MAI

Deep Learning

Guided lab

Transfer Learning

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Dario Garcia Gasulla
dario.garcia@bsc.es

Set-up

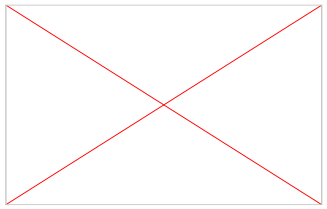
- Upload the code to your account
  - https://github.com/UPC-MAI-DL/UPC-MAI-DL.github.io/tree/master/_codes/3.Embeddings
- Copy the pre-trained models
  - You can run the command locally, and upload the files from your .keras/models folder to your home directory in GPFS
- Copy target datasets
  - /gpfs/projects/nct00/nct00001/mit67
  - Used in:
    - fne_main.py
    - fine_tunning.py (L38-39)

HIGH PERFORMANCE ARTIFICIAL INTELLIGENCE

Summary

Sample codes for:

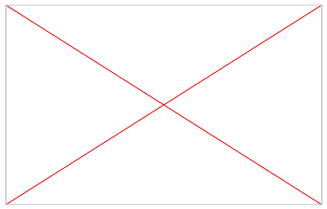
1. **Fine-tuning:** Use a pre-trained network and re-train it for a different task
2. **Feature-extraction:** Use a pre-trained network as feature descriptor for a different task



Disclaimer

Sample codes:

1. **Kind of work**
2. **May have bugs**
3. **Are inefficient** (particularly feature extraction)
4. **Will not work out-of-the-box. Need to upload the models and datasets**



Don't try to fix or extend the code. Copy something if it's useful and make your own code.

Fine-tuning

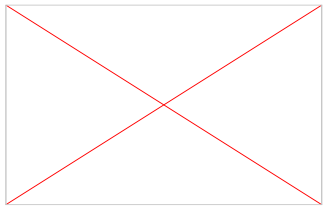
- Training from scratch is often a bad idea [1,2]
- Many factors affect transferability
  - Similarity between tasks
  - Size and variance of source task
  - Size and variance in target task
  - Layers transferred, locked and re-trained
- Play with:
  - Sources. VGG16 on ImageNet/Places is easy to find
  - Targets
  - Transferred layers
  - Frozen/retrained layers

[1] **How transferable are features in deep neural networks?** Yosinski et. Al.

[2] **Factors of Transferability for a Generic ConvNet Representation** Azizpour et. al.

Fine-tuning

Let's take a look at the code



Fine-tuning

- Command

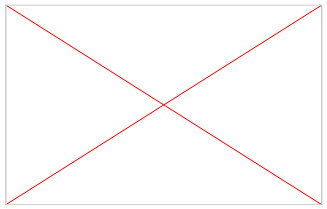
python fine_tuning.py

- Hyper-parameters can be tuned within the code

HIGH PERFORMANCE ARTIFICIAL INTELLIGENCE

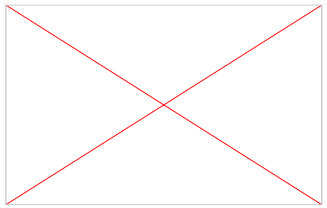
Feature Extraction

- Code sample for
  - Extract neural activations for images as processed by a pre-trained network
  - Apply a postprocessing to these activations
  - Train a SVM with the resulting vector representations
  - Check classification performance
- To play:
  - Sources & Targets (same as fine-tuning)
  - Post-processing (FNE implemented)
  - Extracted layers



Feature Extraction

Let's take a look at the code

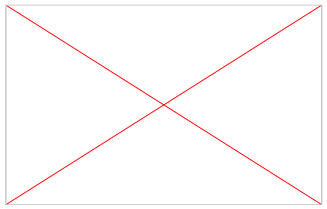


Feature Extraction

- Command

python fne_main.py

- Layers extracted can be tuned within the code
- Using both fc's increases the cost significantly



Embedding Spaces

- Let's play around in a word embedding space
- Command
  - python word_embeddings.py

1. Find pairwise distances
2. Find the most similar word
3. Compute analogy (aka regularities) and find closest result
4. ToDo: Clustering on the embedding space