Modelling Generalised Symmetry in Neural Networks

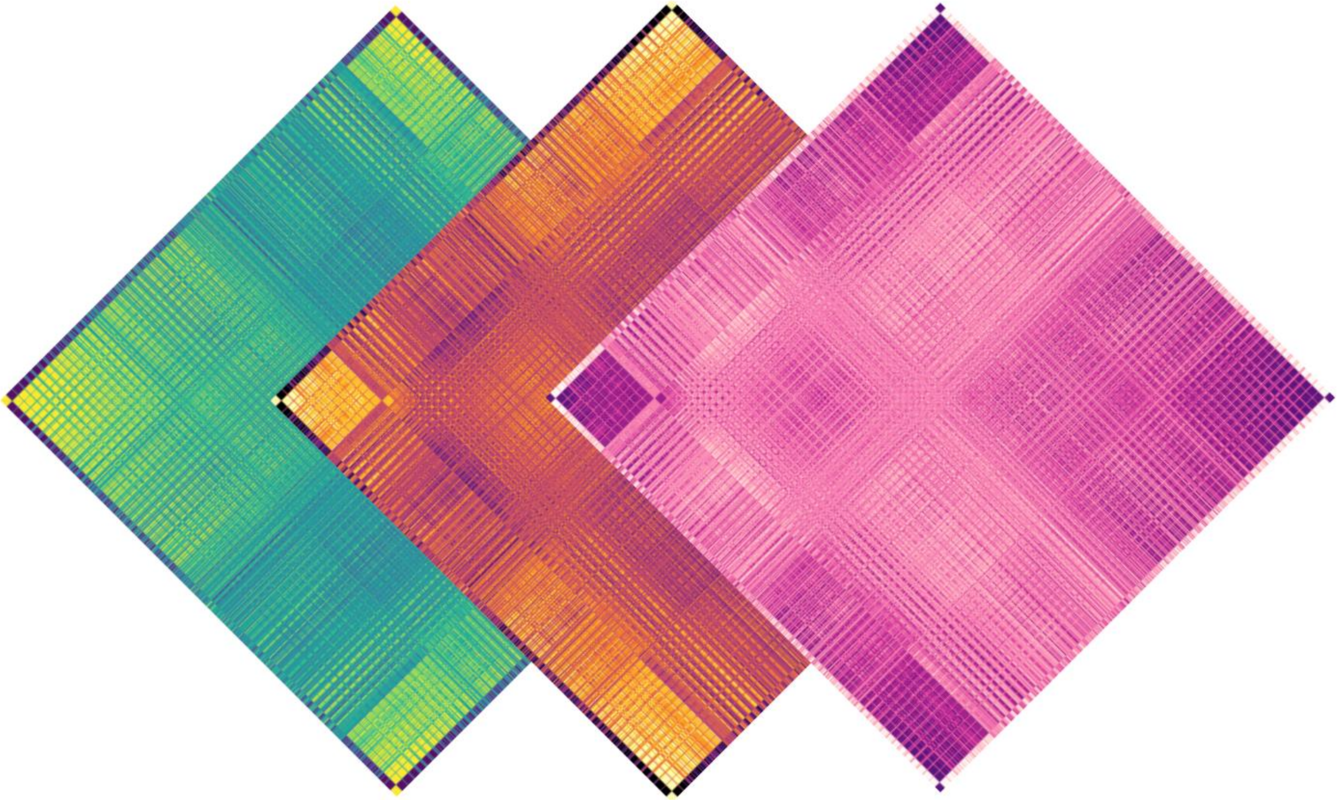
Nathan Duran

UWE Bristol University of the West of England

C S R C

Overview

- Symbolic vs Computational AI
- Feed Forward Neural Networks
  - Binary
  - Multi-class
- Hopfield Networks
- Graph Neural Networks



Symbolic VS Computational AI

Symbolic AI (GOFAI)

Inspired by ideas of the mind and building on thousands of years of philosophy about:

- World models, absolute truths
- Types of logic, facts and rules

Reasoning with *symbols* that represent entities.

