Inductive Biases for Higher-Level Cognition Deep Learning

Anirudh Goyal
Mila, University of Montreal

October 23, 2020

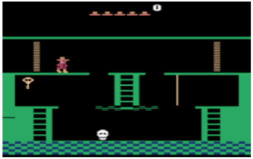
Papers under Discussion

- Recurrent Independent Mechanisms, arxiv
- Object Files and Schemata: Factorizing Declarative and Procedural Knowledge in Dynamical Systems, arxiv
- A Meta-Transfer Objective for Learning to Disentangle Causal Mechanisms (Bengio et. al) (ICLR'20)
- Learning Neural Causal Models from Unknown Interventions (Ke. et. al)
- Learning to Combine Top-Down and Bottom-Up Signals in Recurrent Neural Networks with Attention over Modules (Mittal. et. al) (ICML'20)

Outline

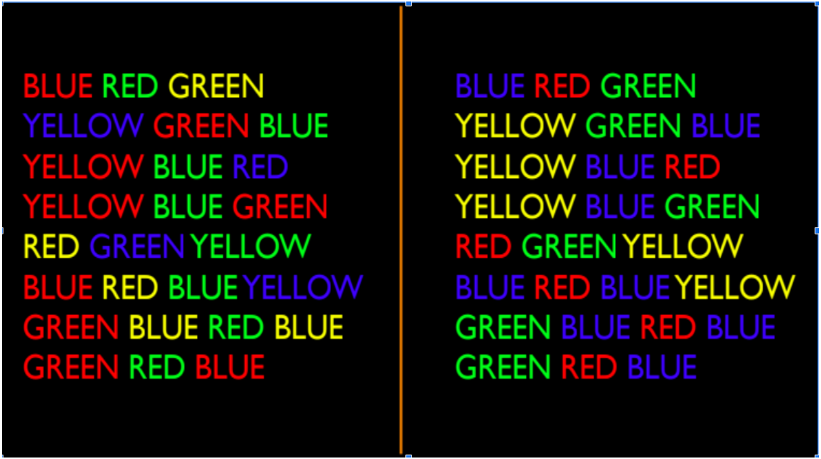
- Knowledge Factorization.
- Modularity and Dynamical Systems.
- Programming Languages and Generalization.

Beyond Performing a “Single” task



“Generalists”: ability to perform many tasks

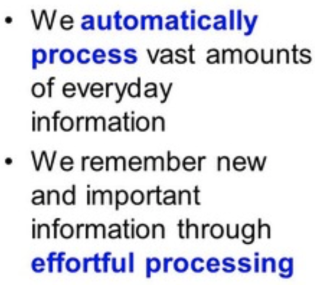
Stroop Effect



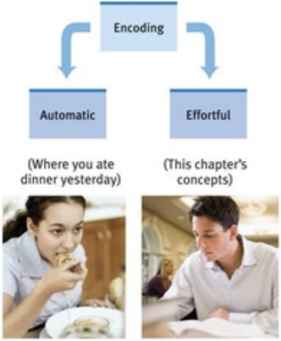
Automatic and Controlled Processing

Automatic Processing
- Fast and efficient
- Unavailable to consciousness
- Non Linguistic
- Habitual

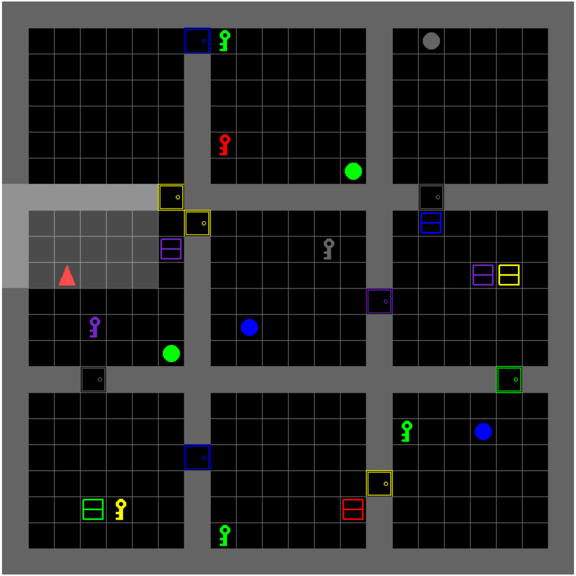
Controlled Processing
- Slow and less efficient
- Unavailable to consciousness
- Linguistic
- Algorithmic, planning, reasoning



- We **automatically process** vast amounts of everyday information
- We remember new and important information through **effortful processing**



Data Generating Distribution



Task distribution - Diverse, growing, compositional.

Missing from Current Dynamical Systems ?

