Introduction to Nengo

Iwo Błądek

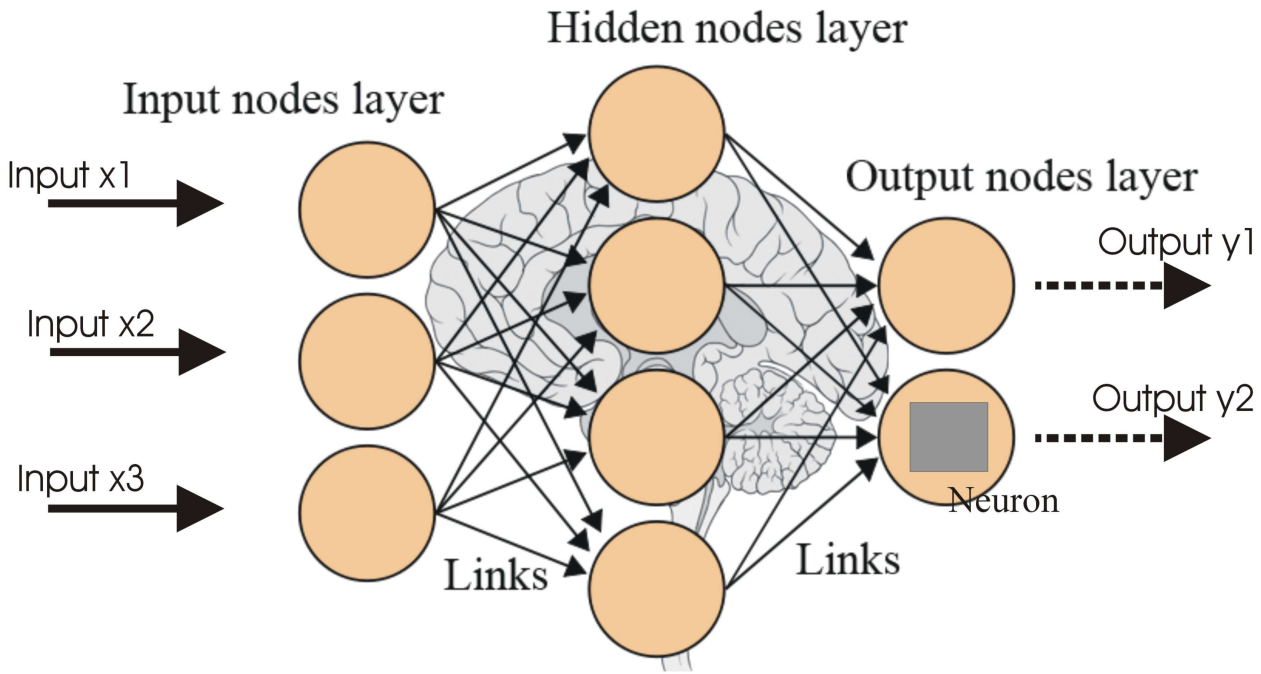
Poznan University of Technology

Poznań, 2018

Connectionism

Connectionism

- Realizing *complex* activities by use of many *simple* processing units (e.g. neurons).



Source: http://futurehumanevolution.com/artificial-intelligence-future-human-evolution/artificial-neural-networks

Localist representations

- **Idea: every concept is represented by a dedicated neuron.**
- It is an intermediate step between fully symbolic and fully distributed systems.
- **Pros:** easy to implement, learn and understand.
- **Cons:** very inefficient when the data has compositional structure. Also: brain most likely doesn't work like that.

Distributed representations

- Idea: **every concept is represented by different activities of the same set of neurons.**
- Traditionally understood symbols and inferences made on them are obscured in the interactions between processing units.
- Distributed connectionist approaches are closer to the real brain on the implementation level than symbolic approaches, which focus on the high level description.
- **Pros:** more efficient when the data has compositional structure. Closer to the inner workings of the brain.
- **Cons:** harder to implement, learn and understand.

Connectionist models

List of arguably the most recognized connectionist models of the brain:

- Semantic Pointer Architecture (SPA)
- Leabra
- LISA
- DORA
- Neural Blackboard Architecture
- The Integrated Connectionist/Symbolic Architecture (ICS)

Connectionist models

List of arguably the most recognized connectionist models of the brain:

- **Semantic Pointer Architecture (SPA)**
- Leabra
- LISA
- DORA
- Neural Blackboard Architecture
- The Integrated Connectionist/Symbolic Architecture (ICS)

In the class, we will learn about *Semantic Pointer Architecture*, which is implemented in the **Nengo** python library.