If class(**X**) == class(**A1**) => **B1**

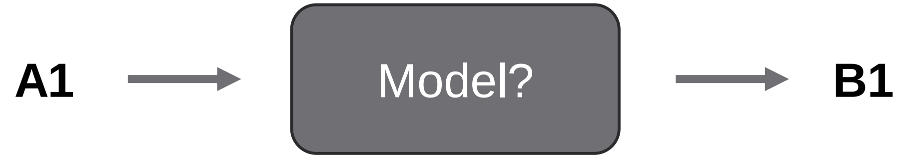
Inflexible – *how to create a rule that is general to any arbitrary stimuli?*

Computational AI (ML)

Inspired by ideas of problem solving arising from natural computational processes:

- The brain
- Darwinian evolution and genetics

Reasoning with *numbers* that represent entities.



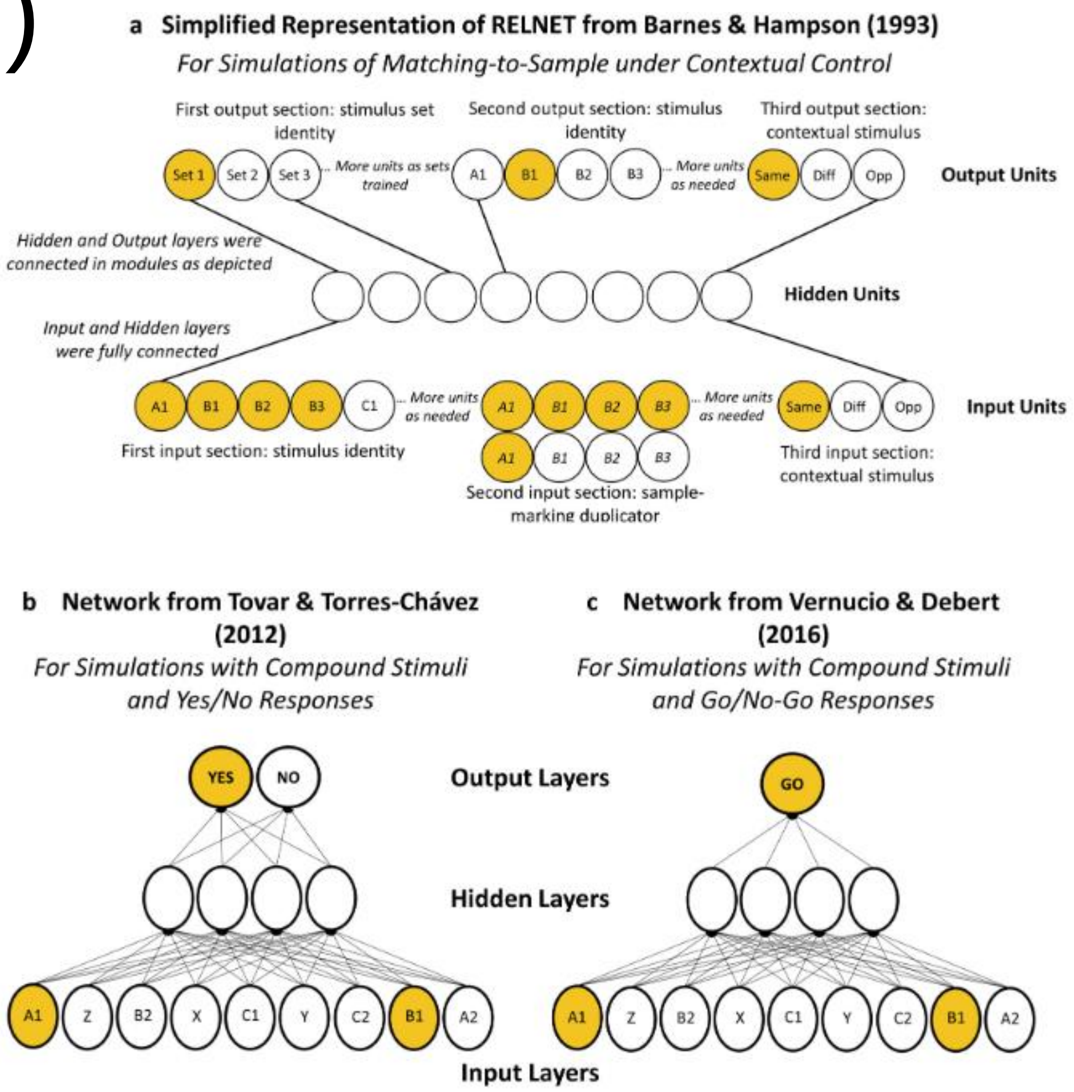
Data driven – *how to create a model if meaning is derived from arbitrary (data) stimuli?*

