Is Deep Learning relevant for the brain and mind?

ELSC Deep Learning Course

January 2020

Lior Fox

Introduction

Why bother?

- There are good reasons why “deep learning” is biologically implausible

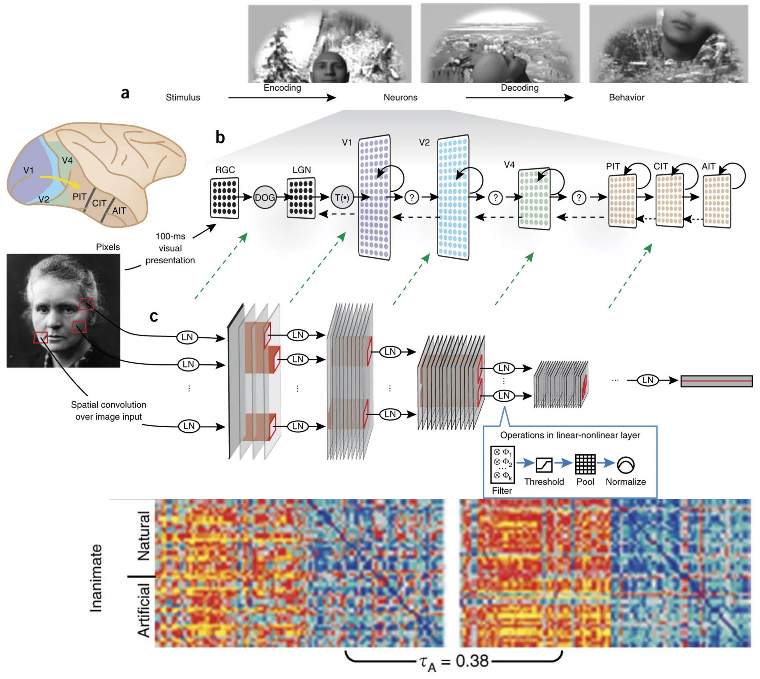
- Then again, for years deep learning “didn’t work” in ML either
- Hopefully, this course demonstrated to you that this is no longer the case

- **learning distributed, hierarchical** representations from data is highly powerful[1]
- In fact for some problems these are our best, if not only, working models
- “Deep” models turn out to be powerful models of cortical sensory processing

[1] *although stay tuned to Part II*

Quick example – CNNs as models for sensory-cortex

- CNNs as a model for neural encoding:
    - Hierarchical, simple-to-complex representations
    - Image computable
    - Trained on “natural” data
- The best current quantitative models for predicting neural responses in IT, etc.
- Some important aspects are not modeled
    - Feedback and lateral connectivity
    - Restrictive “behavioral” task
- We should at least re-evaluate our objections



Yamins, Daniel LK, and James J. DiCarlo. "Using goal-driven deep learning models to understand sensory cortex." Nature neuroscience 2016

Warmup – some traditional objections (1)

- Mainstream DL: “neurons” communicate continuous, real-valued activations
- Biological neurons communicate in spikes

- We are used to rate models, so this somehow doesn’t bother us
    - Activation can represent firing “rate”
    - Also, some works that attempt to bridge deep ANNs and spiking networks

- But this hints to some big questions
    - For example, should “units” in deep ANNs be thought of as neurons in the first place?

More traditional objections (2)

- A lot of the focus of DL has been on supervised learning
- Arguably, this is a limited model for biological learning
    - Also some technical questions – the source of supervision signal, etc.

- Current DL methods are now widely used in other forms of learning as well
    - Unsupervised and Reinforcement Learning are prominent examples
    - “Backpropagation” is not a supervised learning algorithm

- There are more challenges, depending on what we think we try to model
    - e.g., feed-forward architectures vs. recurrent/feed-back models

Part I

Dynamics and computation

Towards more biologically plausible learning rules for “deep” models

How to learn deep architectures?

- Need to solve the **credit assignment** problem
- A natural choice is *gradient learning* on a loss function: $\Delta W_{ij} \propto -\frac{\partial \mathcal{L}}{\partial W_{ij}}$
- For neural networks, backpropagation is a dynamic-programming that let us compute all these gradients (w.r.t all weights) in one forward+backward passes

Reminder: Backpropagation

- **Forward Pass**:
  - Set activation in input layer **a**^{1}=**x**
  - Compute input and activations for all units in the network, *l=2,...,L*:

$$\mathbf{z}^l = W^l \mathbf{a}^{l-1} + \mathbf{b}^l, \qquad \mathbf{a}^l = \sigma\left(\mathbf{z}^l\right)$$

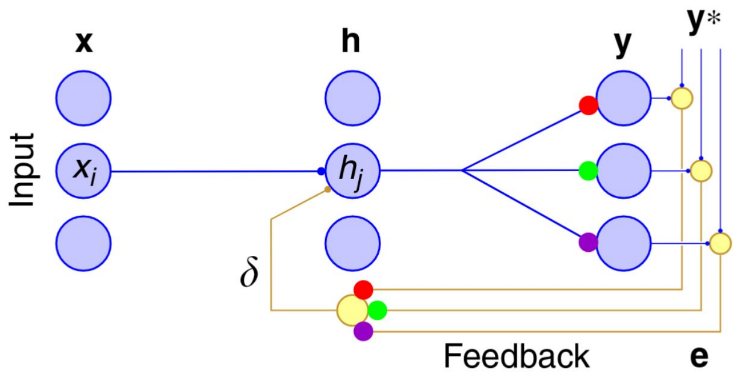
- **Backward Pass**:
  - Get target **y** and compute loss function $\mathcal{L}$
  - Calculate the *errors* – derivative of loss w.r.t to **layer input** for all layers, backward:

$$\boldsymbol{\delta}^L = \nabla_{a^L} \mathcal{L} \odot \sigma'\left(\mathbf{z}^L\right), \quad \boldsymbol{\delta}^l = \left(\left(W^{l-1}\right)^\top \boldsymbol{\delta}^{l+1}\right) \odot \sigma'\left(\mathbf{z}^l\right)$$

