Pet (Dog) Detection

Rahil Mehta
Ardalan Ahanchi
Drew Nelson

UNIVERSITY *of* WASHINGTON

Project Overview

- Project aimed to improve performance of pet detection algorithms.
  - Limited our scope to detect dogs in images.
- Performance focus on MLP algorithm
- For MLP to work, need to manipulate images and implement convolution layer as input to MLP

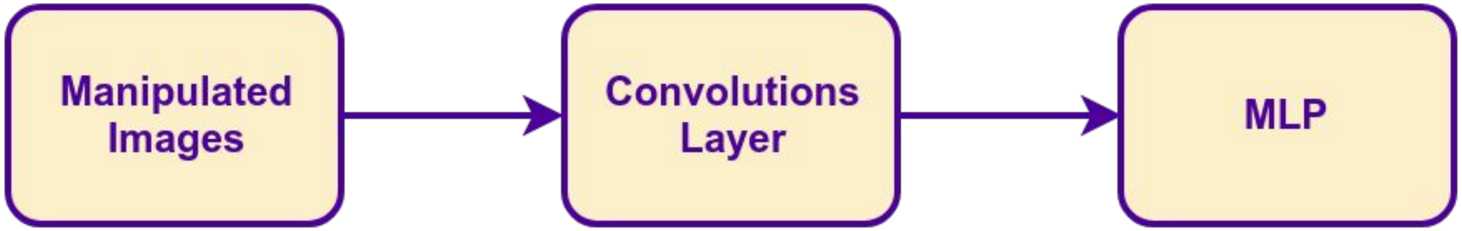


Image Manipulation

- Use the data from OpenImages and convert the data into a format easier for the rest of the application

- Steps
    - Load the image
    - Convert to grayscale
    - Get boundary box of image
    - Resize the image
    - Normalize data

Convolution Layer

- Hope to improve accuracy and performance
- Applying Gaussian filter for now
- Max-pooling reduces dimensions
- Still being integrated into rest of the code

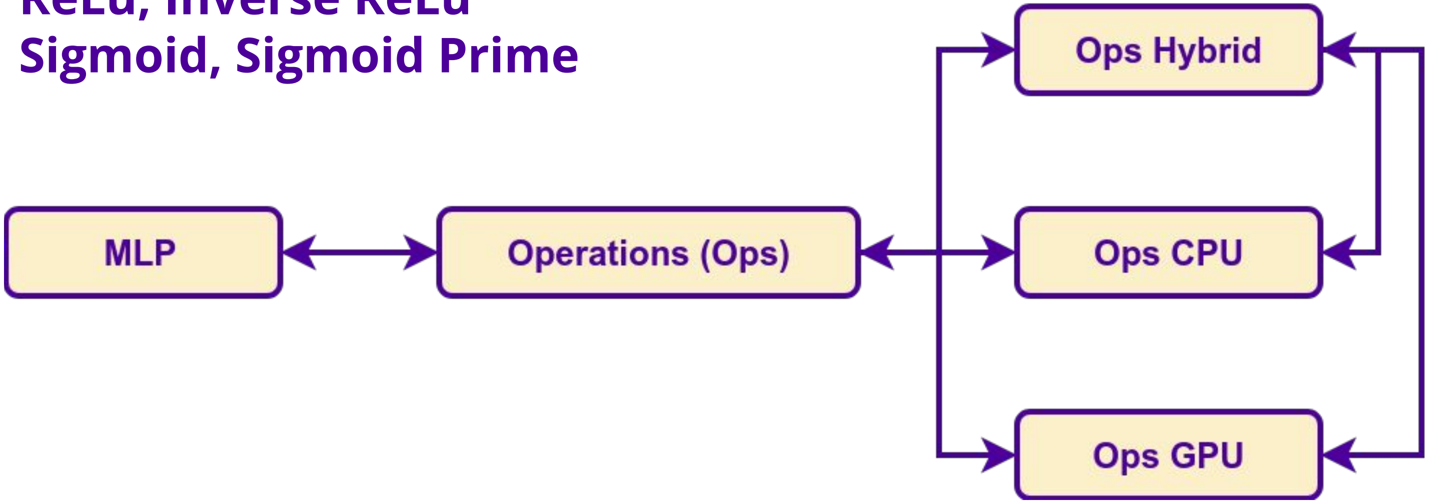
Multi-Layer Perceptron (MLP)

- Trained to find patterns from the input
  - Tested directly on the images
- Uses a custom operations interface
  - Supports CPU, GPU, and Hybrid modes.
- Tested MLP independently
  - XOR function
  - Linear function

Operations

Many operations are used in Deep Learning.

- Add, Subtract
- Multiply, One-by-One Multiply
- Scale
- ReLu, Inverse ReLu
- Sigmoid, Sigmoid Prime



Other Potential Improvements

- More MLP and convolution layers
- Make CPU operations parallel
- Parallelize image manipulation
  - Time to parse images and manipulate images became time consuming. After realizing the time it takes to process, we could have parallelized this operation.

Other Technologies

- CMake
  - Used to help configure on build our source on different platforms.
- OpenImages
  - Primary resource for gathering images.
- OIDv4_Toolkit
  - Python script to get specific image types from OpenImages.
  - Allowed us to specify dog images from OpenImages
  - As well as other miscellaneous images for testing.

Demo

Questions?

Thank you!

UNIVERSITY *of* WASHINGTON

Backup Slides

UNIVERSITY *of* WASHINGTON

Formatting Data