Feed Forward NN (MLP)

RELNET - Barns and Hampson (1993) and several others used this method.

Tovar and Torres-Chavez (2012) pointed out a flaw in its design. The input pattern for the 'sample marking duplicator' is identical for all input stimulus sets for a given task.

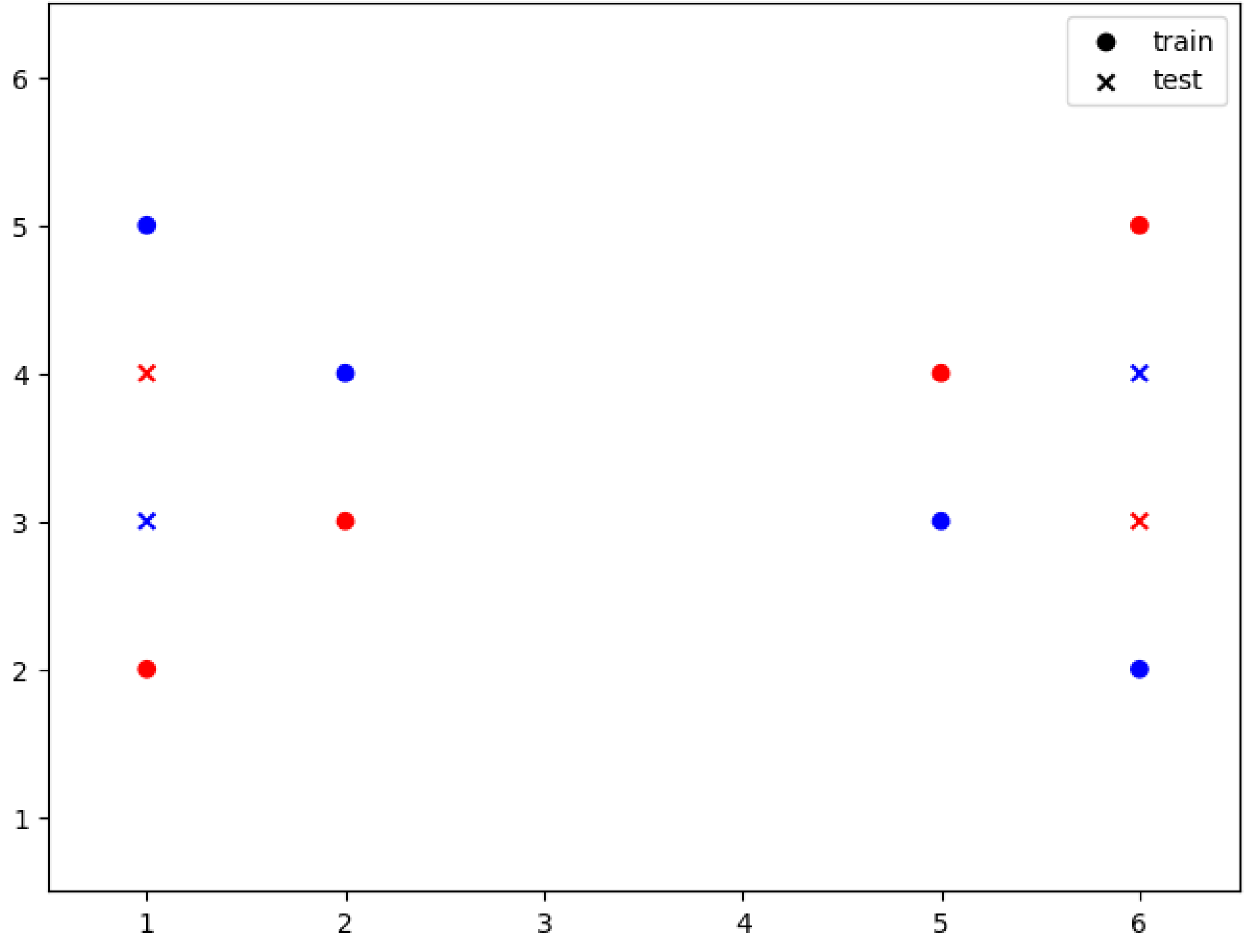
Binary, Go/No Go and EVA[3] - Valid, but different formulation of the problem which drastically simplifies it from multi-class to binary yes/no outputs.



