

Deep Learning - MAI

Guided lab - Transfer Learning

Dario Garcia Gasulla
dario.garcia@bsc.es

Goal

- Experiment with transfer learning methods
- In the guided lab:
 - Model pre-trained in Imagenet
 - Try to solve the MIT67 indoor classification task

Set Up #1

- ❖ Upload the code to your account

https://github.com/UPC-MAI-DL/UPC-MAI-DL.github.io/tree/master/_codes/3.Embeddings

- ❖ Upload pre-trained models (~/.keras/models)

You can run the command locally, and upload the files from your `.keras/models` folder to your home directory in GPFS

A couple available here: `/gpfs/projects/nct00/nct00038/ (VGG16 w/o top)`

Set Up #2

- Link target dataset

`/gpfs/projects/nct00/nct00038/mit67`

- Used in:
 - `fne_main.py`
 - `fine_tunning.py` (L38-39)

Sample codes

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

Disclaimer

- ❖ Sample codes:
 - Kind of work
 - May have bugs
 - Are inefficient (particularly feature extraction)
 - Will not work out-of-the-box: Upload pre-train models and datasets
- ❖ Don't try to fix or extend the code. Copy something if it's useful and make your own code

Let's look inside

Fine-tuning

- ❖ Training from scratch is often a bad idea. Factors of transferability:
 - Similarity between tasks
 - Size and variance of source task / target task
 - Layers transferred, locked and re-trained
- ❖ Play with:
 - Sources. VGG16 on ImageNet/Places is easy to find
 - Target tasks
 - Randomized/fine-tuned/frozen layers

Fine-tuning

❖ Code

https://github.com/UPC-MAI-DL/UPC-MAI-DL.github.io/blob/master/_codes/3.Embeddings/fine_tuning.py

- Keep fc layers or not (L46)
- To freeze or not to freeze (L49)
- Adding rand init layers (L55)

❖ To speed things up during the guided lab

- Freeze lots of layers
- Use only a subset of the train set

Feature Extraction

- ❖ Code sample for
 - Extract neural activations for images as processed by a pre-trained network
 - Apply a post-processing 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

❖ Code

https://github.com/UPC-MAI-DL/UPC-MAI-DL.github.io/blob/master/_codes/3.Embeddings/fne_main.py

- Create output variable (L48)
- Define layers to capture (L55)
- Store activations of current batch (L80)
- Postprocessing (L81, L87, L91)

https://github.com/UPC-MAI-DL/UPC-MAI-DL.github.io/blob/master/_codes/3.Embeddings/fne.py

- Load full pre-trained model (L16)
- Define layers to extract (L22)
- Reduce problem size (L30), train & test SVM (L63)

Dario Garcia-Gasulla (BSC)
dario.garcia@bsc.es