  - Get the gradients of loss w.r.t **weights**:

$$\frac{\partial \mathcal{L}}{\partial W^l_{jk}} = a^{l-1}_k \delta^l_j$$

Issues with backpropagation



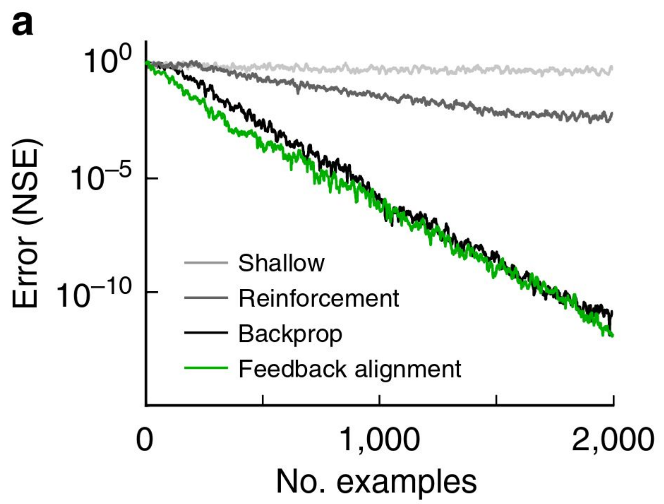
- Learning rule is non-local (the "weight transport" problem)
  - Have to send the error *backward* using the ***same weights*** used to propagate activity *forward*
- Activity in neurons has to "represent" ***two different things***:
  - Forward pass: features, activations, etc
  - Backward pass: derivative of loss w.r.t input (generally, an error term)
- More generally, different types of computation for forward/backward passes
  - Also, has to be 'synchronized' somehow

Note: If we have computed $\boldsymbol{\delta}^k$, updating $\mathbf{W}^k$ can be done by a Hebbian-like rule

Overcoming the weight transport issue

- BP requires precise, symmetric feedback
- But it is reasonable to assume *some* feedback

- What if we use random backward connections?
- This sounds like it shouldn't work...
  - ... but it turns out that it can actually work



Lillicrap, Timothy P., et al. "Random synaptic feedback weights support error backpropagation for deep learning." *Nature communications* 2016

Feedback alignment

- The entire logic of the method is in the calculation of ***δ***:

$$\boldsymbol{\delta}^l_{\text{BP}} = \left(W^{l-1}\right)^\top \boldsymbol{\delta}^{l+1} \qquad \boldsymbol{\delta}^l_{\text{FA}} = B^l \boldsymbol{\delta}^{l+1}$$

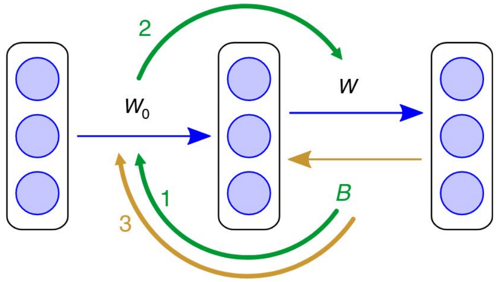
- Three basic observations:
  - We don’t necessarily need **W**$^\top$. Any matrix **B** suffices as long as on average $\boldsymbol{\delta}^\top_{\text{BP}} \boldsymbol{\delta}_{\text{FA}} > 0$
  - Even if this doesn’t hold initially, learning cold push **W** and **B** to “align”
  - In fact, this can be done by only learning **W**, while using a fixed random **B**.

(The formulas are for linear neurons for simplicity. Extension to non-linear activation is straightforward)

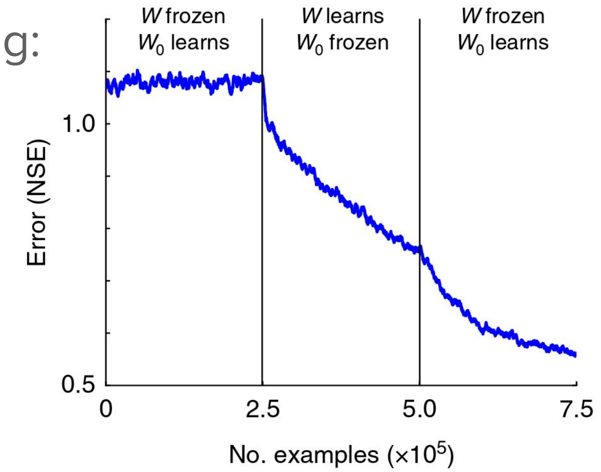
Why does this work?

Learning pushes **W** and **B** to “align”, so that **B** can send useful teaching signal

- This alignment is a 2-stages process
  - **W_0** accumulates information about **B*Error**
  - **W** aligns with **B** (accumulates information from **W_0**)



- Can see this by artificially separating the learning:
  - Freeze **W**, learn **W_0**
  - Freeze **W_0**, learn **W**
  - Freeze **W**, learn **W_0**



What happens in the first stage?