(1) Barnes, D. and Hampson, P.J. (1993) Stimulus Equivalence and Connectionism: Implications for Behavior Analysis and Cognitive Science.
(2) Tovar, A.E. and Chávez, A.T. (2012) A Connectionist Model of Stimulus Class Formation with a Yes/No Procedure and Compound Stimuli.
(3) Ninness, C., et al. (2018) The Emergence of Stimulus Relations: Human and Computer Learning.

Binary Problem - Replicating EVA

Set	Pair	a1 (1)	b2 (2)	c1 (3)	c2 (4)	b1 (5)	a2 (6)	labels
Train	a1-b1	1	0	0	0	1	0	1
Train	a1-b2	1	1	0	0	0	0	0
Train	b1-c1	0	0	1	0	1	0	1
Train	b1-c2	0	0	0	1	1	0	0
Train	a2-b2	0	1	0	0	0	1	1
Train	a2-b1	0	0	0	0	1	1	0
Train	b2-c2	0	1	0	1	0	0	1
Train	b2-c1	0	1	1	0	0	0	0
Test	a1-c1	1	0	1	0	0	0	1
Test	a1-c2	1	0	0	1	0	0	0
Test	a2-c2	0	0	0	1	0	1	1
Test	a2-c1	0	0	1	0	0	1	0



