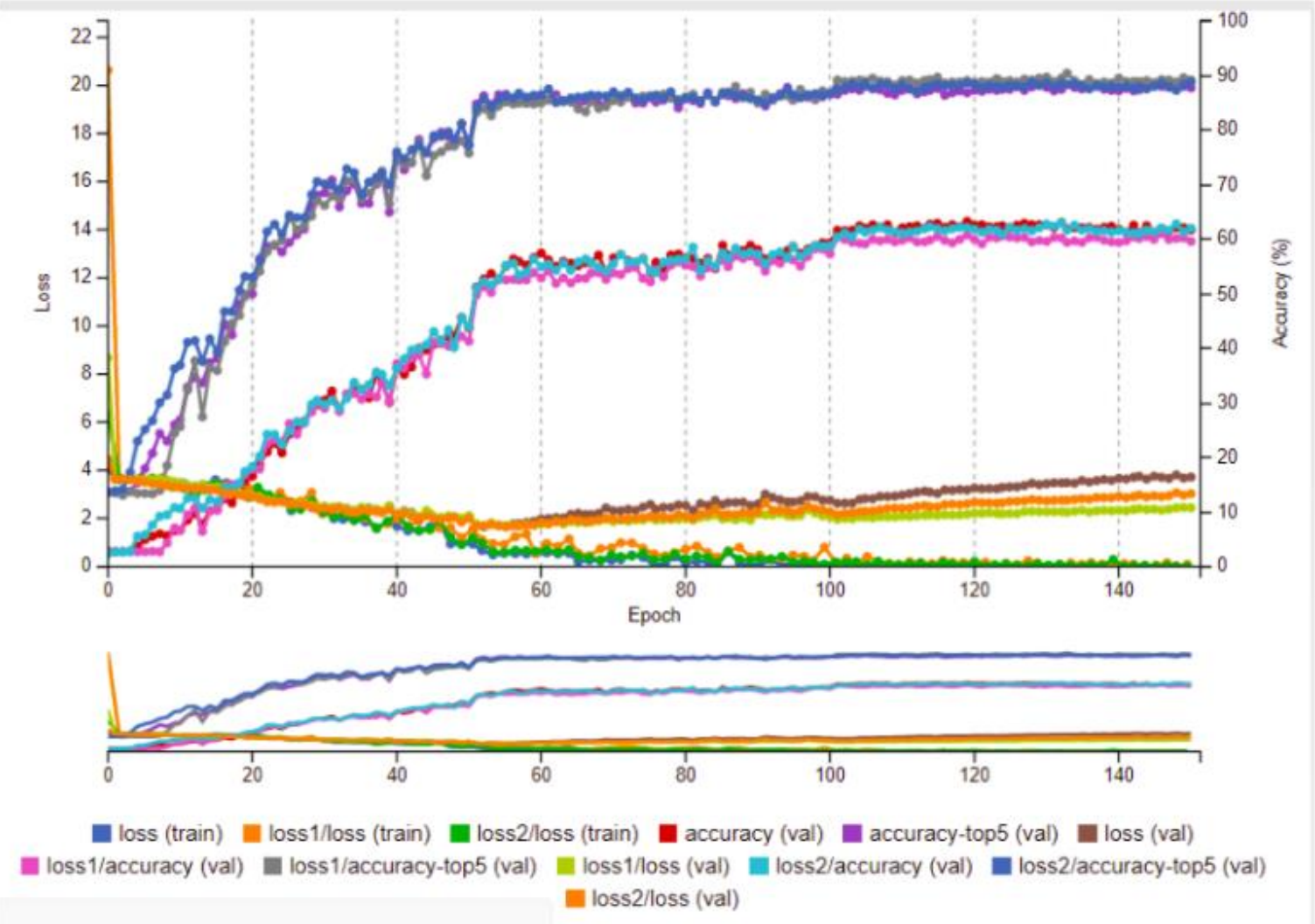
Part 2: Initiate training



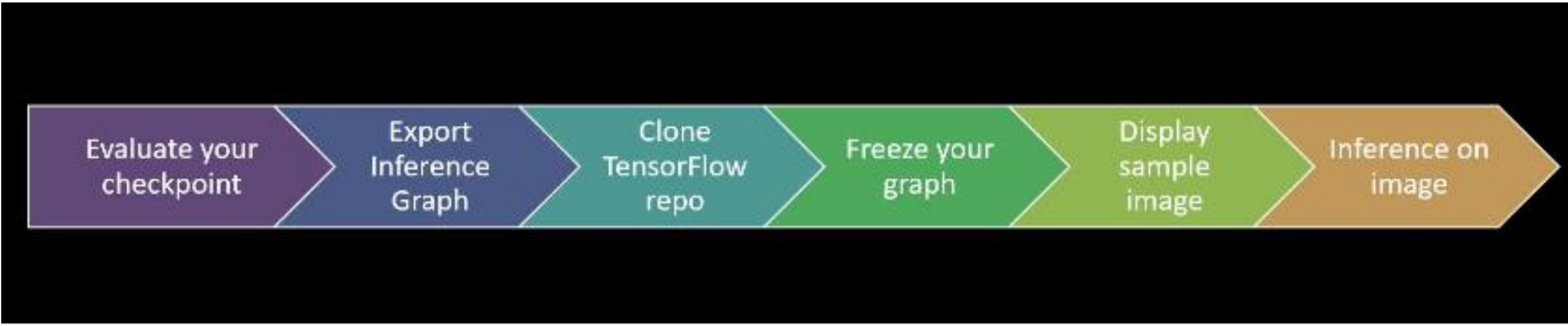
Initiate training and review live training logs:

- When using a pre-trained model on a different dataset, note that the final layer will change to indicate the new set of categories.
- Indicate which subset of layers to retrain while keeping others frozen.
- View results.

Part 2: Results on googlenet inception v1 using breeds



Part 3: evaluate, freeze graph, and test



SUMMARY

Summary

- End-to-End TensorFlow workflow presents in steps how all of the pets imaging data that is to be used in the final custom inception_V1 model is curated – all the way from data aggregation, clean-up and preparation for consumption by TensorFlow.
- The next steps in the workflow is about preparing for training using the TensorFlow framework and an GoogLeNet Inception v1 topology. It initiates training and view a completed graph, and learn about the relationship between accuracy and loss.
- The final steps in the workflow perform a test evaluation of the trained graph and prepares it for exporting out of TensorFlow for eventual runtime usage by either the OpenVINO toolkit or Windows Machine Learning.


intel
experience
what's inside™