- The error doesn't go down, so it seems nothing interesting is going on
  - But sudden drop at the beginning of stage 2 hint that perhaps there is more to it
- Indeed there is no reason for it to go down, as the information that is sent backward tells the units nothing about how to decrease the loss
- But, the information can still be useful for adapting $\mathbf{W}_0$

Intuition for feedback alignment

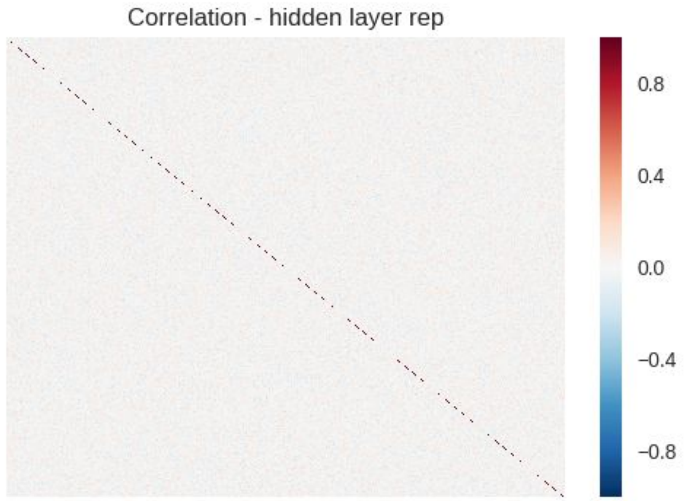
- Consider a multiclass classification task, (Softmax last layer + CE loss)
- What happens when we get an example from class 3?

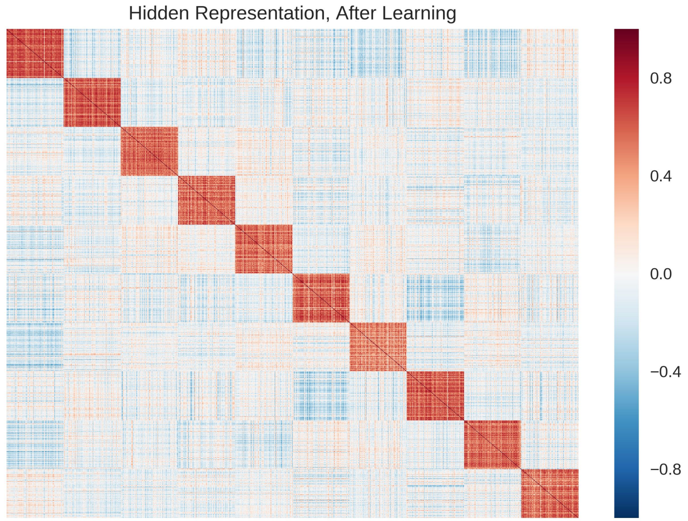
$$\begin{array}{ccllllllllll}
\hat{\mathbf{y}} & \approx & [0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1] \\
\mathbf{y} & = & [0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0] \\
\boldsymbol{\delta} & = & [0.1 & 0.1 & -0.9 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1]
\end{array}$$

- The network is initially random, so it predicts more or less uniformly
- So the *error derivatives* are approximately the same for all class members
- Which means they will be roughly the same when sent backward using **B**
- **Even if the examples were very different in input space**

Intuition for feedback alignment – extreme example

1000 binary random vectors were assigned to 10 classes. 784-400-10 network with hidden *tanh* units, training as in previous slide (fixed top layer, random feedback).

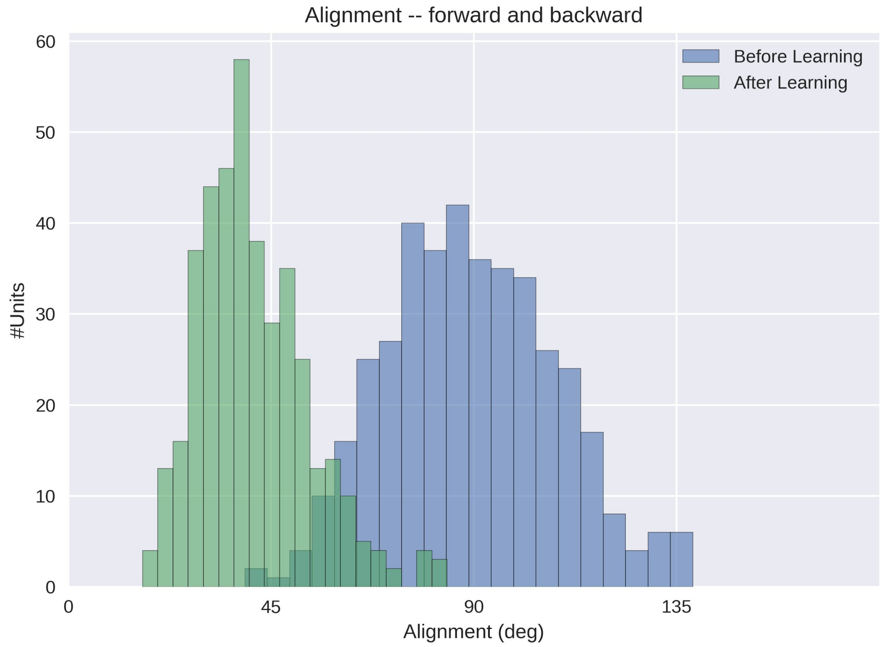




Replicated based on Geoffrey Hinton talk "Can the brain do back-propagation"

