# Neural TTR

Robin Cooper
University of Gothenburg
cooper@ling.gu.se

Centre for Linguistic Theory and Studies in Probability,
http://clasp.gu.se/

# Outline

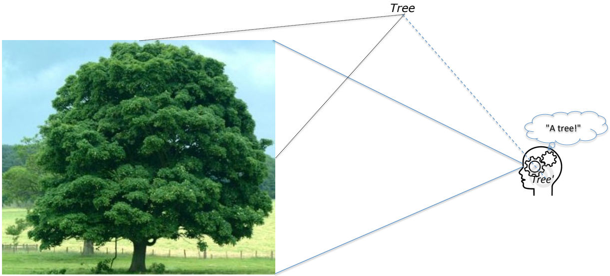
# Outline

# Type theory and cognition

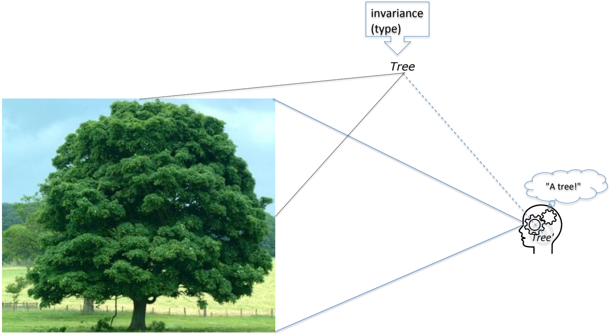
- TTR (Type Theory with Records) Cooper (2010, 2012); Cooper and Ginzburg (2015); Cooper (in prep), `https://sites.google.com/site/typetheorywithrecords/`
- a *rich* type theory (includes types of objects like *Tree* and events *boy-hugs-dog*)
- inspired by Martin-Löf type theory (Martin-Löf, 1984; Nordström *et al.*, 1990)
- central is the notion of *judging* an object (or event), $a$, to be of a type, $T$:

  $a : T$

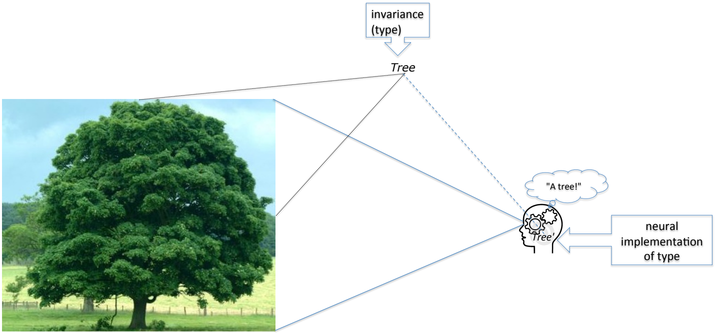
# Seeing a tree



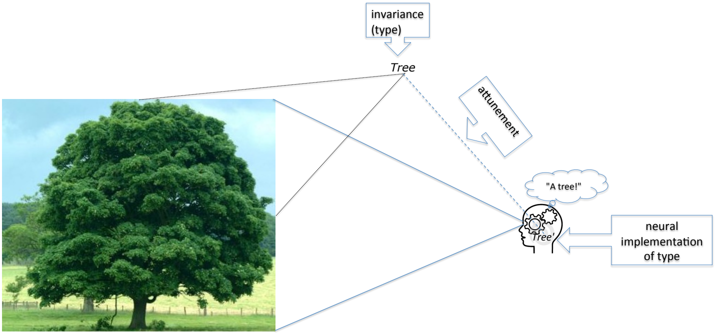
# Seeing a tree



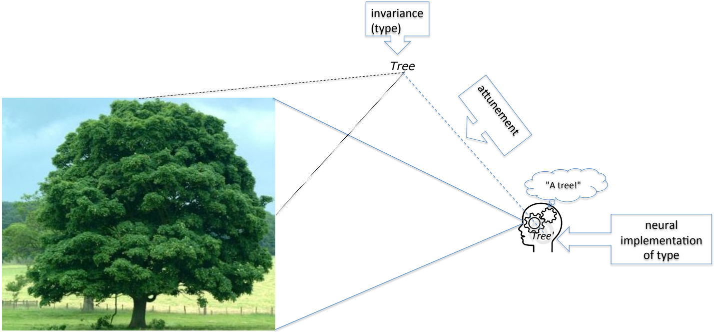
# Seeing a tree



# Seeing a tree

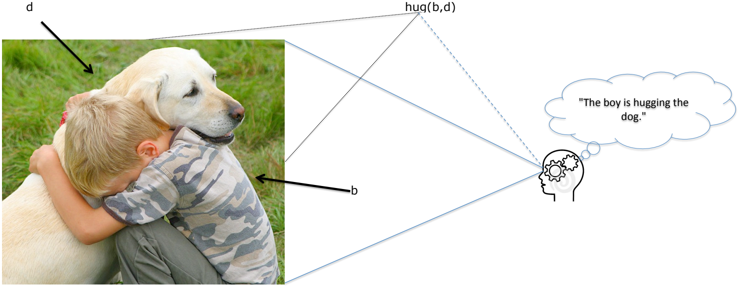


# Seeing a tree




Gibson (1986); Barwise and Perry (1983)

# Seeing a hugging event



# A boy hugs a dog (record type)

$$\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c_{boy}} & : & \mathrm{boy(x)} \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c_{dog}} & : & \mathrm{dog(y)} \\ \mathrm{e} & : & \mathrm{hug(x,y)} \end{bmatrix}$$

A record of this type:

$$\begin{bmatrix} \mathrm{x} & = & \mathrm{sam} \\ \mathrm{c_{boy}} & = & s_1 \\ \mathrm{y} & = & \mathrm{fido} \\ \mathrm{c_{dog}} & = & s_2 \\ \mathrm{e} & = & s_3 \\ \ldots & & \end{bmatrix} \quad \text{where:} \quad \begin{array}{l} \mathrm{sam} : \mathit{Ind} \\ s_1 : \mathrm{boy(sam)} \\ \mathrm{fido} : \mathit{Ind} \\ s_2 : \mathrm{dog(fido)} \\ s_3 : \mathrm{hug(sam,\ fido)} \end{array}$$

# Dependent types

- dependent types — functions from objects to types
- $\lambda v{:}\mathit{Ind}\ .\ \mathrm{boy}(v)$
- more precise rendering of *a boy hugs a dog*, a *dependent* record type

$$\left[\begin{array}{lll} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_{\mathrm{boy}} & : & \langle\lambda v{:}\mathit{Ind}\ .\ \mathrm{boy}(v),\ \langle\mathrm{x}\rangle\rangle \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c}_{\mathrm{dog}} & : & \langle\lambda v{:}\mathit{Ind}\ .\ \mathrm{dog}(v),\ \langle\mathrm{y}\rangle\rangle \\ \mathrm{e} & : & \langle\lambda v_1{:}\mathit{Ind}\ .\ \lambda v_2{:}\mathit{Ind}\ .\ \mathrm{hug}(v_1,v_2),\ \langle\mathrm{x},\mathrm{y}\rangle\rangle) \end{array}\right]$$

## Generalized quantifiers as relations between dependent types

- dog′ — $\lambda v{:}\mathit{Ind}\ .\ \mathrm{dog}(v)$
- run′ — $\lambda v{:}\mathit{Ind}\ .\ \mathrm{run}(v)$
- functions of type $(\mathit{Ind}{\rightarrow}\mathit{Type})$ — *properties*
- a ptype — every(dog′, run′)
  "the type of situations in which every dog runs"

