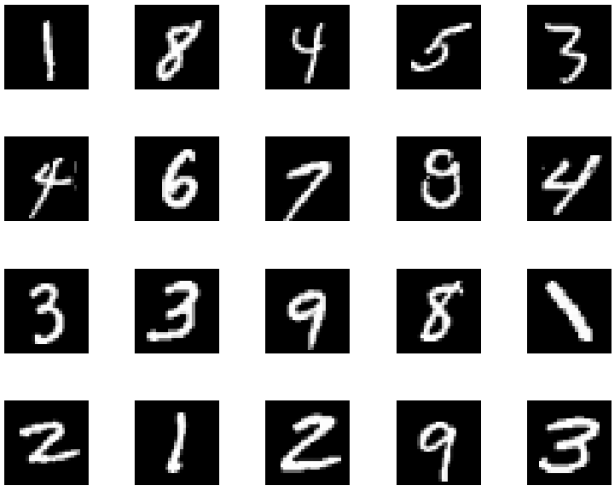
1ST GRADED ASSIGNMENT: DESIGNING AND TUNING A DENSE NETWORK

MNIST

Data:

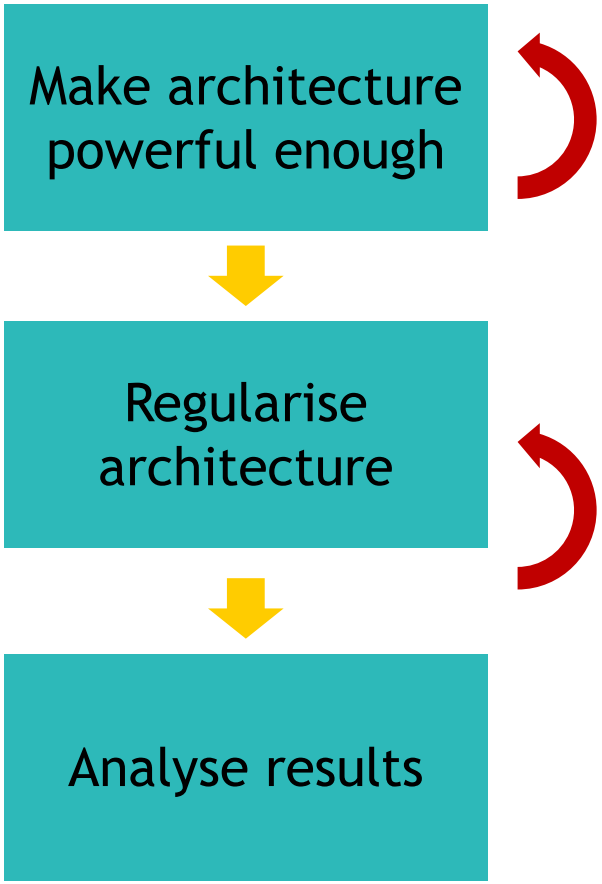
- low-resolution gray-scale images (28x28) of hand-written digits
- dataset is nicely balanced
- test-set is “standard”



Task:

- Classify these images (10 classes) as well as you can
- Quality metric: ACCURACY (can reach well above 0.98)
- Using a network with **dense layers only** (28x28=784 features)
- **Tune learning convergence:** batch size, learning rate, epochs (set epochs high enough and use early stopping)
- **Tune regularisation:** L1, L2, dropout (no others yet!)
- Experience the impact of these hyperparameters
- Analyse network performance and learn to make the right conclusions on what to improve

What to do?