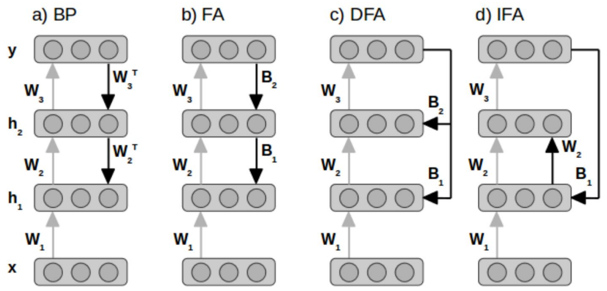
Feedback alignment with an actual alignment of feedback

In the same “extreme example”, if both layers are learned concurrently, error does go down and the last layer align with the random backward connections:



Feedback alignment – remarks and conclusions

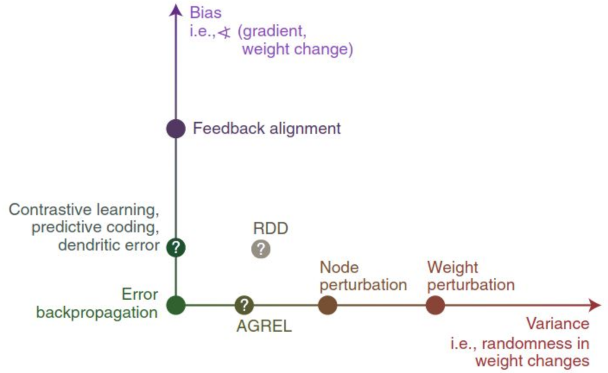
- Learning result from complex, non-trivial dynamics
  - The dynamics prescribed by FA does not follow the gradient of any function, let alone the loss
  - Still the network can learn useful things, so that later on “shallow”/quick adaptations are easy
  - Can be analytically handled in some simple cases (i.e linear nets; beyond our scope for today)
- How scalable the method is remains a question[1]
  - FA might be sub-optimal compared to “full-blown” BP
  - There have also been more variants of the idea[2]
- FA struggles with more ‘constrained’ architectures
  - Narrow bottlenecks (e.g typical autoencoders)
  - Convolutional NNs (more specifically, the weight sharing)



[1]Bartunov, Sergey, et al. "Assessing the scalability of biologically-motivated deep learning algorithms and architectures." NeurIPS 2018
[2]Nøkland, Arild. "Direct feedback alignment provides learning in deep neural networks." NeurIPS. 2016.

Feedback alignment – remarks and conclusions

- “Exact” gradient learning is an extreme case of a larger continuum of possible learning rules.
- Other extrema might be ‘nudge connections at random and correlate changes to reward’


Bias
i.e., ∢ (gradient, weight change)
Feedback alignment
Contrastive learning, predictive coding, dendritic error
RDD
Node perturbation
Weight perturbation
Error backpropagation
AGREL
Variance
i.e., randomness in weight changes

- Studies with ‘biological’ motivation can teach us things about ANNs

- Then again, it’s not clear how well are these suggestions supported or grounded in the biology
    - There might be opportunities here for systems-neuroscientists…

Richards, Blake A., et al. "A deep learning framework for neuroscience." Nature neuroscience 2019.

Is error representation required?

- It might be possible to learn deep networks without “asking” the neurons to represent two different things at different stages

- “Contrastive” learning has typically two stages, but the dynamics/computation of the neurons is the same

- Overall structure
  - Present an input to the network, compute predictions
  - Present the required output/target to output neurons (clamping or “nudging” them)
  - Let the network dynamics re-settle
  - The difference in neural activities in two phases can serve as a teaching signal

Hinton, Geoffrey E. "Training products of experts by minimizing contrastive divergence." Neural computation 2002
Xie, Xiaohui, and H. Sebastian Seung. "Equivalence of backpropagation and contrastive Hebbian learning in a layered network." Neural computation 2003
Scellier, Benjamin, and Yoshua Bengio. "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation." Frontiers in comp. neur. 2017

Intuition for contrastive learning

- Consider a Hopfield network with its standard learning rule, after presenting the μ example:

$$\Delta W_{ij} \propto p_i^\mu p_j^\mu$$

- The update *decrease energy* of the state $\mathbf{p}^\mu$, making it a stronger attractor

- To overcome spurious memories, we can let the network settle to a state **p'** starting from noise input. This will find an attractor state that is unrequired, and try to *increase* its energy ("unlearning" **p'**):

$$\Delta W_{ij} \propto -p_i' p_j'$$

Hopfield, John J., D. I. Feinstein, and R. G. Palmer. "'Unlearning' has a stabilizing effect in collective memories." Nature 1983

Contrastive methods – remarks

- Two phases (“free/clamped”, “wake/sleep”, etc), but with similar dynamics
- Some general intuition or hypothesis that this might be related to sleep?
  - Is there any evidence for that?
- A lot of these models assume or require symmetric connections
  - It is not fully clear how to combine solutions from the previous discussion with these models
- Other suggestions exist for unifying learning and inference dynamics
  - e.g, generative models perspective (“Wake-sleep” algorithm, some predictive-coding models)

Hinton, Geoffrey E., et al. "The "wake-sleep" algorithm for unsupervised neural networks." Science 1995.
Rao, Rajesh PN, and Dana H. Ballard. "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects." Nature neuroscience 1999

Part II

learning, generalization, and understanding

Some lessons from Reinforcement Learning

Reinforcement learning: a quick reminder

- Learn a ***policy*** – a mapping from states to actions – that yields maximum expected return, in the long run
    - Often learned through *action-values* (e.g. Q-Learning and variants)