Understanding the brain

We may construct all the artificial models we want, but our aim is understanding the real brain. We have no idea, if something can be computed by the brain, unless we carry it out in the similar architecture. (Chris Eliasmith)

Understanding the brain

What I cannot create, I do not understand. (Richard Feynman)

Semantic Pointer Architecture (SPA)

Chris Eliasmith

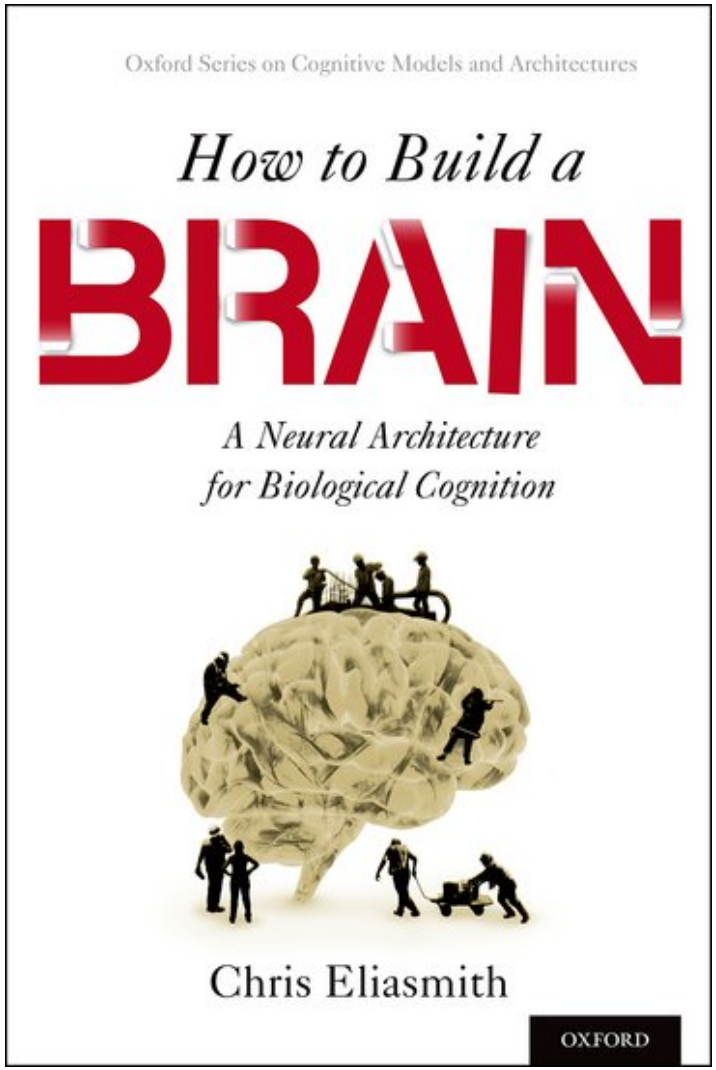
The director of the *Centre for Theoretical Neuroscience* on **University of Waterloo** (Canada). One of the main designers of Nengo and SPA.

"We have not yet learned what the brain has to teach us."

Webpage: http://arts.uwaterloo.ca/~celiasmi/.

“How to Build a Brain”, 2013

Materials in this course are in big part based on this book:



Let's listen to Chris Eliasmith

https://www.youtube.com/watch?v=g2HHJfovb5E

Main ideas

- "Symbols" as **vectors of numbers** in multidimensional space.

  $\circledast$ – circular convolution (pol. *splot cykliczny*)