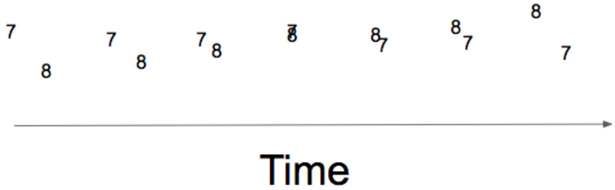
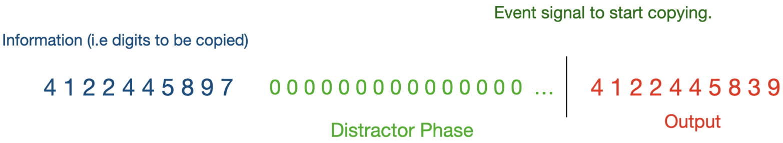


Figure 1

- Normal neural networks generally put all of these computational streams together.
- Hard for the model to share information in a dynamic way (maybe inclined towards either always sharing information or doing it in a fixed way).

Systematicity in Copying Task



Event signal to start copying.

Information (i.e digits to be copied)

4 1 2 2 4 4 5 8 9 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ... | 4 1 2 2 4 4 5 8 3 9

Distractor Phase

Output

Figure 2: Copying Example: Two seperate Phases, information phase and distractor phase.

Intuitive Physics: Modelling the motion of specific objects

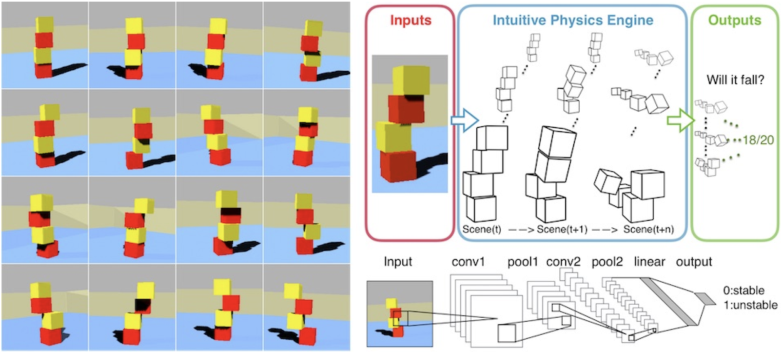
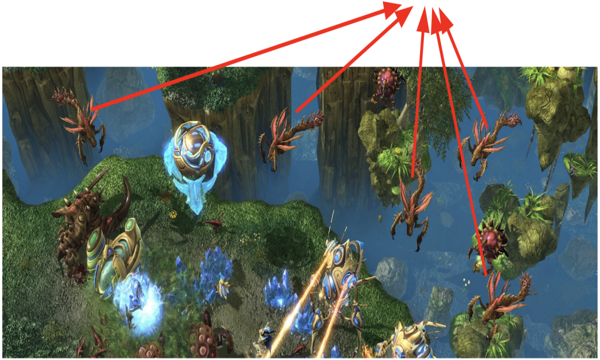


Figure 3: Intuitive Physics Engine and CNNs: Different objects have different dynamics, as well as subset of objects can share underlying dynamics.

Systematicity in Starcraft

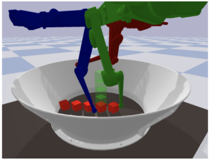


These “Vipers” have different state but share the same behavior.

Figure 4: Starcraft example

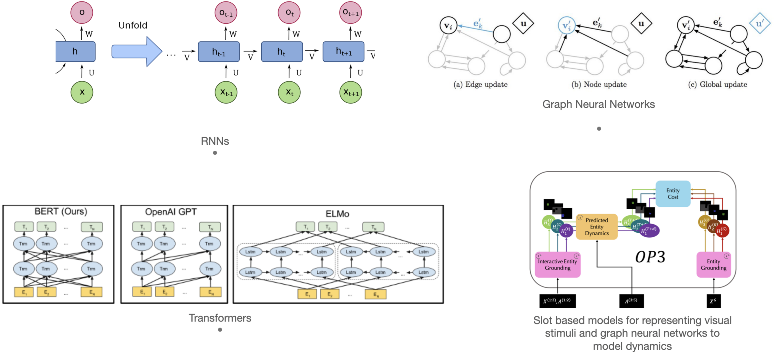
CausalWorld: Causal Structure Learning

- **Object based representations:** Given a set of available objects with different sizes, shapes… build a given target 3D shape
- **Affordances:** Natural environment to study object affordance aspects
- **Planning** How good is a robot at building these shapes?
- **Large space** of robotic manipulation environments that challenge state-of-the-art RL methods.
- **Meta-Transfer** How well can an agent transfer and generalise to other shapes, physical properties, ...?



(Trauble*, Ahmed*, Goyal, Wuthrich, Bengio, Scholkopf, Bauer, arxiv)

What has been done to solve these problems ?