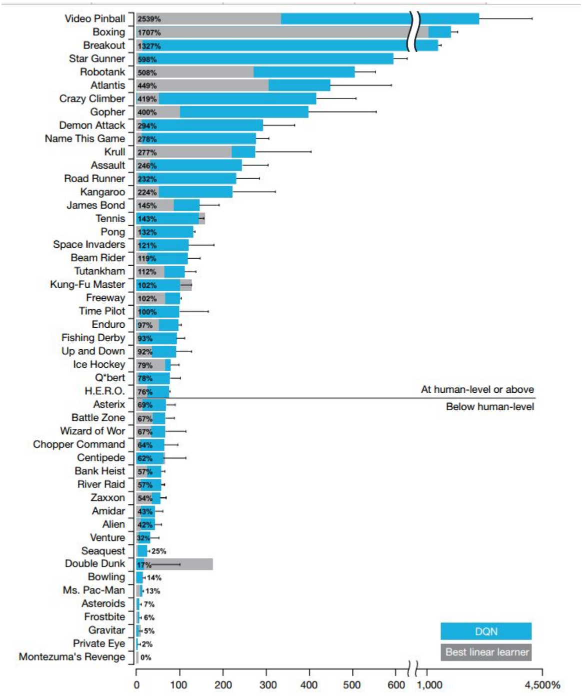
- Key challenges compared to supervised learning:
    - Actions affects the environment (what examples the agent get to see)
    - Actions has long-term consequences, in terms of reward
    - No “right answer”/labeling, only reward signal

- A compelling model for learning complex behavior in a more “natural” way

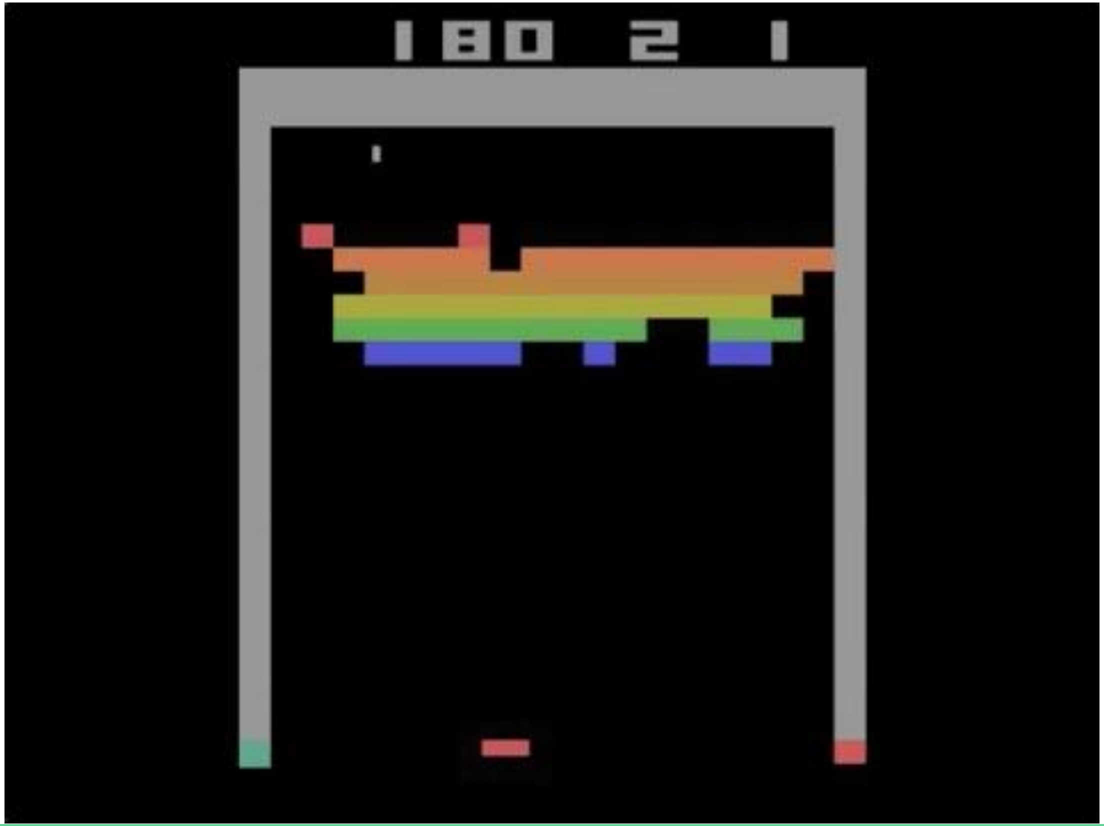
- From RL to Deep RL:
    - Parameterize policy and/or value-function by a neural network