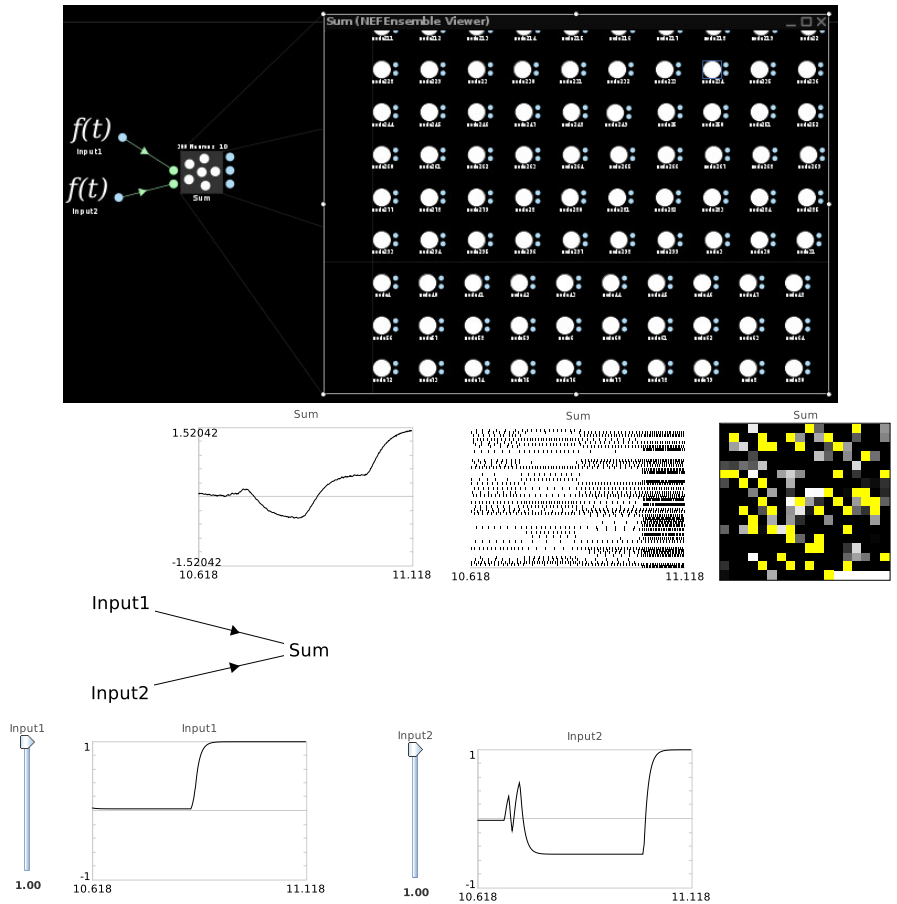
$$\begin{pmatrix} 1.69 \\ 1.69 \\ 1.73 \\ 1.77 \end{pmatrix} = \begin{pmatrix} 0.2 \\ 0.6 \\ 0.6 \\ 0.1 \end{pmatrix} \circledast \begin{pmatrix} 0.3 \\ 0.9 \\ 0.1 \\ 0.5 \end{pmatrix} + \cdots + \begin{pmatrix} 0.4 \\ 0.2 \\ 0.3 \\ 0.2 \end{pmatrix} \circledast \begin{pmatrix} 0.6 \\ 1.0 \\ 0.7 \\ 0.4 \end{pmatrix}$$

$$\mathbf{P} = \mathbf{agent} \circledast \mathbf{student} + \mathbf{verb} \circledast \mathbf{learn} + \mathbf{what} \circledast \mathbf{kogni}$$

$$\mathbf{P} \circledast \mathbf{agent}^{-1} = \mathbf{student} + \textit{noise} \approx \mathbf{student}$$
$$\mathbf{P} \circledast \mathbf{verb}^{-1} = \mathbf{learn} + \textit{noise} \approx \mathbf{learn}$$
$$\mathbf{P} \circledast \mathbf{what}^{-1} = \mathbf{kogni} + \textit{noise} \approx \mathbf{kogni}$$