RNNs

Graph Neural Networks

Transformers

Slot based models for representing visual stimuli and graph neural networks to model dynamics

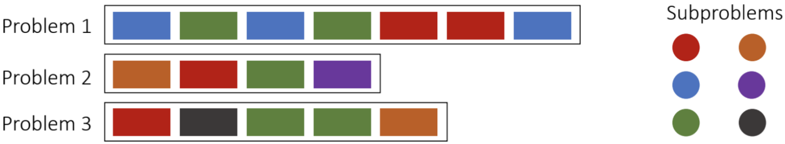
Figure 5: RNNs, Graph Neural Networks, Transformers, Slot Based Models

Need for Out of Distribution Generalization

Learning Agents face non-stationarities. Changes in distribution due to:

- Their actions.
- Actions of other agents.

Systematic Generalization/Compositional Generalization

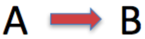


- Dynamically recombine existing concepts.
- Even when new combinations have 0 probability under training distribution.

Question: How to learn and then re-compose reusable computations ?

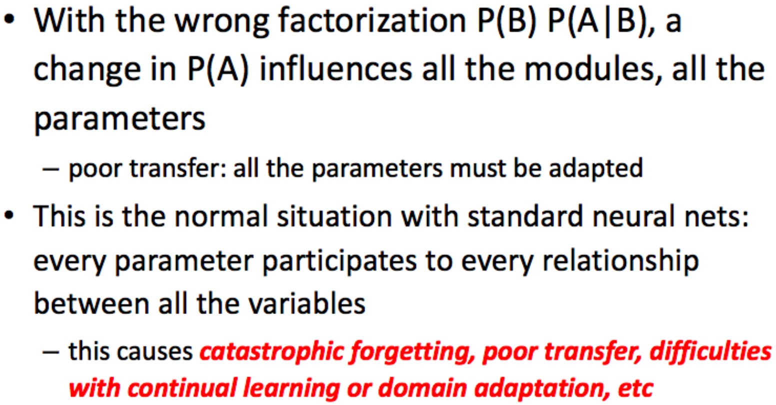
Incorrect Knowledge Factorization Leads to Poor Transfer

- Consider two r.v. A, B, with A cause of B.
- Correct causal model decomposes
    - $P(A,B) = P(A)\,P(B|A)$



- Consider 2 distributions P_1 and P_2, only $P(A)$ changes
- If we first train on P_1 and we have the right decomposition, adapting on P_2 is fast because

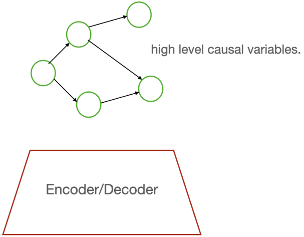
Incorrect Knowledge Factorization Leads to Poor Transfer



- With the wrong factorization P(B) P(A|B), a change in P(A) influences all the modules, all the parameters
  - poor transfer: all the parameters must be adapted
- This is the normal situation with standard neural nets: every parameter participates to every relationship between all the variables
  - this causes ***catastrophic forgetting, poor transfer, difficulties with continual learning or domain adaptation, etc***

High Level Variables are Causal Variables

- High level variables are also causal variables corresponding to actions, controllable objects, intentions etc.
- How is raw sensory data mapped to high level causal variables ?
- How to discover the relationship between these causal variables ?
- How are actions corresponding to causal interventions ?
- How do high level causal variables turn into low level actions and partial observations ?



Causality provides us a language to talk about changes in distribution.

Beyond iid: Independent Mechanisms and Small Change Hypothesis

- Independent Causal Mechanisms.
- Small Change Hypothesis: Non-stationarities, changes in distribution, involve few mechanisms (e.g. the result of a single-variable intervention).

How can we discover these mechanisms i.e factor knowledge ?

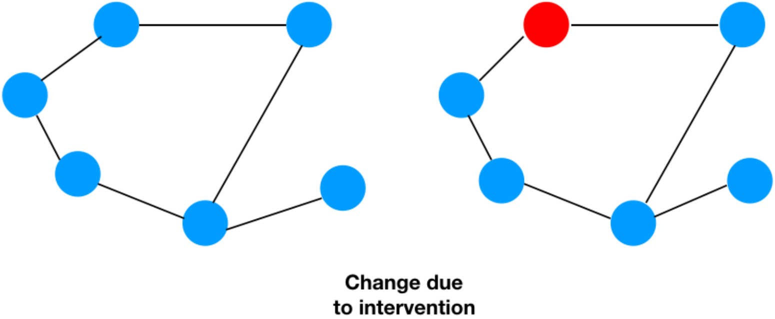
Independent Mechanisms

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{m} p(X_i \mid PA_i). \tag{1}$$

- Interventions are feasible if the mechanisms (i.e., causal conditions) are independent. Changing one conditional $p(X_i \mid PA_i)$ does not change other $p(X_j \mid PA_j)$ for $(i \neq j)$, they remain invariant.
- if all PA_i are empty, the factors are statistically independent. (disentanglement as independence).

Separating Knowledge in Small Re-Usable Pieces

- Mechanisms which can be used combinatorially.
- Mechanisms which are stable versus non-stationary subject to non-stationarity.



Learning Representations, where change can be localized.