Successes of Deep RL

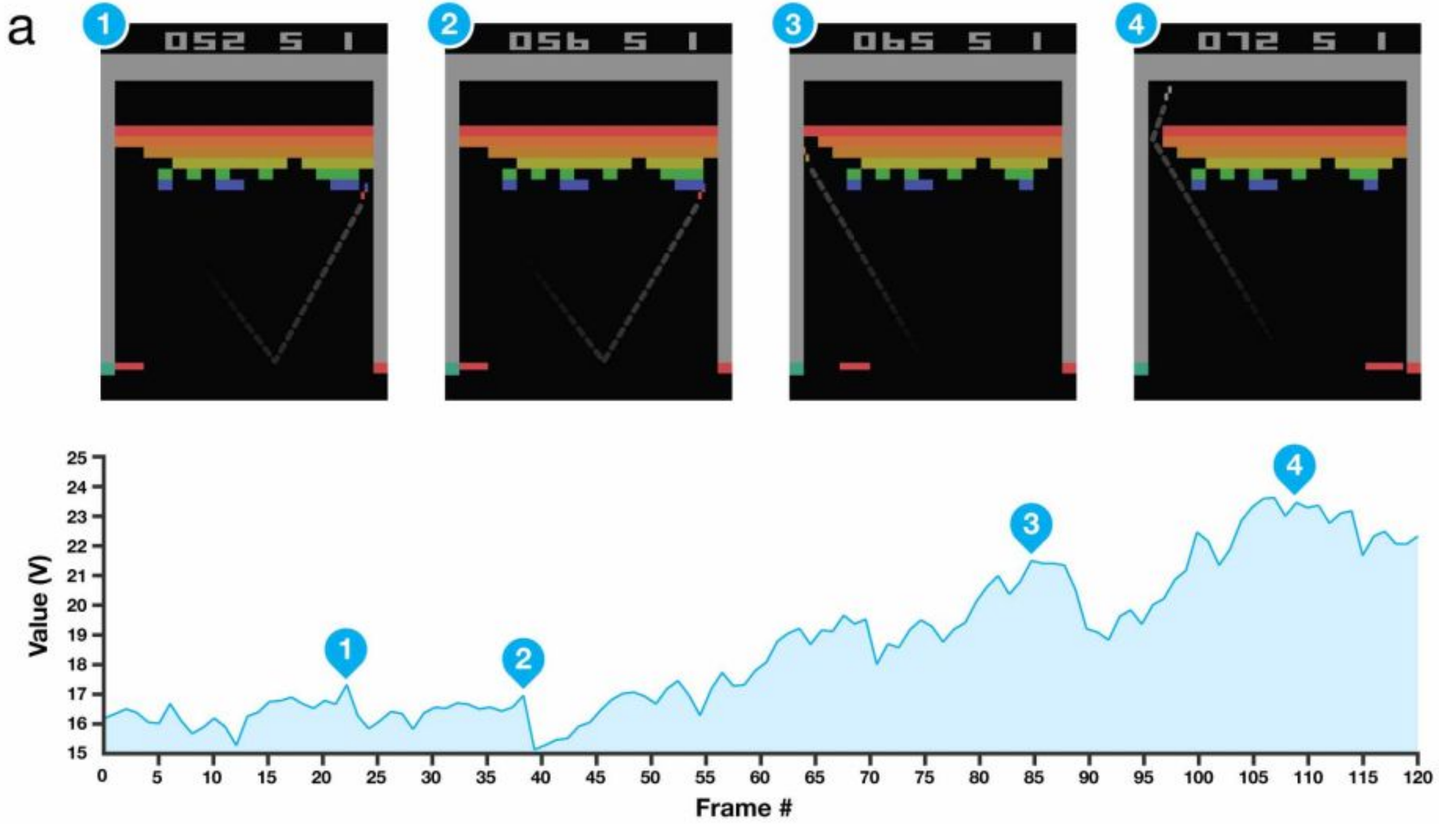
- In the past few years, deep RL methods showed significant success
  - Mostly on game playing (Atari, GO, Starcraft, etc.)
  - To some extent on [simulated] robotic tasks
- There are $N \rightarrow \infty$ variants and algorithms
  - we will not discuss them in detail here in detail
- One of the first important models was DQN, trained on the Atari2600 suite
  - “Superhuman” performance on many games
  - [Complete failure on some other games]
  - Importantly – one model can play many games
  - Training is for each task separately



Closer look: “breakout”



Closer look: “breakout”



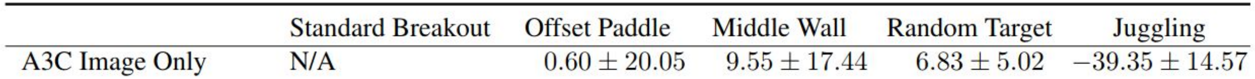
Does the trained network understand “Breakout”?

- Obviously at some level, as it can play it [and play it rather well]
- Seemingly, it can also make useful predictions about “semantic” aspects
- What are good representations?
  - When we say this word, we intuitively think about concepts – the paddle, bricks, score, etc.
  - The learned deep “representations” might be very different
- What is “understanding of breakout?”
  - This is not only a philosophical question

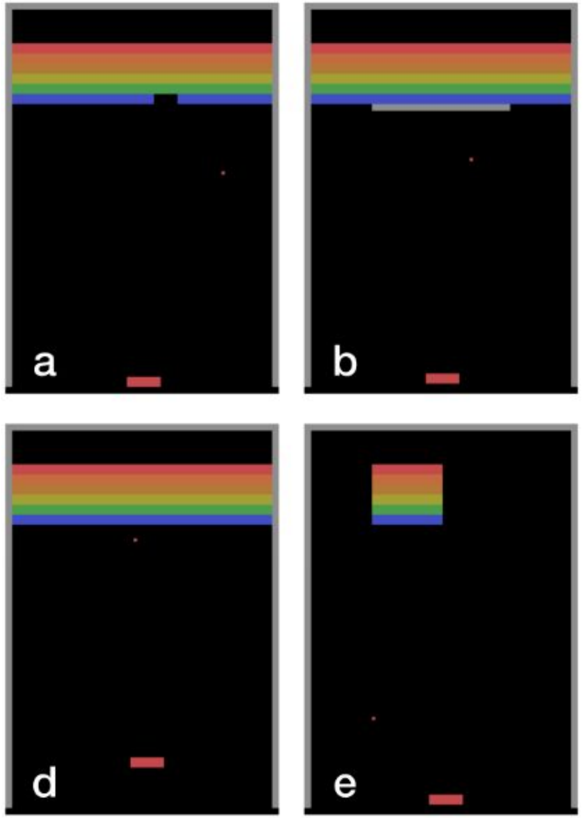
Understanding should let you adapt

- Consider some variants of breakout
    - Original (a)
    - Middle wall (b)
    - Offset paddle (d)
    - Random target (e)