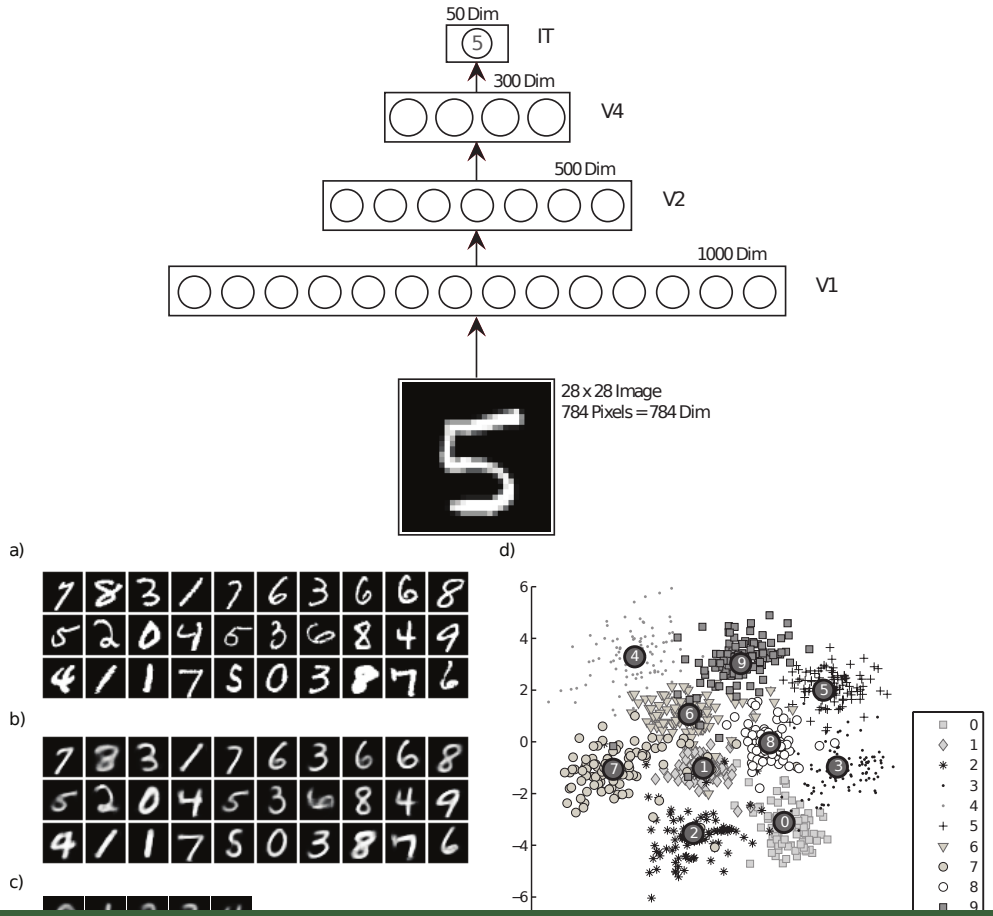
Main ideas

- Vectors may be relatively easily encoded by populations of **biologically realistic** spiking neurons.



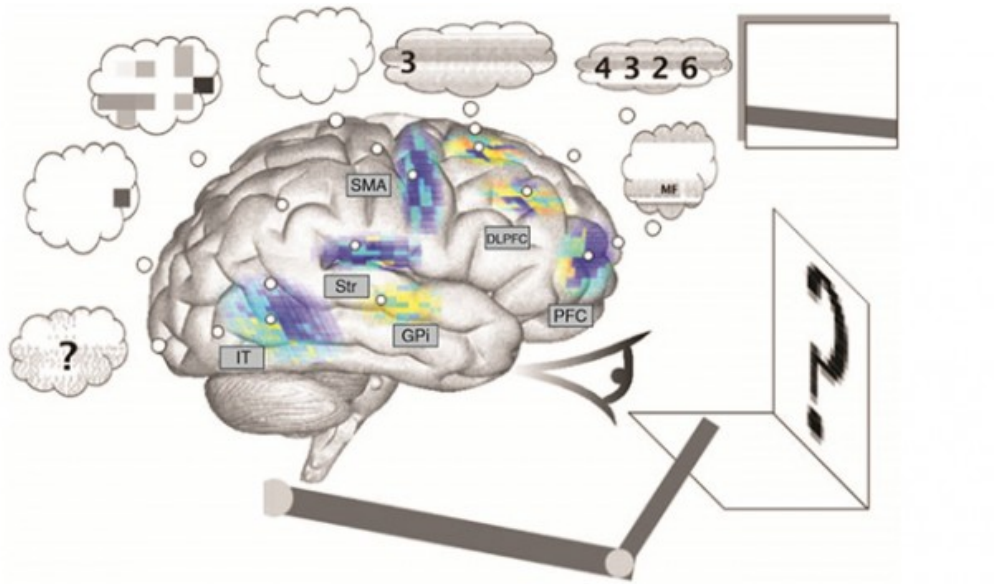
Main ideas

- Compression of information into the so called **semantic pointers**, which may be decompressed to restore content.



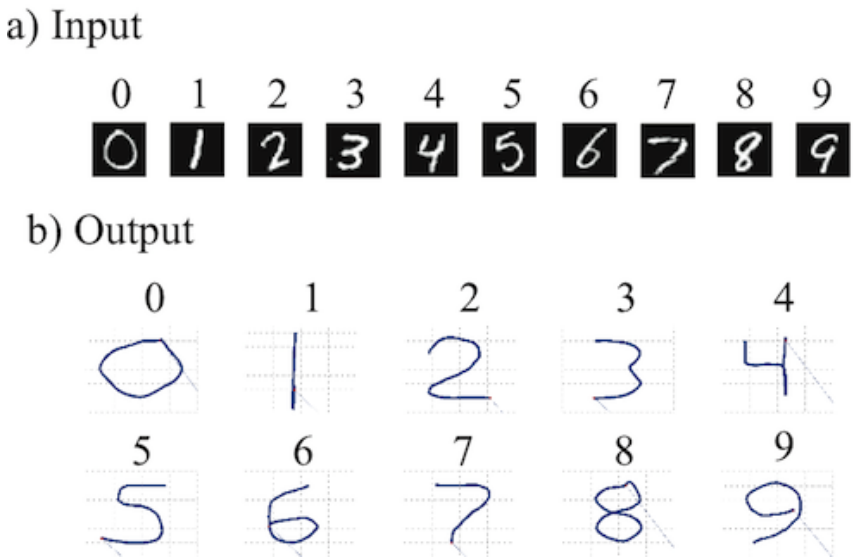
Spaun (2012)