Recap: Uptil now

- Need to deal with out of distribution changes.
- High Level variables are causal variables.
- Assumptions on the joint distribution of these high level variables.
- Localized changes upon interventions in the space of these high level variables.

Missing from Current Dynamical Systems ?

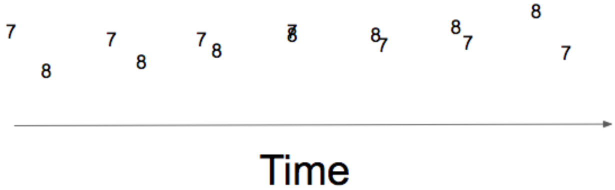


Figure 6

- Normal neural networks generally put all of these computational streams together.
- Hard for the model to share information in a dynamic way (maybe inclined towards either always sharing information or doing it in a fixed way).

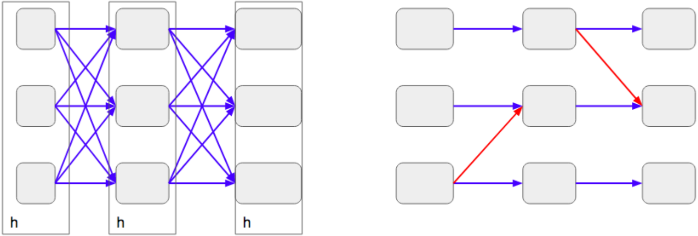
Why could it help to have separated dynamics?

- Reuse, Recompose, Repurpose.
- Learning independent mechanisms, buys invariance.
- Invariance buys extrapolation.
- Better transfer and continual learning (if a new task shares some dynamics but not others)
- Makes it easy for the model to keep two processes separate, which could also make long-term information storage much easier (no interference).

What form of knowledge representation would support these goals ?: What kind of assumptions on the joint distribution of high level variables ?

Ensemble of Sparingly Interacting Modules

Intuitive Figure



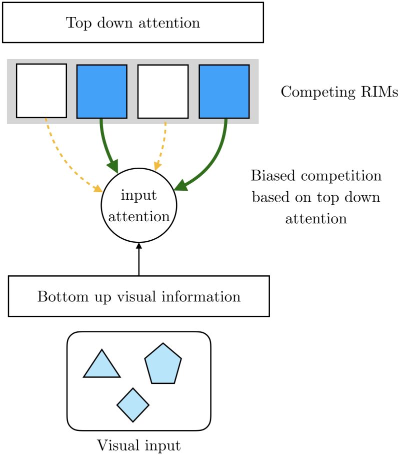
Normal RNN

RIMs

Desiderata for Learning Independent Mechanisms

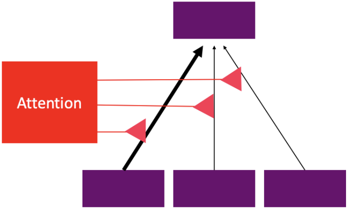
- **Seeking relevant information:** Each Module only attends to relevant information.
- **Capacity:** Limiting the capacity of each module.
- **Diversity:** Diversity among the different modules i.e. capture different aspects of the world.
- **Coherence between different mechanisms:** Need to think about “coherence” between different mechanisms.

Top Down Competition: Attentive Information Seeking


Top down attention
Competing RIMs
Biased competition based on top down attention
input attention
Bottom up visual information
Visual input

- The top-down filter not only enhances the target but also suppresses other stimuli, including the distractors.

From Attention To Indirection



- Attention = dynamic connection
- Receiver gets the selected value
- Value of what? From where?

  → Also send ‘name’ (or key) of sender
- Keep track of 'named’ objects: indirection
- Manipulate sets of objects (transformers)

Multiheaded Key Value Attention: Manipulate sets of objects

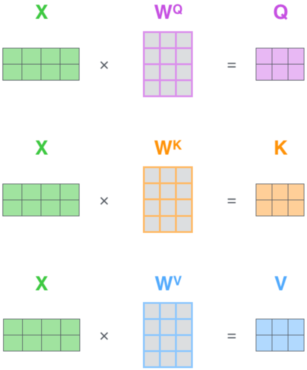


Figure 7: Linear Projection

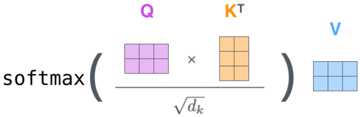
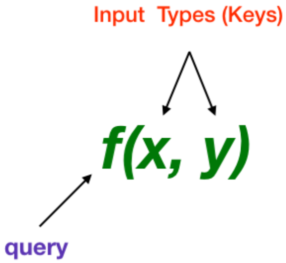


Figure 8: Softmax Calculation

Typed Argument Functions and Generalization



Input Types (Keys)