- But standard deep RL agents trained agent on (a) fails almost completely on other variants:



	Standard Breakout	Offset Paddle	Middle Wall	Random Target	Juggling
A3C Image Only	N/A	0.60 ± 20.05	9.55 ± 17.44	6.83 ± 5.02	-39.35 ± 14.57



Kansky, Ken, et al. "Schema networks: Zero-shot transfer with a generative causal model of intuitive physics." ICML 2017.

Is this nothing but an extreme case of overfitting?

- Not a “generalization” issue in the DL commonly-used sense
  - Test examples are really from a “different distribution”
  - Pure statistical learning without further assumptions is not helpful for this kind of learning

- *Interpolation* versus *extrapolation*

- This happens not only in RL
  - Natural language processing is a good place to look for examples (surprising result: it is possible to translate pretty good without much language understanding)

- **Lesson 1**: Sophisticated behavior does not require or entail “understanding”

Marcus, Gary F. "Rethinking eliminative connectionism." Cognitive psychology 1998
Marcus, Gary F. "Deep learning: A critical appraisal." arXiv preprint 2018

Some ingredients which seem to be missing

- Model causal relationships
- Handle “non-stationary” elements, environments, etc
  - And sometime non-stationarity comes from the agent’s own actions
- Draw conclusions or learn based on very few examples, and without “forgetting” things learned previously

Are there ways to get what’s missing?

- Representation of discrete, algebraic-like entities (“symbols”), the relationships between them, and the inference rules to process these
  - Think about code in high-level languages like python

- Handling uncertainty with generative/causal models

- These ideas are not mutually exclusive with (statistical) *learning*
  - But just saying “give me a large enough data-set and a GPU to burn and I shall move the earth” is probably not going to solve these kind of problems

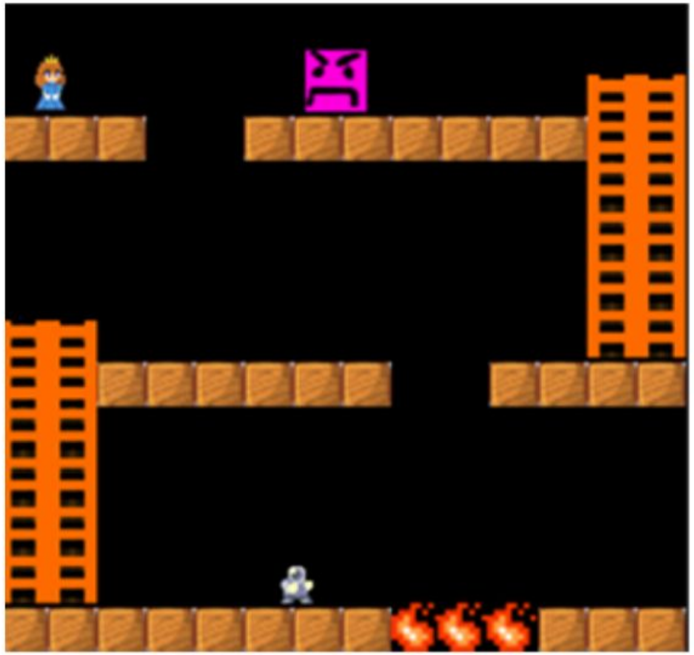
Lesson 2: Statistical learning is [probably] not the final answer to AI, or to cognitive sciences. The fact that DL is dominant today, doesn’t mean that the almost century-long debate of connectionists and symbolic methods is settled

Can’t DL just “figure it out”?

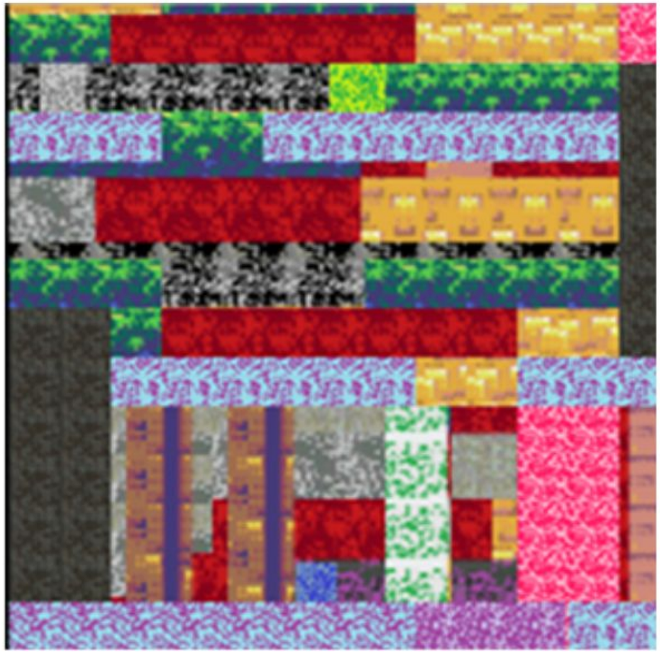
- Maybe
- But it seems that this require strong prior knowledge and structure of the architectures, objectives, learning rules, etc
    - e.g, “there are *objects* in the world”
- Prior knowledge is extremely important in learning, and our ability to know how to build it into ANNs is currently rather limited