# Relating to classical GQ theory

- if $T$ is a type, we use $[\check{\ }T]$ to represent $\{a \mid a : T\}$
- *propositions as types* — $T$ is "true" just in case $[\check{\ }T] \neq \emptyset$
- if $P$ is a property, we use $[\downarrow P]$ to represent the *extension of property $P$*:

$$\{a \mid [\check{\ }P(a)] \neq \emptyset\}$$

- $[\check{\ }\mathrm{every}(P, Q)] \neq \emptyset$ iff $[\downarrow P] \subseteq [\downarrow Q]$

# Outline

# Neuroscience fiction

- Cannot (yet) hope to observe brain activity corresponding to single types
- Available techniques (e.g. FMRI) do not have fine enough resolution
- Too much noise
- The hope: if we have a theory of what we might be looking for, then perhaps at some point we will be able to find it amongst all the noise

# Top-down *vs* bottom-up approach to neuroscience

bottom-up
- show a subject a picture of a boy hugging a dog
- see what is common in brain activity on the basis of a large number of trials

top-down
- create a theory which makes a prediction of brain activity corresponding to a boy hugging a dog
- test the prediction in subjects shown a picture of a boy hugging a dog

# Neural plausibility

- We do not know how a boy hugging a dog is represented
- . . . but we aim for neural plausibility based on what we do know about the brain

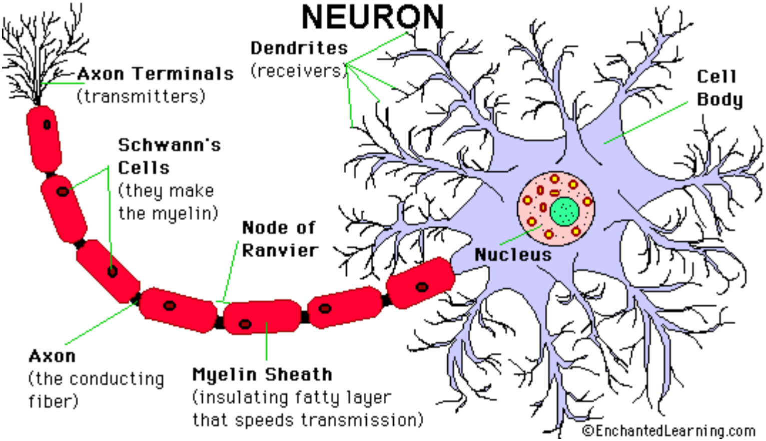
# Computational modelling

- Python implementation of TTR: `pyttr` (`https://github.com/GU-CLASP/pyttr`)
- a simple implementation of transparent neural networks (Strannegård and Nizamani, 2016), accessible to a treatment in terms of types of neural events (`https://github.com/GU-CLASP/pyttr/blob/master/neurons.py`)
- implementation of a mapping, $\nu$, from external (non-neural) types to types of neural events. (`https://github.com/GU-CLASP/pyttr/blob/master/nu.py`)

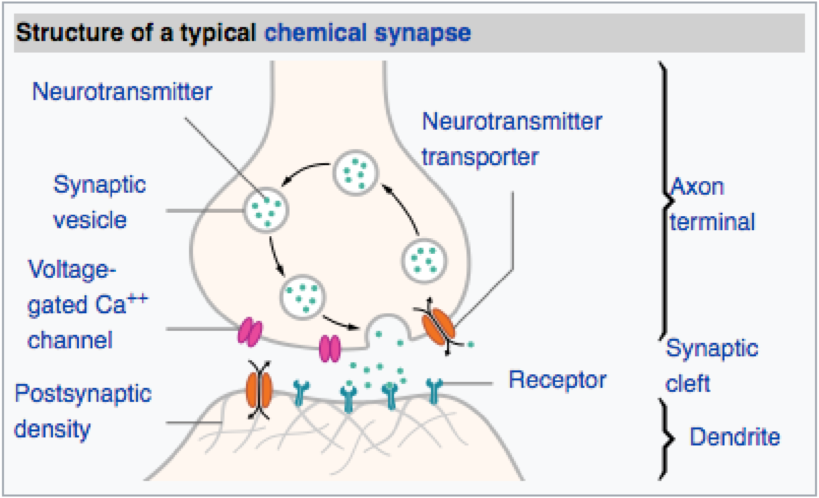
# What we are not doing (yet)