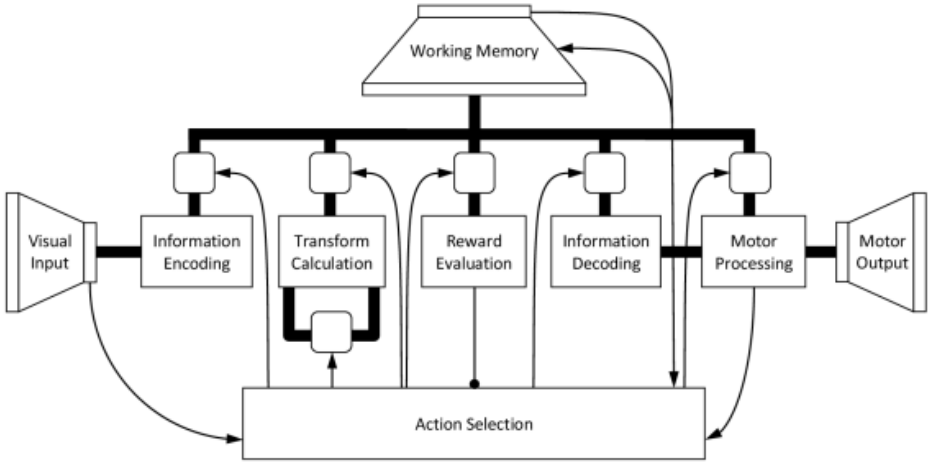
- Full name: **Semantic Pointer Architecture Unified Network**.
- Probably the biggest neurologically plausible model capable of realizing several different tasks.
- Publication [2]: `http://compneuro.uwaterloo.ca/files/publications/eliasmith.2012.pdf`



Spaun (2012)

- Uses simple leaky integrate-and-fire (LIF) neurons.
- 2.5 million spiking neurons, about 8 billion synaptic connections.
- RAM usage: 24 GB.
- Needs 2.5 hours to simulate 1 s.





Spaun (2012)

Realized tasks:

1. Copy drawing
2. Image recognition
3. Reinforcement learning (gambling)
4. Serial working memory (list memory)
5. Counting
6. Question answering
7. Rapid variable creation
8. Fluid reasoning

To learn more about tasks:
http://www.nengo.ca/build-a-brain/spaunvideos

Spaun (2012)

However, Spaun has little to say about how that complex, dynamical system develops from birth. Furthermore, Spaun has many other limitations that distinguish it from developed brains. For one, Spaun is not as adaptive as a real brain, as the model is unable to learn completely new tasks. In addition, both attention and eye position of the model is fixed, making Spaun unable to control its own input. (Eliasmith, et al. [2])

- There is still a long way to truly understand brain inner workings.
- Which means that there is still much to discover for cognitive scientists. :)

BioSpaun (2016)

- Version of the Spaun with detailed compartmental conductance model of the neurons.
- Authors demonstrate that BioSpaun can be used to simulate the effects of adding the drug tetrodotoxin to the simulated areas of cortex. (possible because of increased biological realism)
- Publication [1]: https://arxiv.org/pdf/1602.05220.pdf

Neural Engineering Framework (NEF)

NEF

*Conceptually, the NEF can be thought of as a **neural compiler** which allows a researcher to specify a computation as a general nonlinear dynamical system in some state space, which is then implemented in a spiking neural substrate using an efficient optimization method.* (Eliasmith, et al. [1])

- In short, NEF approximates mathematical computations as interactions between populations of spiking biologically-realistic neurons.
- User decides the function, so NEF is all about **implementing functions using the neurons**.

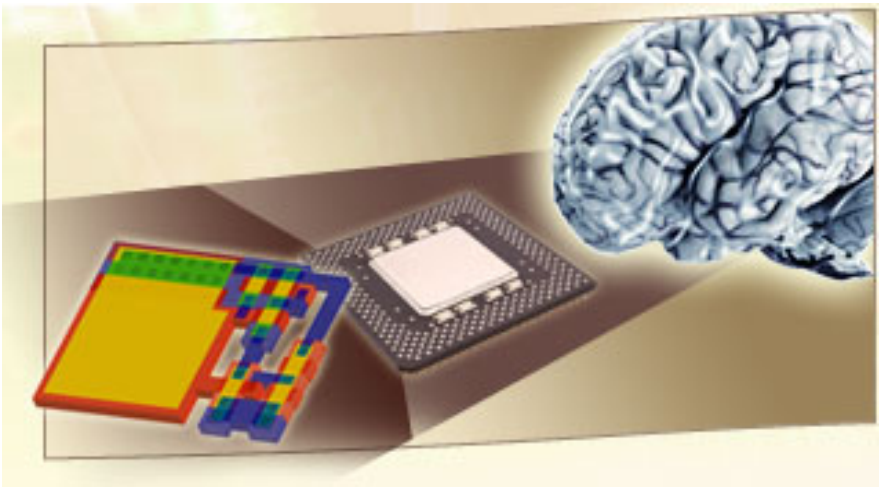
Nengo

Nengo

- Python library for building and simulating brain models using the methods of the Neural Engineering Framework (NEF).
- Implements the NEF and SPA methods.
- Website of the project: http://nengo.ca.