$f(x, y)$

query

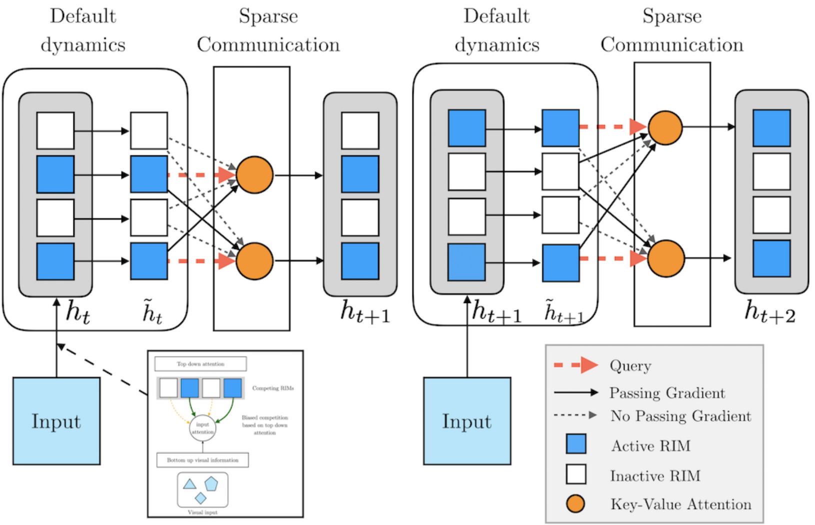
Binding: the mechanism would look at the input object's key and evaluate if its "type" matches with what this RIM expects

Efficient Credit Assignment: Each mechanism itself can decide what input it want to operate on.

Recurrent Independent Mechanisms

- Data Dependent Activation of Mechanisms
    - Top Down Activation of Mechanisms
- Coherence between representations of different mechanisms.
    - Active Mechanisms communicate with other mechanisms.
- Inactive Mechanisms follow the *default* dynamics.

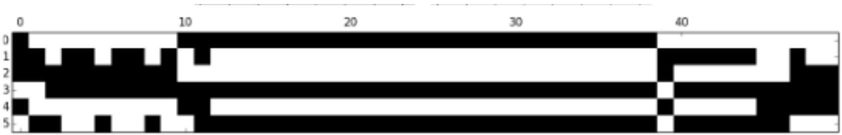
Recurrent Independent Mechanisms

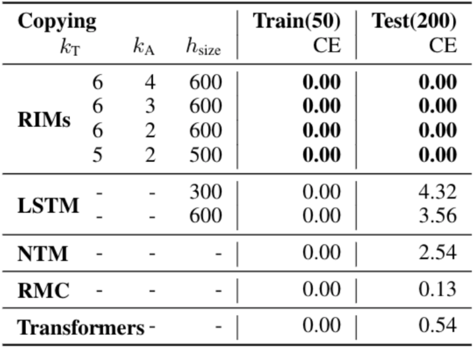


Experiments: Diagnostic Interventions

- **Robustness to Distractors:** Ignore Irrelevant Information.
- **Event Based Representations:** Specialization over temporal patterns.
- **Object Based Representations:** Specialize over objects and generalize over them.

Specialization over temporal patters: Copying Task





Copying	k_{T}	k_{A}	h_{size}	Train(50) CE	Test(200) CE
RIMs	6	4	600	**0.00**	**0.00**
	6	3	600	**0.00**	**0.00**
	6	2	600	**0.00**	**0.00**
	5	2	500	**0.00**	**0.00**
LSTM	-	-	300	0.00	4.32
	-	-	600	0.00	3.56
NTM	-	-	-	0.00	2.54
RMC	-	-	-	0.00	0.13
Transformers		-	-	0.00	0.54

Specialization over objects and generalize over them

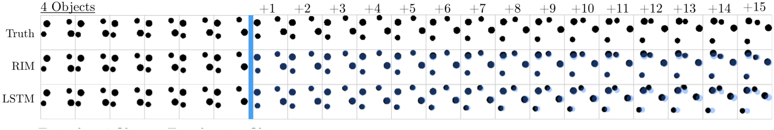


Figure 9: Predicting Movement of Bouncing Balls. The first 15 frames of ground truth are given (last 6 of those shown) and then the system is rolled out for the next 15 time steps. We find that RIMs perform better than the LSTMs (predictions are in black, ground truth in blue). Notice the blurring of LSTM predictions.

Specialize over objects and generalize over them

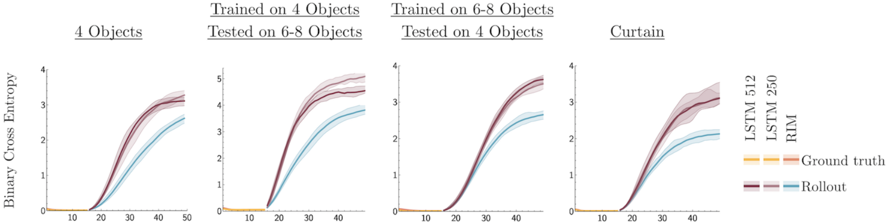


Figure 10: Handling Novel Out-of-Distribution Variations. Here, we study the performance of our proposed model compared to an LSTM baseline. The first 15 frames of ground truth are fed in and then the system is rolled out for the next 10 time steps. During the rollout phase, RIMs perform better than the LSTMs in accurately predicting the dynamics of the balls as reflected by the lower Cross Entropy (CE) [see blue for RIMs, purple for LSTM]. Notice the substantially better out-of-distribution generalization of RIMs when testing on a different number of objects than during training.