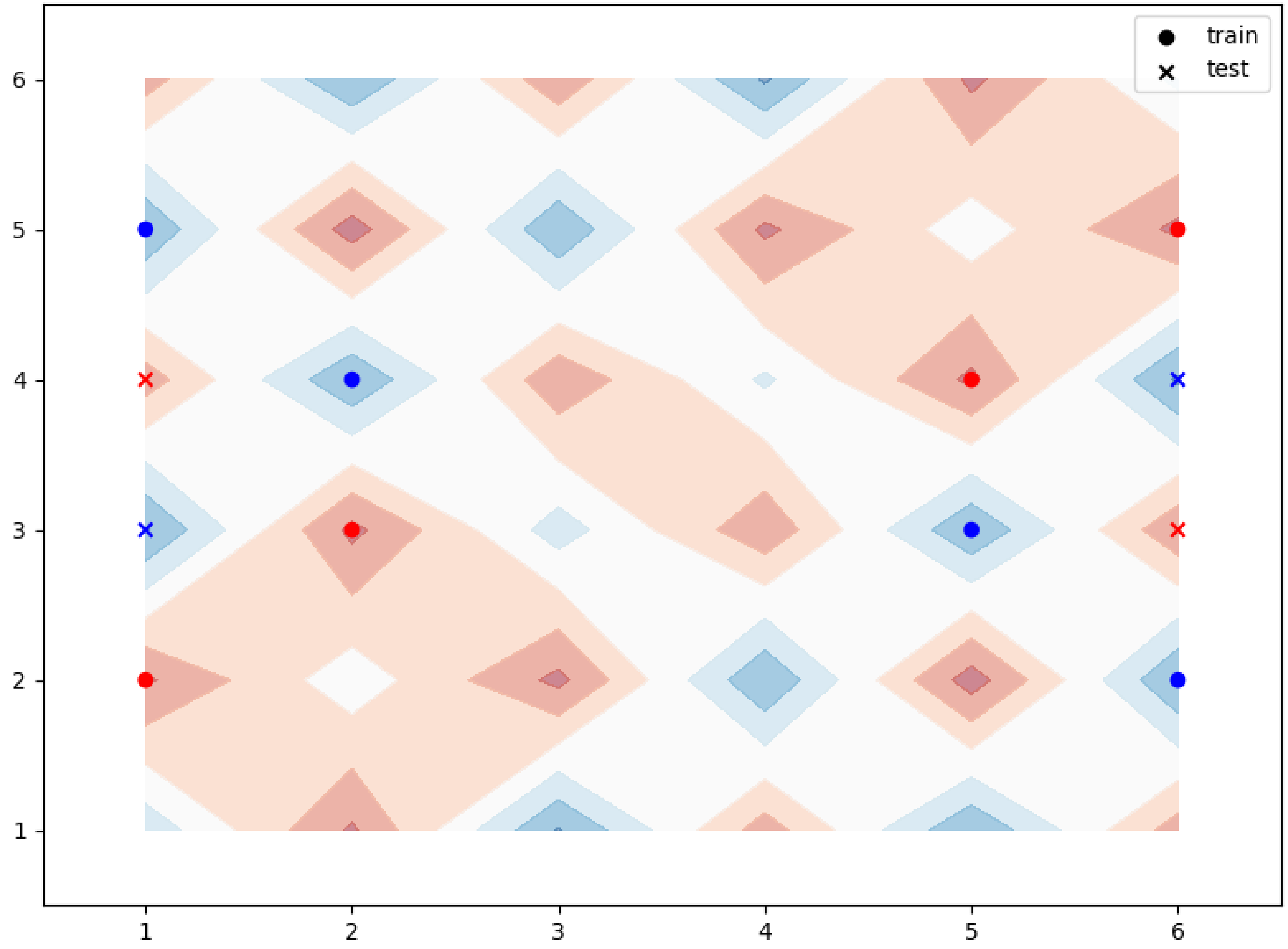
(1) Tovar, A.E. and Chávez, A.T. (2012) A Connectionist Model of Stimulus Class Formation with a Yes/No Procedure and Compound Stimuli.
(2) Ninness, C., et al. (2018) The Emergence of Stimulus Relations: Human and Computer Learning.

Binary Problem - Replicating EVA

Reaches 100% accuracy.

But not always!