Neuromorphic hardware

- SpiNNaker (developed at University of Manchester)
  http://apt.cs.manchester.ac.uk/projects/SpiNNaker/
- Neurogrid (developed at Stanford University)
  https://web.stanford.edu/group/brainsinsilicon/index.html



Neuromorphic hardware

Computers' inherent limitations as neural simulation platforms are addressed by neuromorphic chips: Their fundamental component is not a logic gate but a silicon neuron.

Source: https://web.stanford.edu/group/brainsinsilicon/approach.html

NengoDL

- Deep learning[1] integration for Nengo.
- Source code: https://github.com/nengo/nengo_dl
- Documentation: https://nengo.github.io/nengo_dl/

Nengo DL is a backend for Nengo that integrates deep learning methods (supported by the TensorFlow framework) with other Nengo modelling tools. This allows users to optimize their models using deep learning training methods, improves simulation speed (on CPU or GPU), and makes it easy to insert TensorFlow models (such as a convolutional neural network) into Nengo networks. (Daniel Rasmussen, NengoDL developer)

[1] More about deep learning later.

Installation

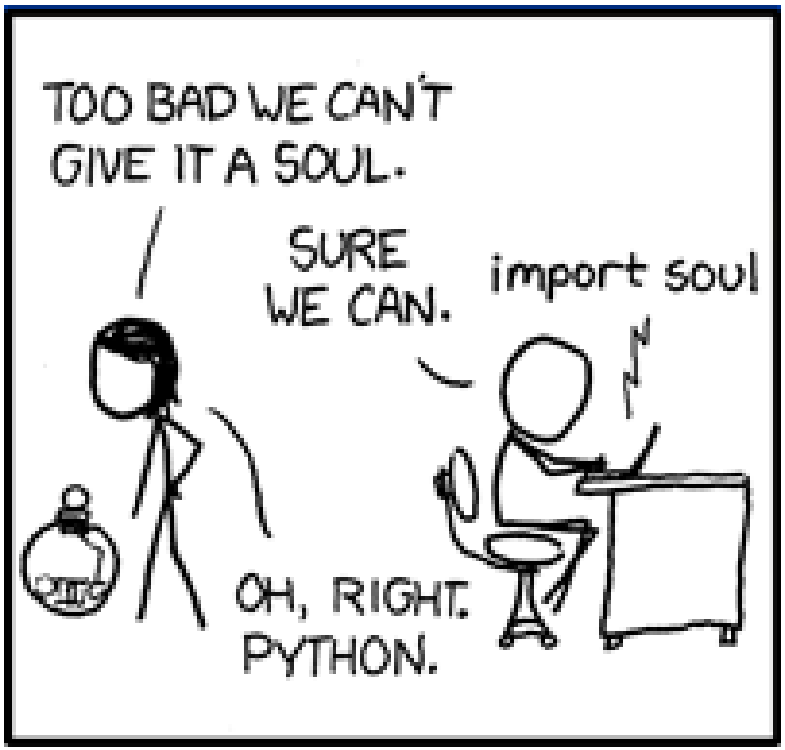
The easiest way to install the newest version of Nengo is to use standard python package manager: "pip install nengo_gui".

There are two major versions of Nengo:

- **Nengo 1.4** – Java application, in which simulations are built from elementary blocks, no programming required. This version of Nengo was also used in the tutorials in the Chris Eliasmith's book "How to Build a Brain".
- **Nengo 2.7** (as of May 2018) – a Python library. Simulations must be manually written in the form of Python scripts. There is a simple GUI which helps with a visualization of the network.

