1️⃣ Getting started

Learning steps

- ☑️ ~~YT video~~
- ☑️ ~~Notebook~~
- ☑️ ~~Book chapters~~

Resources

- Website 💻
  - Lesson 1
- Notebooks 🗒️
  - Kaggle notebook
- Book 📘
  - Book chapter
  - Solutions to exercises

Quick notes

YT video

- Image classifying has gone from a impossibility (and even a joke) in 2015 to a 2min easy task now
- Output generation based on text promts inputs
  - Image generation using Dalle-2 or comparable models (Midjourney, ...)
  - Text generation using Pathways Language Model (PaLM), Google
- Check out Radek's book (link in website lesson 1)
- Top-down approach like in sports: Start with a full professional game and then get better yourself from there
- Jeremy Howard's qualifications

    - Wrote a very popular book that is used for this course ⇒ Read the book because learning the same things with different approaches works best
        - Deep Learning for Coders with fastai & PyTorch
    - Worked in ML for 30 years and built multiple companies, won Kaggle competitions
    - Worked on the basis of the NLP revolution that we see today
        - Universal language model fine-tuning [...]
    - Teaches courses
- Why can we now create such a bird classifier, and not before?
    - A neural network (NN) discovers features, we don't give it to them ⇒ we can try to understand what the NN uses to make classification
    - We use layers to find more and more advanced features
        - More layers = more complex images classifiers
    - The NN uses images and learns itself from them
    - A NN can be used to classify sounds as well using waveforms
        - 💡With some creativity, you can use an image classifier to classify almost anything using vectors, waveforms, etc
- Myths and requirements for the course
    - Math level for the course is not high and specific to some subjects
    - Can use low amount of data
    - Don't need expensive computer
- PyTorch 🆚 TensorFlow in 2022
    - TensorFlow has been dying in recent years
    - On the contrary for PyTorch which is becoming the majority of usage for research papers and repo's
    - PyTorch requires more code
        - ➡️ FastAI is a library that has been built on top of PyTorch to make it easier to use (less code)
- Use of Jupyter Notebooks to run code in the course
    - You can run it on a cloud server
    - **Kaggle**, Colab, Sagemaker, etc
- About the notebook for this session
    - Libraries from FastAI often start with fast[...]
    - FastAI has many useful features to find input:
        - `download_url`
        - `resize_images`
        - `search_images`

- DataBlock API
  - Will create the right kind of model for your usage (most of the time)
  - Focus on practicality during this course
  - Use a lot of functional style programming
  - Parameters for DataBlock
    - `blocks`: inputs' and outputs' types
    - `get_items`: get inputs for training the model
    - `splitter`: cross-validation parameter
    - `get_y`: how do we know the correct label for a photo?
    - `item_tfms`: standardize images
  - Dataloaders are the things that PyTorch iterates through to get multiple images at a time (using GPUs)
    - Basically parallelizes the training using (mini)batches
  - 🔥 To find the parameters for your model, use the FastAI docs
    - Start from something similar to see how it is done
    - Use API docs as well
- We train the model with `vision_learner` for example
  - *timm*: deep-learning lib used to classify images
  - `fine_tune`: used for fine-tuning the model automatically using best practises
    - The weights are usually already created for you, they are just adjusted to your needs
- What about kinds of DL uses other than image recognition?
  - **Image segmentation**: Recognize multiple parts of an image
    - Classify every pixel of an image (what is it a pixel of?)
    - `SegmentationLoaders`: Similar to Datablocks for im seg
  - **Tabular analysis**: Make predictions from spreadsheets
    - Here you fit your model instead of fine-tuning it, because the models are really specific to the datasets that are used
  - **Collaborative filtering - recommandation system**
    - Use a dataset of what users used/liked, and guess other related things that they might like/find useful
      - Relation between users that like the same kind of things
    - Creation of a learner here as well
      - You pass it the training data
- Use of DL in various domains
  - If something can be done relatively quickly by a human, it can probably be done by DL

        - *envision*
    - If it requires a lot of reflection for a human, probably not...
- Classic software 🆚 ML
    - CS: `inputs → program → results`
    - ML: `inputs + weights` → `model` → `results` → compute the `loss` → update the weights
        - The `loss` and `update` steps are crucial to compute the weights we need
        - When the weights are known we can "integrate" them into the model ⇒ A trained model makes it look like this: `inputs → model → results`