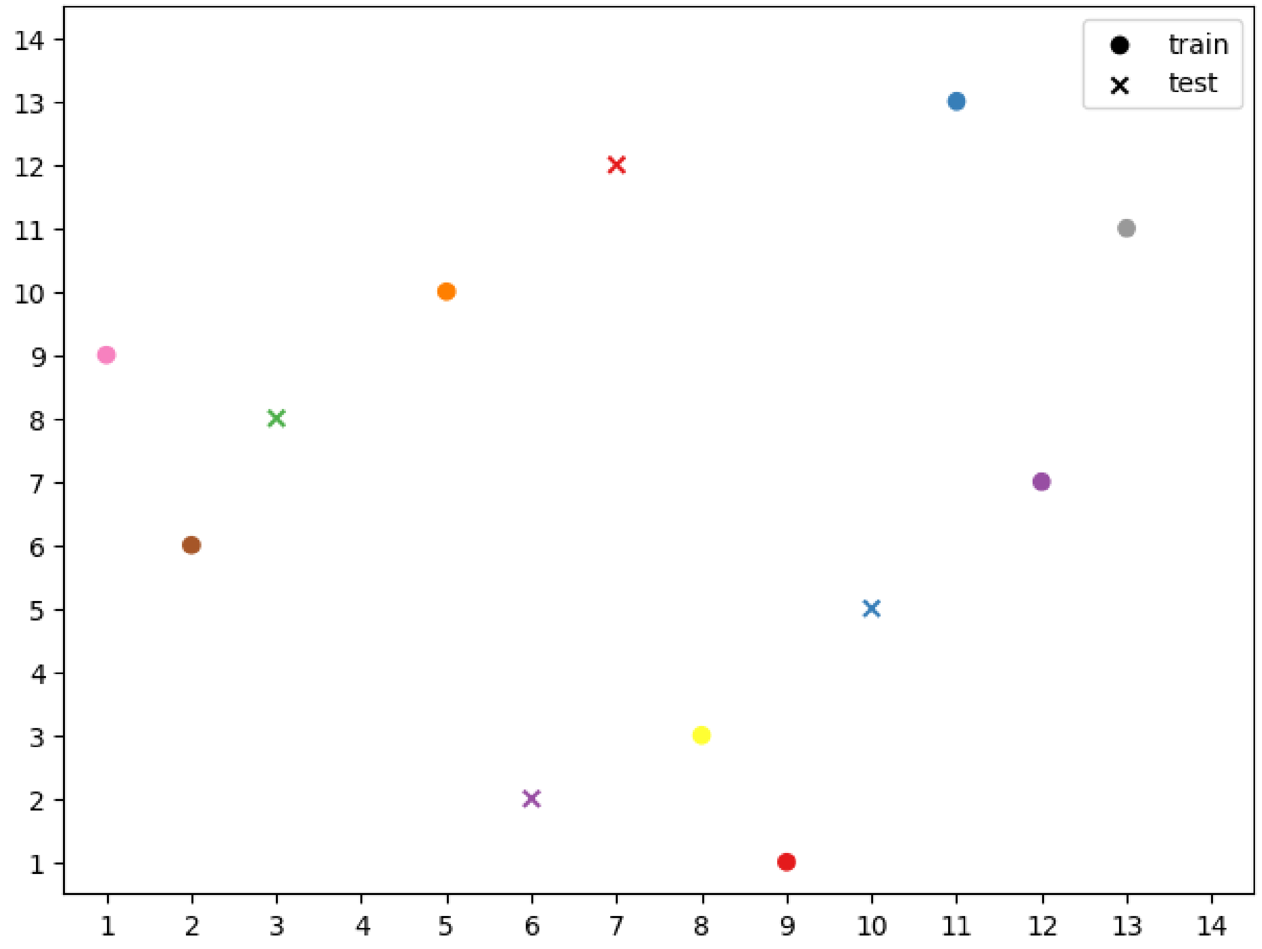
Changing the ‘pattern’ can make it easier/harder.



(1) Tovar, A.E. and Chávez, A.T. (2012) A Connectionist Model of Stimulus Class Formation with a Yes/No Procedure and Compound Stimuli.
(2) Ninness, C., et al. (2018) The Emergence of Stimulus Relations: Human and Computer Learning.