Better Robustness to Distractors

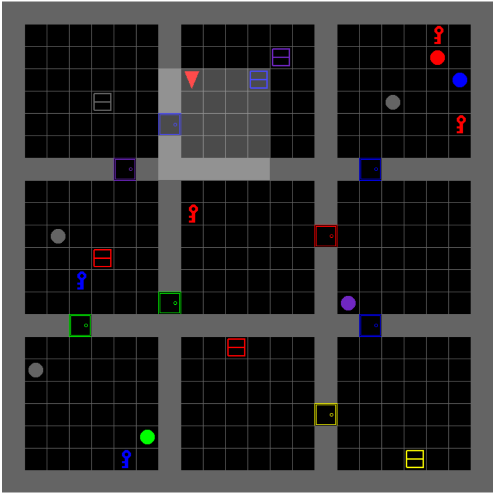


Figure 11: An example of the minigrid task.

Better Robustness to Distractors

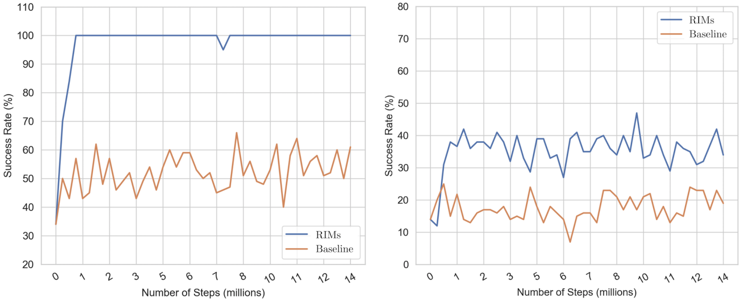


Figure 12: Robustness to Novel Distractors:. Left: performance of the proposed method compared to an LSTM baseline in solving the object picking task in the presence of distractors. Right: performance of proposed method and the baseline when novel distractors are added.

RIMs improve Generalization in Complex Atari Environments

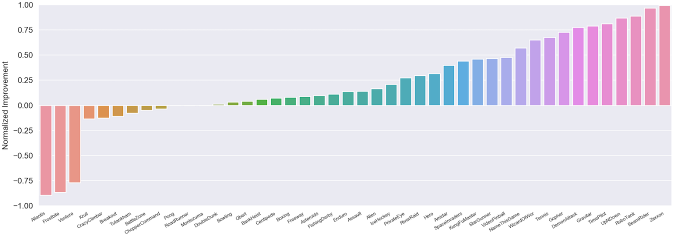
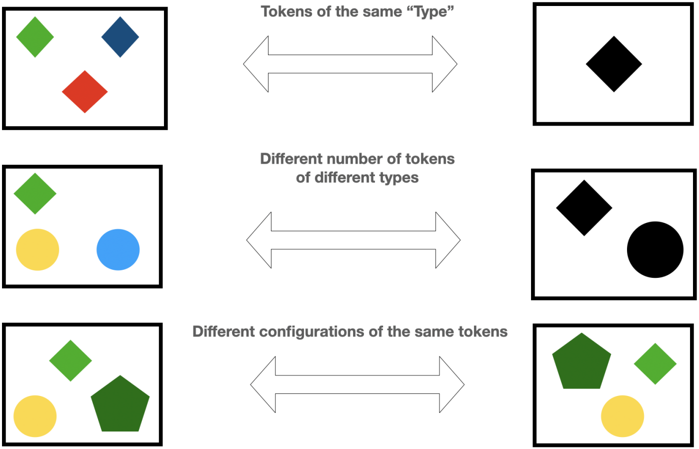


Figure 13: RIMs-PPO relative score improvement over LSTM-PPO baseline across all Atari games averaged over 3 trials per game. In both cases, PPO was used with the exact same settings, and the only change is the choice of recurrent architecture.

Summary: Useful Properties of RIMs

- Processing sets of elements rather than fixed-size vectors as 'states' of the computation.
- Computation is sparse and modular.
- Computation is dynamic rather than static.
- Inputs and outputs of these modules are sets of objects.
- Inputs and outputs are similar to variables in logic and arguments in programming, in that the attention-driven control flow selects which actual objects will be fed as input to activated modules.