Then try again for different type of architecture, e.g. ‘wide’ versus ‘tall’

- Architecture - only Dense():
  - how many layers ?
  - how many neurons per layer (use **ReLu**)?
  - (don’t be afraid to go back, look for examples)
- **Learning convergence (“inner loop”):**
  - batch size, number of EPOCHs
  - Initial learning rate (sticking to Adam is fine for this GA)
- Regularisation hyperparameters:
  - L1/L2/maxnorm parameters
  - stopping criterion (early stopping: when?)
  - dropout in which layers? Which dropout rate (can differ per layer)?
- Keep a logbook of everything you try:
  - architecture, with all parameters
  - trained model filename (in case you need it later)
  - convergence plots + **numbers!!!** (train/val, loss/accuracy)
  - what you conclude from these

You get 2 “getting started” notebooks

- Having 2 notebooks allows you to uncouple **network training** and **network analysis** in separate windows – this way you can update your logbook while another network trains
- Make copies of training notebook for each type of architecture you work on (File -> save a copy on Drive) or to work in parallel on different architectures
- Example code shows:
  - Saving to and loading from Google Drive
  - Data loading
  - Train – validate split (use same split every time: random seed or save split to share among team members)
  - Simple, unregularised architecture (WAY too simple)
  - Possible starting code for analysis and visualisation (feel free to add your own functions)

What to to?

GHENT UNIVERSITY

- Split up original train set into train&validate sets (SAME for all team members)
- Explore & optimise model hyperpameters and design (try to improve your understanding!!)
  - document optimisation procedure according to steps from previous slide in your group logbook (shared document, e.g., google doc or Ms OneNote file) – **will be the basis for your report**
  - for each step: save trained models, store hyperparameters (or model generation and training code) and trained model file names + train & validation scores (cross entropy error + accuracy) and learning convergence plots (screenshots)
- Select final best model(s) in your team based on validation performance (can be more than 1 model if they are sufficiently different, e.g. “deep” versus “wide”)
- Retrain best model(s) on entire original train set
- Analyse performance and errors of selected model(s) on test set

What to hand in? -> report + code

GHENT UNIVERSITY

- Report = single, coherent pdf file
- Clear description of your path towards your best model (or models)
- Use material from logbook to illustrate what you tried, which option(s) was/were best in each stage, based on plots and numbers
- No extensive text - omit attempts that led nowhere or for which you know in hindsight they were wrong choices
- For final model(s) only: brief discussion of performance and errors made by the model
- The aim of the “report” is to convince us that you have mastered the steps in the previous slide, that you understand the effect of the hyperparameters used in this assignment and can interpret curves and scores correctly

What to hand in? -> report + code

GHENT UNIVERSITY

- **Adapted training notebook (s)** that only reproduces your final model(s), trained on whole training set and evaluated on test set
- Important: need to fix number of epochs now (based on early stopping point during validation) since you do not have a validation set any more!
- DO NOT use test set for early stopping!!
- Analysis notebooks **for final model(s)**
- All handed in notebooks must be clean (all non-requested code blocks removed), self-contained, and fully executed:
  - Network training executed in the notebook
  - All plots and scores visible
  - No error messages!
  - (given the large number of groups, we will only re-execute them when there is doubt)

How to hand in

GHENT UNIVERSITY

- Make a single ZIP file that contains all files
- Submit on **Ufora** before deadline (Friday, Feb 26th, 23:30)
- Deadline is a **hard** deadline – don't wait till the last minute!
- Can submit multiple times but only last submission will be stored – make sure each submission is complete and coordinate with your team-mate(s)

Evaluation / grading

- Most of the points on **methodology and understanding**:
  make sure all decisions are well motivated and documented!!
- Impact of final model quality:
  - small amount of minus points if final model test accuracy at or below 0.98 (probably, something is wrong with your understanding too, in that case)
  - small amount of bonus points for final model test scores that are considerably higher (don't spend too much time on getting high scores!)

- Specific subparts that are graded:
  - quality of unregularised model (power and learning convergence),
  - regularisation steps and methodology,
  - analysis/discussion of final model result and errors,
  - general understanding of hyperparameters,
  - clarity and completeness of report and code

Study effort and peer assessment

- After each assigment, you will be asked to fill in a very brief Google form where you are asked:
  - whether you team mate(s) made a fair contribution to the assignment
  - whether you feel you made a fair contribution to the assignment
  - how much effort (in hours) you put into studying the material and making the assignment
  - what your background is (previous coding experience, machine learning experience) – only the first time!

- So ideally: try to keep track of the time you invest
- **Filling in these polls is mandatory!**