Final example – the importance of prior knowledge

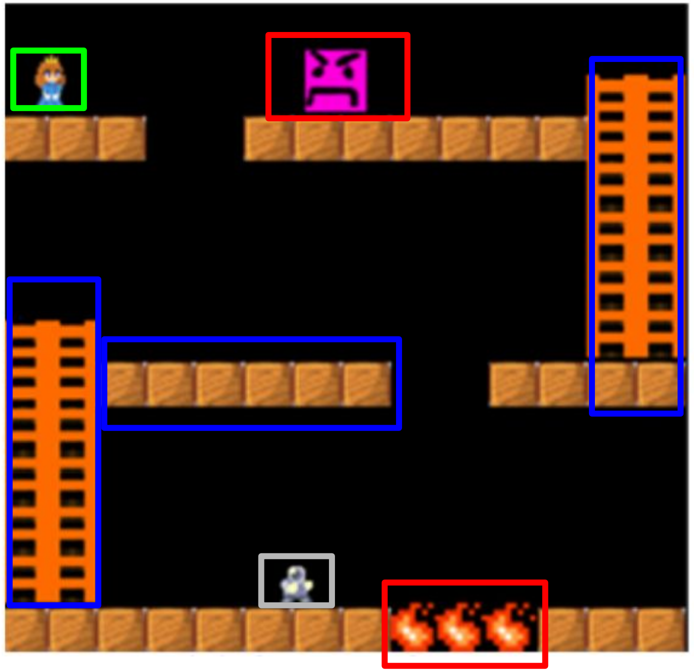
How about this one?

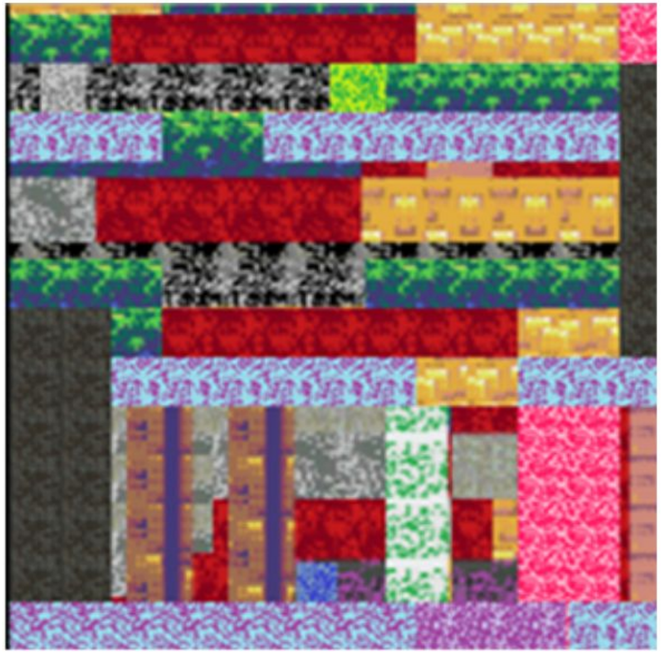


Do you know how to play this game?



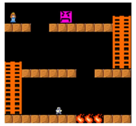
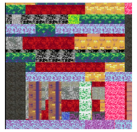
These are “the same” game





How long does it to take to learn these?

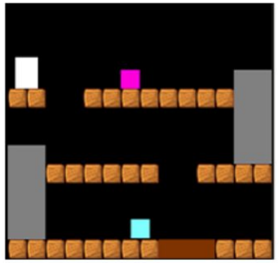
Human Subjects

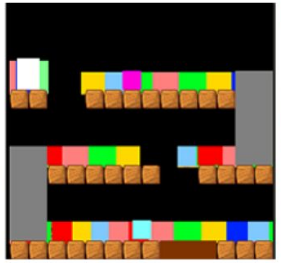
- <1 minute (~3000 actions) for 
- 2 minutes (~6500 actions) for 

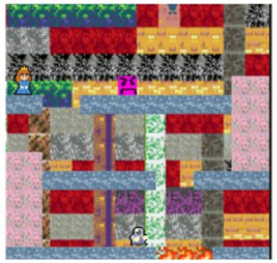
RL Agent

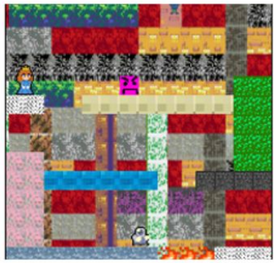
- Standard methods are too inefficient to even finish individual games [sparse reward]
- Using a more sophisticated exploratory agent, about **4,000,000 actions** for both versions

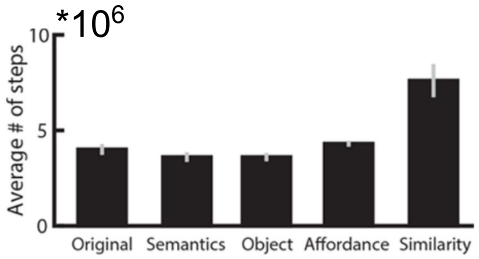
Big effect on human subjects – does masking such visual priors affect RL agents?


(a) Masked semantics


(b) Masked object identities


(c) Masked affordances


(d) Masked similarity



(e) Performance of RL agent

Some concluding remarks for Part II

- At least some of these questions are being actively studied today
  - Some partial answers exist
  - But it is important to first recognize the questions
  - Obviously there are also more challenges that we didn’t cover
- We should be careful about overattribution of deep ANNs
- How can insights from cognitive and developmental psychology help?