Classes, Objects, and Invariances



Tokens of the same "Type"

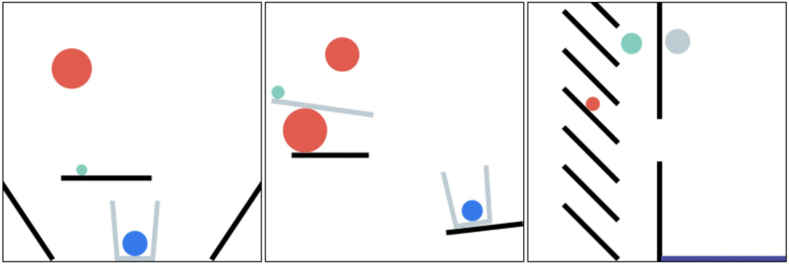
Different number of tokens of different types

Different configurations of the same tokens

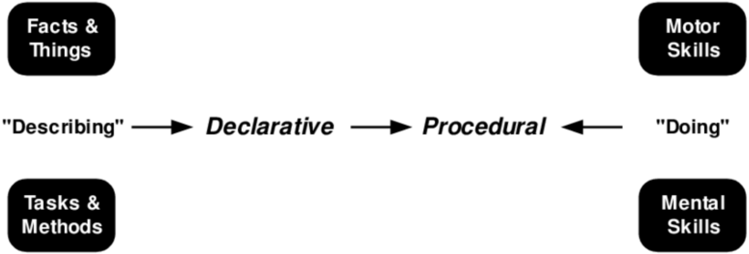
The properties of an entity are invariant to changes in the rest of the scene.

Essential Ingredients for Modelling Structure

- Separating Procedural (*schema*) and Declarative Knowledge (*Object files*).
  - Learning *object-centric* Representation
  - Learning *event-centric* Representation.
- Procedural Knowledge that can be instantiated on different variables.



Procedural and Declarative Knowledge



Facts & Things

Tasks & Methods

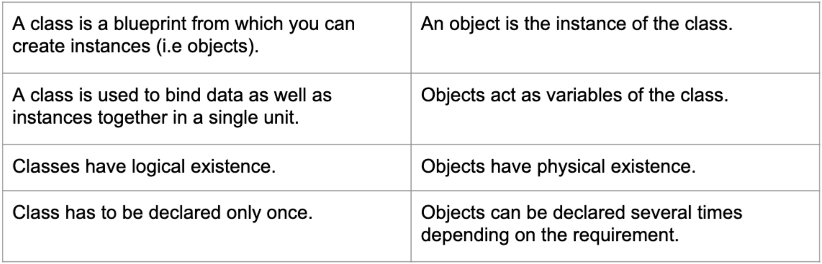
"Describing" → *Declarative* → *Procedural* ← "Doing"

Motor Skills

Mental Skills

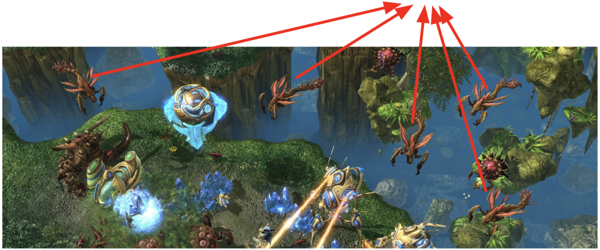
Generic Knowledge on Variables which can be Instantiated

- Classes vs. Objects



A class is a blueprint from which you can create instances (i.e objects).	An object is the instance of the class.
A class is used to bind data as well as instances together in a single unit.	Objects act as variables of the class.
Classes have logical existence.	Objects have physical existence.
Class has to be declared only once.	Objects can be declared several times depending on the requirement.

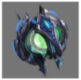
Intuitive Example of Classes vs. Objects



These “Vipers” have different state but share the same behavior.

Figure 14: Starcraft example

Procedural Knowledge in the form of Schemata

Object File	Schema #1	Schema #2	Schema #3 
A	X		
B	X		
C		X	
D	X		
E			X
F		X	
G			X

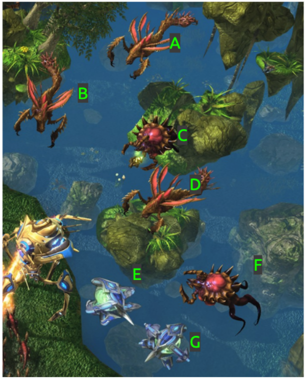
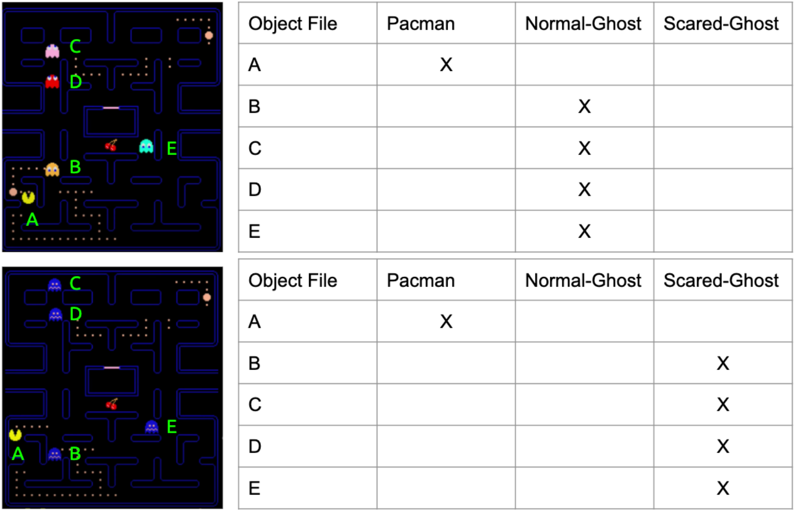


