Dog Breed Identification

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH


Paolo Testolina


Mattia Lecci


Aude Razafimbelo

Objectives

First contact with

Keras · python · MACHINE LEARNING · Google Cloud Platform · GitHub · TensorFlow

Objectives

Gain confidence with the available resources

Apply the concepts seen during the classes

Understand and improve already existing models

Challenge

KAGGLE DOG BREED IDENTIFICATION CHALLENGE

AIM: Identify the breed from a photo of a dog

DATABASE: 10.200 train + 10.300 test samples

DIFFICULTY: 120 dog breeds

REASONS:
- Easy Database accessibility
- Well-known problem
- Active community

kaggle

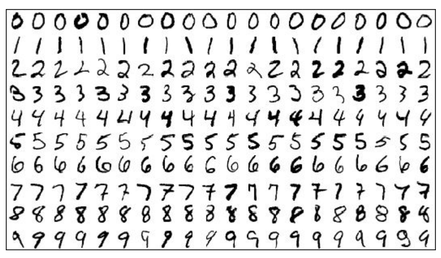
First Approach

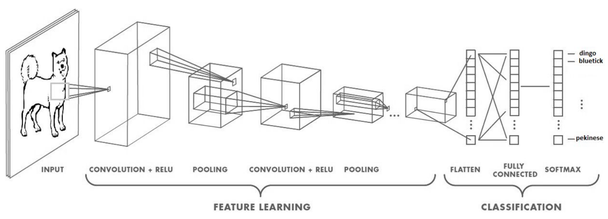
LET'S START FROM SOMETHING WE CAN ACTUALLY UNDERSTAND AND VISUALIZE

LAYERS FROM A **BINARY** CLASSIFIER

3 2D-Convolutional Layers
3 ReLU Functions
3 2D Max Pooling
1 Flattening Layer
2 Dense Layers
1 Dropout
1 SoftMax

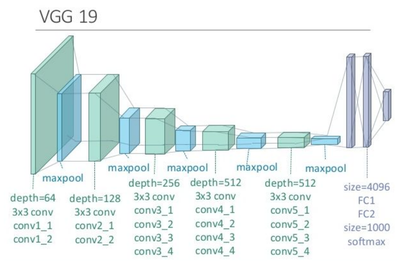
LAYERS FROM A **MULTICLASS** CLASSIFIER





Transfer Learning

VGG19



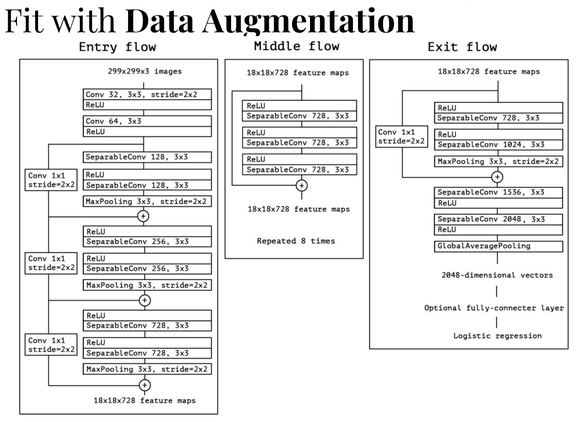
Several **optimization rounds**:

- **First** Run:
  - Dropout Rate
  - Optimizer choice
  - Batch Size
- **Extra** Run (SGD's parameters optimization):
  - Learning Rate
  - Momentum
  - Decay
  - Nesterov
- **Extra** Run (Adam's parameters optimization):
  - Learning Rate
  - Beta_1
  - Beta_2
  - Decay
- **Second** Run:
  - Dropout Rate
  - Number of Final Dense Layers
  - Final Base-Model's layers to unfreeze
  - Batch Size
- **Third** Run:
  - Kernel Regularization Parameter
  - Bottleneck Layer

VGG16

- Use **pre-trained** models
- **Omit** last layer
- **Freeze** the original model's weights
- Fit with **Data Augmentation**

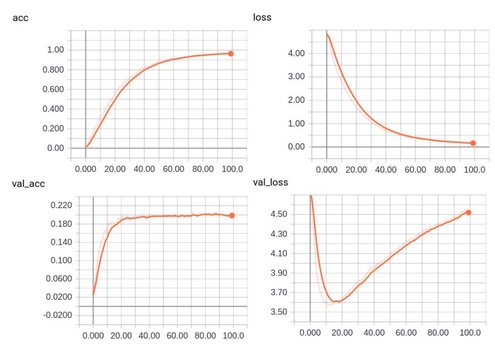
Xception



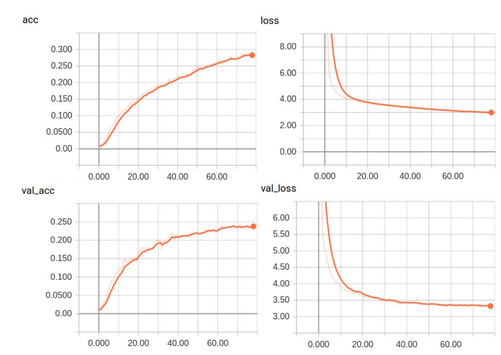
Observations:

- **Overfitting** was a big issue for **VGG19** (see Results section)
- Heavy **regularization** techniques to reduce it
- **VGG16** doesn't overfit as much and performs better
- **Xception** yields unexceptional results

Results

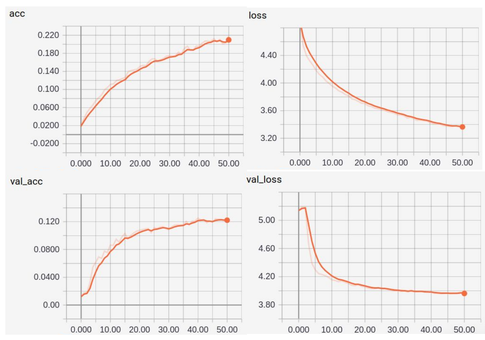


VGG19



VGG16

MNIST



What did we learn?

Transversal Knowledge/Skills:

- Setting up and accessing remote **virtual machines**
- Accessing and using basic Google Cloud instances
- Increased confidence with linux's terminal
- First experience with **Git**/Github
- First steps with a new programming language: **Python**
- First experience with **Jupyter Notebooks**
- Studying **research papers** and obtain implementable code from them

Specific Knowledge/Skills:

- Creating **Neural Networks** from scratch using Keras
- Manipulating existing architectures
- Reading results (distinguishing between training and validation)
- Methods to avoid **overfitting**
- Hyperparameters **optimization**
- Operating with **callbacks**
- Operating with **Tensorboard**

Model and source code: https://github.com/telecombcn-dl/2017-dlai-team4