Installation

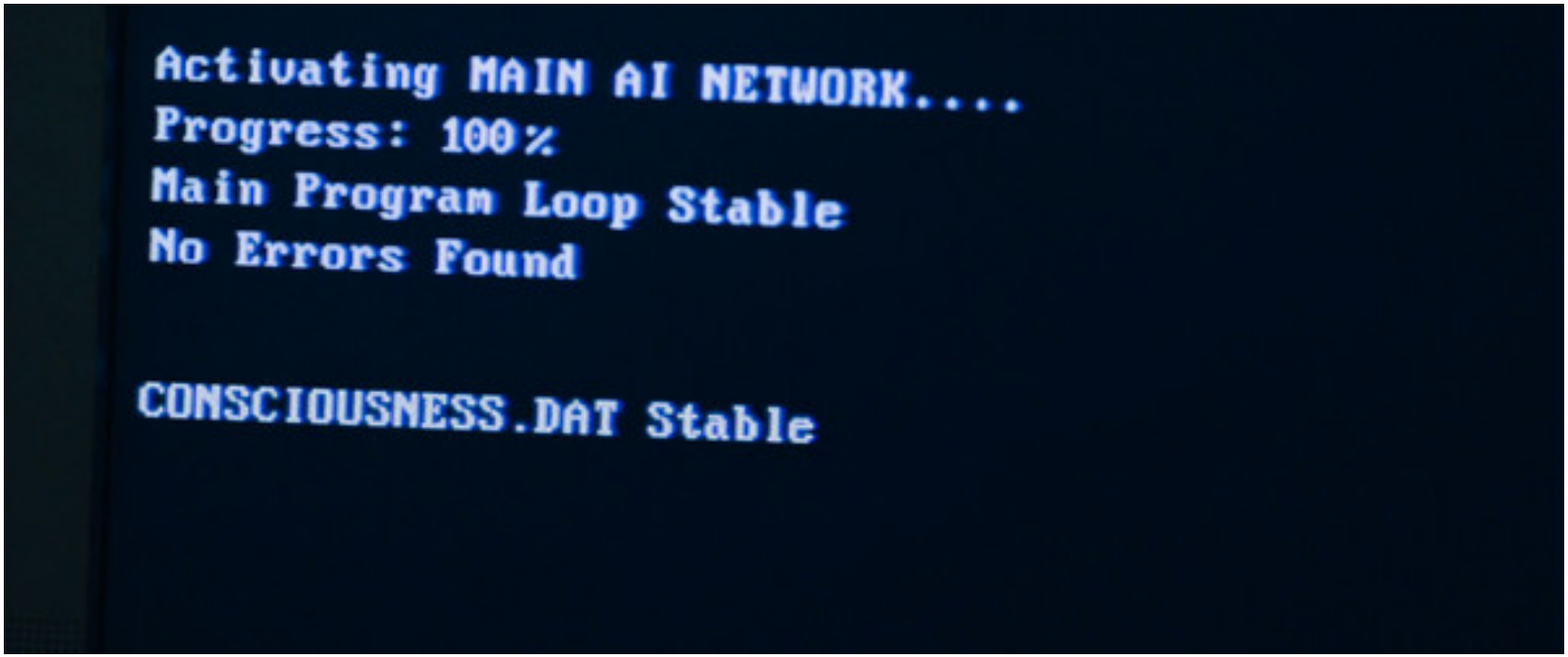
Python allows us to do many useful things (compulsory comics).



(Source: https://xkcd.com/413/)

Installation

But Nengo 1.4 Java GUI probably will suffice for our needs.



(Source: the movie *"Chappy"*)

Nengo 1.4

Steps to install Nengo 1.4:

1. Download Nengo 1.4 (link).
2. Unzip it.
3. Nengo 1.4 is portable, which means that no standard installation is required. To start the program just:
   - on Windows double-click *Nengo.bat*
   - on Linux run command: *./nengo*

Jupyter Notebooks

(In case someone would like to run examples in the Jupyter notebook.)

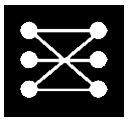
- Main page of the project: http://jupyter.org/.
- Needed for Nengo example notebooks (https://github.com/nengo/nengo_examples).
- Installation of python2 kernel:

```
# install the kernel package for Python 2
python2 -m pip install --upgrade ipykernel
# register the Python 2 kernelspec
python2 -m ipykernel install
```

Nengo 1.4 GUI

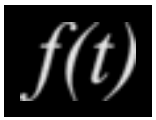
Basic Components in Nengo

We will mainly use the following basic components:

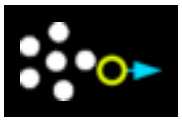


Network

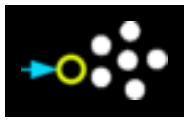
Ensemble



Input



Origin



Termination

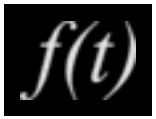
Network

- Contains all processing elements responsible for computing a certain "high-level" module.
- In the editor we place everything in a certain global network.

Ensemble

- Population of neurons.
- The most basic building block of simulations.

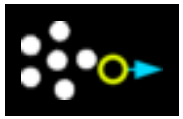
Input

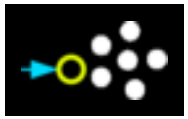


Input

- Input to the system, value of which may change during simulation ("naturally" or artificially by the user).