- machine learning — just the question of how types could be represented in a neurologically plausible network
- recognizing witnesses for types — just the representation of the types themselves
- Staffan Larsson will have something to say about how types can be related to classifiers as in machine learning — types used to classify external objects and events

# A neuron


NEURON
Dendrites
(receivers)
Axon Terminals
(transmitters)
Cell
Body
Schwann's
Cells
(they make
the myelin)
Node of
Ranvier
Nucleus
Axon
(the conducting
fiber)
Myelin Sheath
(insulating fatty layer
that speeds transmission)
©EnchantedLearning.com

# Synapse


Structure of a typical chemical synapse
Neurotransmitter
Synaptic vesicle
Voltage-gated Ca++ channel
Postsynaptic density
Neurotransmitter transporter
Receptor
Axon terminal
Synaptic cleft
Dendrite

## Some assumptions

- input on a dendrite can correspond to a real number
- output on an axon is boolean (based on a computation of dendritic input)
- there is some computation (converting a boolean to a real) carried out by a synapse
- a simplified representation of a neural state is a characterization of which neurons have active axons (i.e. output of 1)

# Outline

# TTR ptypes

- hug($a$,$b$) — *ptype*, constructed from a predicate and its arguments. Intuitively, a type of situation in which $a$ hugs $b$. ("True" if there is such a situation.)
- Representation of a history of activation on a network:

| | | | | | |
|---|---|---|---|---|---|
| `a` | 0 | 0 | 1 | 0 | 0 |
| `b` | 0 | 0 | 0 | 1 | 0 |
| `hug_n` | 0 | 1 | 0 | 0 | 0 |

# Phasing rather than temporal order of firing

- | | | | | | |
  |---|---|---|---|---|---|
  | a | 0 | 0 | 1 | 0 | 0 |
  | b | 0 | 0 | 0 | 1 | 0 |
  | hug_n | 0 | 1 | 0 | 0 | 0 |
  | ptype2 | * | 1 | 1 | 1 | 0 |
  | rel | * | 1 | 0 | 0 | 0 |
  | arg0 | * | 0 | 1 | 0 | 0 |
  | arg1 | * | 0 | 0 | 1 | 0 |
- Phasing rather than strict temporal order (*cf.* Shastri, 1999; Kiela, 2011)
- The network will create “labelling” neurons as needed during the course of a computation. (‘*’ represents neuron not present at time step.)
- Such neurons will remain available for future runs

# Structural modification in biological brains

- compare creation of neurons in computational model with making connections to unused neurons
- But also neurogenesis (structural plasticity) - hippocampus in London taxi drivers vs bus drivers (Maguire *et al.*, 2000, 2006)

## Some important principles

- *neural events* (with phasing) important for neural representation (rather than just neural architecture)
- *neural event types* can be *realized differently* on different networks (*cf.* work by Fedorenko). Which neurons are dedicated to a particular purpose depends in part on previous experience.
- a kind of *compositionality*. Whatever pattern of activation a network uses to represent 'hug' (firing of a single neuron or multiple neurons) that pattern of activation will occur in phase with a 'rel' pattern of activation in representing a ptype with 'hug'.

# Outline

# Recursion

- in the linguistic sense — types can be arguments within ptypes
- believe($c$, hug($a$,$b$))
  know($d$, believe($c$, hug($a$,$b$)))
- Another important thing to get right in a neural representation: an object can play more than one role in such a recursive structure.
  believe($a$, believe($b$, hug($a$,$b$)))

# Dealing with recursion

- Already have the tools we need
- Any snapshot of the network will have the capability to deal with a finite level of embedding
- if it encounters a deeper level it will create/devote the resources needed (up to limits on overall memory)
- *cf.* Christiansen and Chater (1999), a classic discussion of recursion in neural networks, where a network has a finite upper limit on depth of embedding.