Figure 15: Decomposition of Knowledge into Object Files and Schemata

Procedural Knowledge in the form of Schemata



Object File	Pacman	Normal-Ghost	Scared-Ghost
A	X		
B		X	
C		X	
D		X	
E		X	

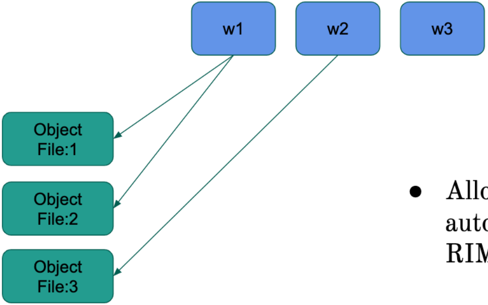
Object File	Pacman	Normal-Ghost	Scared-Ghost
A	X		
B			X
C			X
D			X
E			X

Figure 16: Decomposition of Knowledge into Object Files and Schemata

Proposal to add the "class/object" concept to RIMs

- We can see each RIM as being roughly analogous to a singleton object since each always has its own unique method and states.
- However, there is no clear way for us to give the same methods to multiple RIMs even though they have their own states.
- Idea is that we can decouple these by having a smaller set of parameters which we can learn to flexibly reuse across multiple RIMs.

Object Files and Schemata's



- Allow the model to automatically give multiple RIMs the same weights.

Figure 17: Decomposition of Knowledge into Object Files and Schemata

One Object File and Many Schemata's

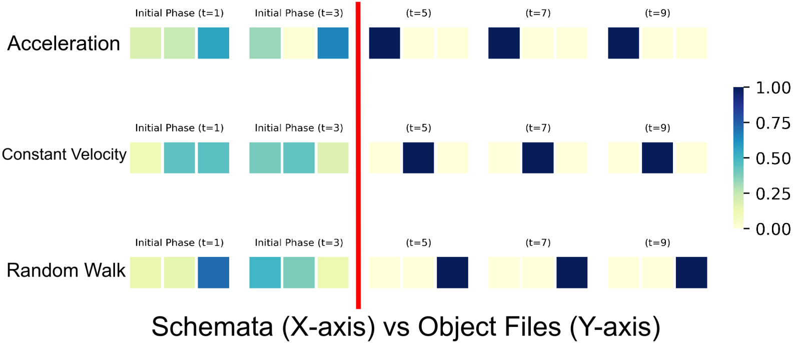
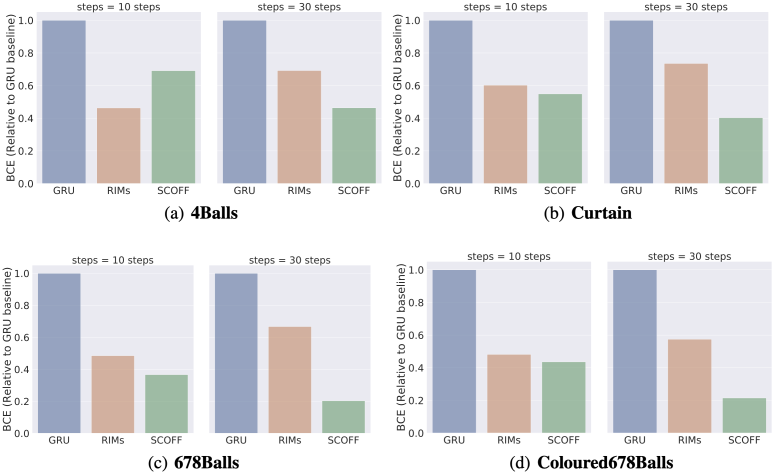


Figure 18: Here, we have a single object file, and that can follow three different dynamics. We found that our method is able to learn these 3 different modes once it's passed an initial phase of uncertainty.

Many Object Files and Many Schemata’s



(a) **4Balls**

(b) **Curtain**

(c) **678Balls**

(d) **Coloured678Balls**

Figure 19: Bouncing ball motion: Prediction error comparison of scoff, LSTM, and RIMs. Given 10 frames of ground truth, the model predicts the rollout over the next 35 steps. scoff performs better than LSTM and RIMs.

Conclusions

- High level variables are causal.
- Distributional Changes due to localized causal interventions.
- Division into Procedural and Declarative Knowledge.
    - Procedural Knowledge can be called upon different instances.
    - Connections to indirection in Attention.
- Sparse factor graph in the space of high level variables.

Questions ?