Origin/Termination

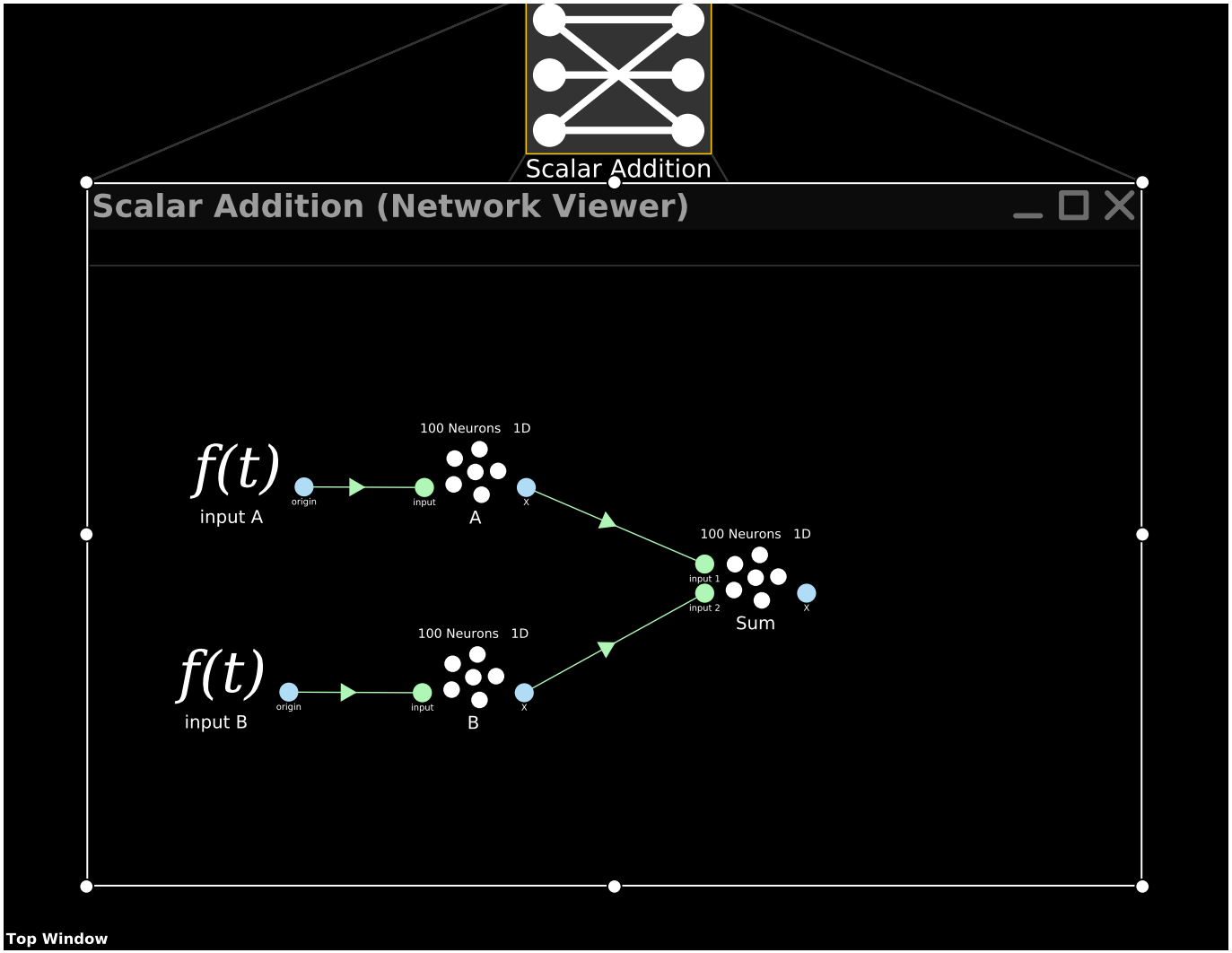
 **Origin**

 **Termination**

- Used to connect ensembles.
- *Origin* is always an output of a certain ensemble, and *Termination* is always an input of a certain ensemble (termination of connection).
- Specify transformation to be used.

Example

Network computing a sum of two variables.



Bibliography I

[1] Chris Eliasmith, Jan Gosmann, and Xuan-Feng Choo. "BioSpaun: A large-scale behaving brain model with complex neurons". In: *ArXiv* (2016).

[2] Chris Eliasmith et al. "A large-scale model of the functioning brain". In: *Science* 338 (2012), pp. 1202–1205. DOI: 10.1126/science.1225266. URL: http://nengo.ca/publications/spaunsciencepaper.