# believe($c$, hug($a$,$b$))

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| a | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| b | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| hug_n | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| ptype2 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| rel | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| arg0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| arg1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| c | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| believe_n | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| ptype2 | * | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| rel | * | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| arg0 | * | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| arg1 | * | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

# Dependent types as functions of arbitrary depth

- Creating a dependent type in pyttr
- ```
  T = DepType('v',Ind,PType(hug,['v','b']))
  print(show(T))
  ```
  $\Rightarrow$
  ```
  lambda v:Ind . hug(v, b)
  ```
  ```

# Neural event representing this dependent type

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| b | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| hug_n | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| ptype2 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| rel | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| arg0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| arg1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| lambda | * | 1 | 1 | 1 | 1 | 1 | 0 |
| dom | * | 1 | 0 | 0 | 0 | 0 | 0 |
| var | * | 1 | 0 | 1 | 0 | 0 | 0 |
| rng | * | 0 | 1 | 1 | 1 | 1 | 0 |

## *Every dog runs*

- compositional combination of function representation and ptype representation
- `every(lambda x:Ind . dog(x), lambda x:Ind . run(x))`

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| every_n | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| dog_n | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| run_n | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Ind_n | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| ptype2 | * | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| rel | * | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| arg0 | * | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| arg1 | * | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| lambda | * | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| dom | * | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| var | * | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| rng | * | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| ptype1 | * | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| rel | * | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| arg0 | * | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

# Outline

# A problem with representing types as neural events

- What would it mean to know, believe or remember something?
- In type theoretic terms: what would it mean to store a judgement that some object (or event) is of a certain type (Cooper *et al.*, 2015)?
- Presumably not: constant repetition of events corresponding to what you have in memory
- Seems like we need something architectural after all

# Solution — memory neurons

- introduce *memory neurons* which, when activated, give rise to an appropriate neural event
- *cf.* *top-active* neurons representing concepts in Strannegård and Nizamani (2016)
- implementing this involves *delay* neurons which will delay firing until a later timestep (Strannegård *et al.*, 2015)
- for delay circuitry in nature (crickets) see Schöneich *et al.* (2015)

# Running a memory of hug($a$,$b$)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| `a` | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| `b` | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| `hug_n` | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| `ptype2` | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| `rel` | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| `arg0` | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| `arg1` | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| `hug(a,b)` | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| `Delay` | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| `Delay` | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| `Delay` | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| `Delay` | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

# Some top-level code

```
m = N.memorize_type(hug_a_b_n,'hug(a,b)')
N.ntrace()
m.excite()
N.run()
N.display_history()
```

# Outline

# Linking to the environment

- what we have seen so far says nothing about how these types are related to the environment
- ...or how a network could learn about this
- Maud is a virtual sheep in a virtual environment based on a simple example in Strannegård *et al.* (2017).
- `https://github.com/GU-CLASP/pyttr/blob/master/animat.ipynb`
- For simplicity Maud's types correspond to neural events where a single neuron is activated
- Shows how a simple variant of reinforcement learning can interact with neural TTR

# Maud — a simple sheep

- Maud's evironment: areas (in a one dimensional array) characterized by type *Green* (grass), *Blue* (water), *Brown* (sand), *Green*∧*Blue* (swamp)
- Maud's available action types: *Eat*, *Drink*, *MoveLeft*, *MoveRight*
- Maud has *Pleasure* and *Pain* neurons
  - Pleasure is activated when eats in a green area or drinks in a blue area (provided she is hungry or thirsty)
  - Pain is activated when eats or drinks in a green and blue area (and loses food and water)
  - Learning involves pleasure seeking and pain avoidance
- Maud dies if she has no food or no water in her body

# Maud's life expectancy

- With no learning and random actions, she survives for around ten actions
- With learning based on single types of the environment (no conjunction), she survives for around 15–20 actions
- With learning including conjunction of types of the environment, she apparently (with luck) can survive indefinitely

# Learning not to drink in the swamp

