Skills required for research internship or thesis on computational modeling / neural networks / deep learning

Computational Cognitive Neuroscience Lab
University of Padova

Mandatory readings

The following papers provide a general overview of our modeling framework, which is mostly focused on unsupervised deep learning. Further references can be found in the bibliography section of each paper.

- Hinton, McClelland, & Rumelhart (1986) "Distributed representations". *PDP book, Chapter 3*
- Hinton (2007) "Learning multiple layers of representation". *Trends in Cognitive Sciences*
- McClelland (2009) "The place of modeling in cognitive science". *Topics in Cognitive Science*
- Stoianov & Zorzi (2012) "Emergence of a visual number sense in hierarchical generative models". *Nature Neuroscience*
- Zorzi, Testolin & Stoianov (2013) "Modeling language and cognition with deep unsupervised learning: a tutorial overview". *Frontiers in Psychology*
- Testolin, Stoianov & Zorzi (2017) "Letter perception emerges from unsupervised deep learning and recycling of natural image features". *Nature Human Behaviour*

Theoretical concepts

The following list represents a non-exhaustive set of general topics that should be mastered **before** starting the research internship:

- Linear *vs.* non-linear systems
- Vector spaces, matrices, hyperplanes
- Matrix operations: scalar product, dot product, transpose, inverse
- Matrix definition of a linear system
- What are derivatives and gradients, local and global maxima and minima of a function
- What is an attractor
- Basic knowledge of probability distributions: Gaussian distribution, mean, standard deviation
- Logistic function

Besides these general concepts, the student should be familiar with the topics covered in the Artificial Intelligence course:

- Main ideas of Connectionism
- Machine learning frameworks: supervised, unsupervised, reinforcement
- Basic formalism of artificial neural networks: artificial neuron, activation functions, network architectures, bottom-up and top-down processing
- Generalization and overfitting, cross-validation, learning regularization
- Supervised learning: linear associator, multi-layer neural feed-forward networks, error backpropagation
- Unsupervised learning: Hebb rule, dimensionality reduction, feature extraction, clustering
- Generative models: restricted Boltzmann machines, contrastive divergence, hierarchical generative models (unsupervised deep learning)

- How to analyze internal representations of a neural network: what is a receptive field, how to use a linear classifier for reading out the representation content
- Computational modeling in psychological research: methods and paradigms

Required programming abilities

- Knowledge of scripting languages (MATLAB, Python)
- What is a data structure and how to use dictionaries with MATLAB or Python
- How to use control structures: selection, iteration, nested loops
- Types of variables: integer, float, double, char, boolean
- Matrix and vector indexing
- Arithmetic operations on vectors and matrices
- Loading/saving data from/into external files
- Functions, how to call an external function, how to pass arguments to a function
- Print to video content of variables, graphic plots (lines, histograms…)
- Debugging and real-time control of program execution

Useful MATLAB commands (detailed explanation available from MATLAB documentation):

- clear
- data structures (e.g.: dictionary.field = value)
- vector indexing for selecting elements, rows and columns
- built-in functions: zeros(); ones(); max(); min(); mean(); std(); rand(); randn(); find(); fprintf();
- matrix transposition: '
- logical operations: == ~=
- graphical functions: plot(); figure(); axes(); imagesc(); colormap(); hist();
- input and output to file: load(); save();
- change matrix dimensions: horzcat(); vertcat(); reshape();
- sending to and collecting data from Graphic Processors: gpuArray(); gather();

Suggested on-line free courses (basic)

Introduction to programming with MATLAB
https://www.coursera.org/learn/matlab

Introduzione a MATLAB
https://learn.eduopen.org/eduopen/course_details.php?courseid=130

Introduzione alla programmazione con Python
https://learn.eduopen.org/eduopen/course_details.php?courseid=194

Suggested on-line free courses (advanced)

Neural networks for machine learning
https://www.coursera.org/learn/neural-networks

Machine learning
https://www.coursera.org/learn/machine-learning

Test your skills!

The following exercises should be used to test your level of competence with the concepts listed above. You should be able (with some help of Google) to carry out all the tasks and write a detailed report with the simulation and analysis results.

- Follow the instructions given in the Appendix of Testolin et al. (2013) to create an unsupervised deep learning model of handwritten digit recognition. You will learn how to download the MNIST dataset, extract the patterns and save them into a convenient MATLAB format, create a training set using mini-batches, and finally training a deep belief network using the provided source code[1].
- NOTE: If you have a CUDA-enabled GPU on your PC, it would be convenient to use the GPU code rather than the CPU code. In order to use the GPU code, make sure you successfully installed the CUDA drivers and you enabled the Parallel Computing Toolbox in MATLAB. If you are using the Python code, make sure you correctly installed the PyCUDA libraries.
- Plot some examples of training images using the "imagesc" MATLAB command. Save the images into a JPEG file in your PC using the "imwrite" command.
- Once you successfully trained the unsupervised deep learning model, analyze the network by plotting the receptive fields of the neurons in the different layers and by reading-out the internal representations using a simple linear classifier. The complete source code for these analyses is also available from our website[2].
- NOTE: In order to train the read-out classifier on the internal (hidden) layers, you will first need to reshape the 3D matrix containing the training patterns into a 2D matrix using the "permute" and "reshape" MATLAB commands. You will also need to compute the hidden activations of each layer in the network by using a sigmoid activation function.
- Try changing some parameters of the unsupervised deep learning model to see if this produces important differences in the internal representations developed by the network. For example, change the network architecture by adding or removing hidden layers or by changing the number of hidden units.
- Try changing some learning hyperparameters (learning rate, weight decay, sparsity constraints). How this affects the shape of receptive fields? How this affects the read-out performance?
- Add some background noise to the test MNIST images to see how this affects the recognition accuracy. You can add an increasing amount of Gaussian noise by using the rand() MATLAB command and changing the standard deviation of the random variable that is added to the image. By plotting the recognition accuracy at different noise levels, you should be able to produce psychometric curve similar to that reported in Fig. 2e of Testolin, Stoianov and Zorzi (2017). Check also the Method section of that paper for further details about the noise simulations.
- Create a confusion matrix to analyze the type of classification errors made by the deep network. Are there classes that are more frequently confused? Why? (NB: the confusion matrix should be created using noisy test images – with a noise level guaranteeing at least a classification accuracy of 60% – in order to force the model to produce errors).

Complementary/alternative to the use of our MATLAB/Python code would be to implement these simulations using an advanced deep learning framework, such as TensorFlow or PyTorch. **Familiarity with these frameworks is considered as a strong plus.**

[1] http://ccnl.psy.unipd.it/research/deeplearning
[2] http://ccnl.psy.unipd.it/research/dbn-analyses