```
3
Green_n to Perception0 Pleasure: False
Blue_n to Perception0 Pleasure: False
Perception0 inhibit Drink_n Pain: True
```

```
----------- - - - - - - -
Green_n     0 0 1 1 1 1 1
Blue_n      1 0 0 1 1 1 1
Brown_n     0 0 0 0 0 0 0
Eat_n       0 0 0 0 0 0 0
Drink_n     0 0 0 0 1 1 1
MoveLeft_n  1 0 0 0 0 0 0
MoveRight_n 0 0 0 0 0 0 0
Pleasure_n  0 0 0 0 0 0 0
Pain_n      0 0 0 0 0 1 1
Perception0 * * * * * * 0
----------- - - - - - - -
food: 0.5800000000000001
water: 0.3800000000000001
loc: loc2
```

## Learning to eat when it's green

```
19
Green_n to Eat_n Pleasure: True
-----------  -  -  -  -  -
Green_n      0  0  1  1  1
Blue_n       0  0  0  0  0
Brown_n      1  0  0  0  0
Eat_n        0  0  0  1  1
Drink_n      0  0  0  0  0
MoveLeft_n   1  0  0  0  0
MoveRight_n  0  0  0  0  0
Pleasure_n   0  0  0  0  1
Pain_n       0  0  0  0  0
Perception0  0  0  0  0  0
-----------  -  -  -  -  -
```

```
food: 0.16000000000000011
water: 0.6600000000000001
loc: loc0
```

# Deciding not to eat in the swamp

```
47
-----------  -  -  -  -  -  -  -
Green_n      0  0  1  1  1  1  1
Blue_n       1  0  0  1  1  1  1
Brown_n      0  0  0  0  0  0  0
Eat_n        0  0  0  0  0  1  0
Drink_n      0  0  0  0  0  0  0
MoveLeft_n   1  0  0  0  0  0  0
MoveRight_n  0  0  0  0  0  0  0
Pleasure_n   0  0  0  0  0  0  0
Pain_n       0  0  0  0  0  0  0
Perception0  0  0  0  0  1  1  1
-----------  -  -  -  -  -  -  -
```

```
food: 0.8400000000000001
water: 0.9400000000000001
loc: loc2
```

# Future work

- learning involving more complex types
- proper reinforcement learning with policies, probabilities
- connecting to the world (and language) through conventional classifiers using conventional neural nets, e.g. connecting to Kille (de Graaf and Dobnik, 2015; Dobnik and de Graaf, 2017)
- implementing Staffan Larsson's work on perceptual meanings for spatial expressions in neural networks

# Conclusions

- simple-minded view of types related to perception and action as a basis for meaning
- a culture of neuroscience fiction — types correspond to events on a network
- the binding and recursion problem in terms of neural events and dynamic networks
- memory as the addition of a neuron which when activated will trigger a neural event corresponding to a type (delay circuitry important)
- something like reinforcement learning seems appropriate

## Bibliography I

Barwise, Jon and John Perry (1983) *Situations and Attitudes*, Bradford Books, MIT Press, Cambridge, Mass.

Christiansen, Morten H and Nick Chater (1999) Toward a connectionist model of recursion in human linguistic performance, *Cognitive Science*, Vol. 23, No. 2, pp. 157 – 205.

Cooper, Robin (2010) Frames in formal semantics, in H. Loftsson, E. Rögnvaldsson and S. Helgadóttir (eds.), *IceTAL 2010*, Springer Verlag.

# Bibliography II

Cooper, Robin (2012) Type Theory and Semantics in Flux, in R. Kempson, N. Asher and T. Fernando (eds.), *Handbook of the Philosophy of Science*, Vol. 14: Philosophy of Linguistics, pp. 271–323, Elsevier BV. General editors: Dov M. Gabbay, Paul Thagard and John Woods.

Cooper, Robin (in prep) Type theory and language: from perception to linguistic communication. Draft of book chapters available from `https://sites.google.com/site/typetheorywithrecords/drafts`.

Cooper, Robin, Simon Dobnik, Shalom Lappin and Staffan Larsson (2015) Probabilistic Type Theory and Natural Language Semantics, *Linguistic Issues in Language Technology*, Vol. 10, No. 4, pp. 1–45.

# Bibliography III

Cooper, Robin and Jonathan Ginzburg (2015) Type Theory with Records for Natural Language Semantics, in S. Lappin and C. Fox (eds.), *The Handbook of Contemporary Semantic Theory*, second edition, pp. 375–407, Wiley-Blackwell.

Dobnik, Simon and Erik de Graaf (2017) KILLE: a Framework for Situated Agents for Learning Language Through Interaction, in J. Tiederman (ed.), *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa, 22-24 May 2017, Gothenburg, Sweden*, p. 337, Linköping University Electronic Press, Linköpings universitet.

Gibson, James J. (1986) *The Ecological Approach to Visual Perception*, Lawrence Erlbaum Associates.

# Bibliography IV

de Graaf, Erik and Simon Dobnik (2015) KILLE: Learning Objects and Spatial Relations with Kinect, in *Proceedings of goDIAL - Semdial 2015: The 19th Workshop on the Semantics and Pragmatics of Dialogue*.

Kiela, Douwe (2011) Variable Binding in Biologically Plausible Neural Networks. Master's thesis, Universiteit van Amsterdam.

Maguire, Eleanor A., David G. Gadian, Ingrid S. Johnsrude, Catriona D. Good, John Ashburner, Richard S. J. Frackowiak and Christopher D. Frith (2000) Navigation-related structural change in the hippocampi of taxi drivers, *Proceedings of the National Academy of Sciences*, Vol. 97, No. 8, pp. 4398–4403.

# Bibliography V


Maguire, Eleanor A., Katherine Woollett and Hugo J. Spiers (2006) London taxi drivers and bus drivers: A structural MRI and neuropsychological analysis, *Hippocampus*, Vol. 16, No. 12, pp. 1091–1101.

Martin-Löf, Per (1984) *Intuitionistic Type Theory*, Bibliopolis, Naples.

Nordström, Bengt, Kent Petersson and Jan M. Smith (1990) *Programming in Martin-Löf's Type Theory* ( *International Series of Monographs on Computer Science* 7), Clarendon Press, Oxford.

# Bibliography VI

Schöneich, Stefan, Konstantinos Kostarakos and Berthold Hedwig (2015) An auditory feature detection circuit for sound pattern recognition, *Science Advances*, Vol. 1, No. 8, pp. e1500325–e1500325.

Shastri, Lokendra (1999) Advances in SHRUTI – A Neurally Motivated Model of Relational Knowledge Representation and Rapid Inference Using Temporal Synchrony, *Applied Intelligence*, Vol. 11, No. 1, pp. 79–108.

Strannegård, Claes, Simone Cirillo and Johan Wessberg (2015) Emotional Concept Formation, in *Proceedings of the Eighth Conference on Artificial General Intelligence*, pp. 166–176, Springer.

# Bibliography VII

Strannegård, Claes and Abdul Rahim Nizamani (2016) Integrating Symbolic and Sub-symbolic Reasoning, in *International Conference on Artificial General Intelligence*, pp. 171–180, Springer.

Strannegård, Claes, Nils Svangård, Joscha Bach and Bas Steunebrink (2017) Generic animats, in *Proceedings of the 10th International Conference on Artificial General Intelligence*, Melbourne.