Multi-class Problem

Set	Pair	0	1	2	3	4	5	6	7	8	9	10	11	12	13	labels
Train	9 --> 1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1
Train	11 --> 13	0	0	0	0	0	0	0	0	0	0	0	1	0	1	13
Train	0 --> 4	1	0	0	0	1	0	0	0	0	0	0	0	0	0	4
Train	12 --> 7	0	0	0	0	0	0	0	1	0	0	0	0	1	0	7
Train	5 --> 10	0	0	0	0	0	1	0	0	0	0	1	0	0	0	10
Train	8 --> 3	0	0	0	1	0	0	0	0	1	0	0	0	0	0	3
Train	2 --> 6	0	0	1	0	0	0	1	0	0	0	0	0	0	0	6
Train	1 --> 9	0	1	0	0	0	0	0	0	0	1	0	0	0	0	9
Train	13 --> 11	0	0	0	0	0	0	0	0	0	0	0	1	0	1	11
Train	4 --> 0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Test	7 --> 12	0	0	0	0	0	0	0	1	0	0	0	0	0	0	12
Test	10 --> 5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	5
Test	3 --> 8	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8
Test	6 --> 2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2

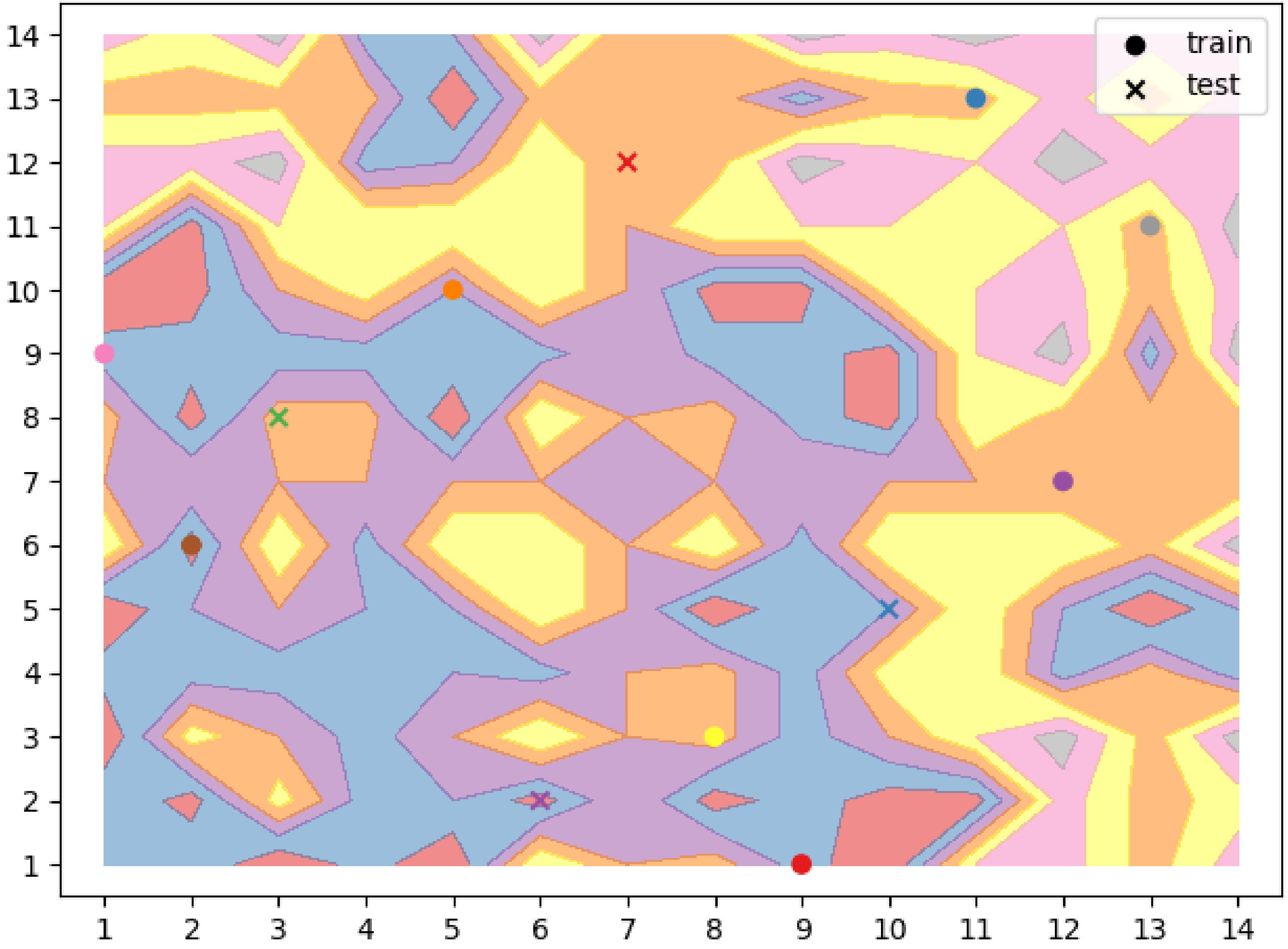


Multi-class Problem

Never reaches 100% accuracy.

Some training data (~70%) is ‘*memorised*’.

There is no pattern to apply to test data.



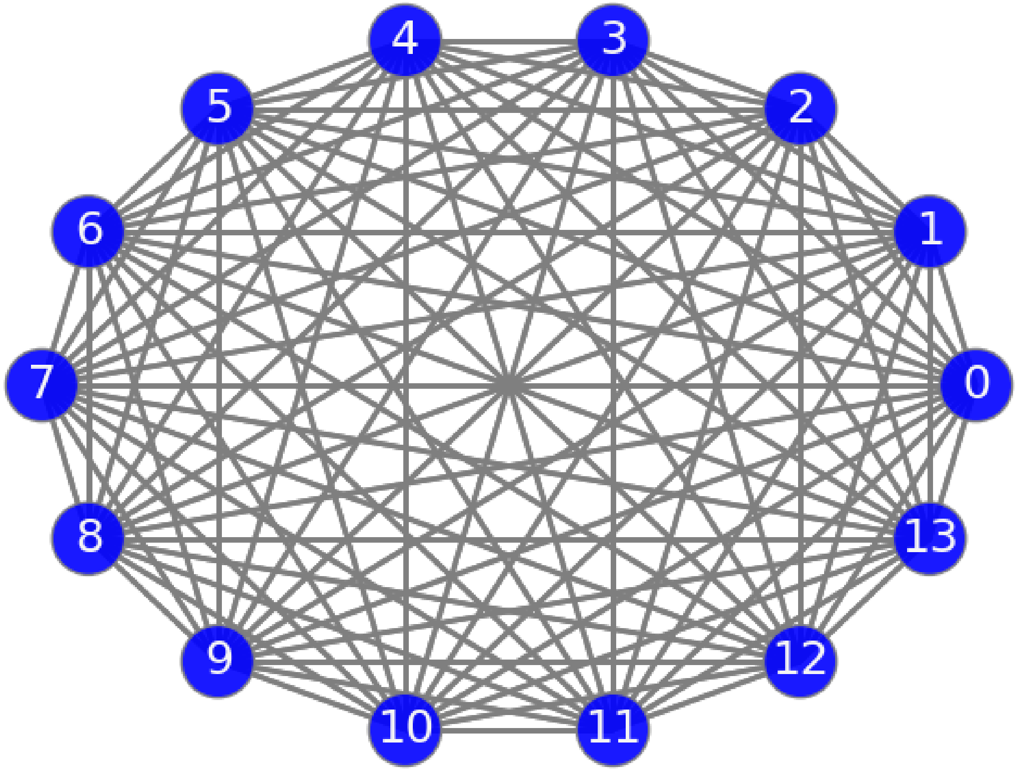
Hopfield Networks

Hopfield NN[1] are fully connected networks that allow for the retrieval and completion of a ‘memory’ using an incomplete or noisy version.

Each neuron (or node) is connected to all other neurons with a unique strength (weight).

The *information*, or *memories* of a Hopfield are stored in the strength of these connections.

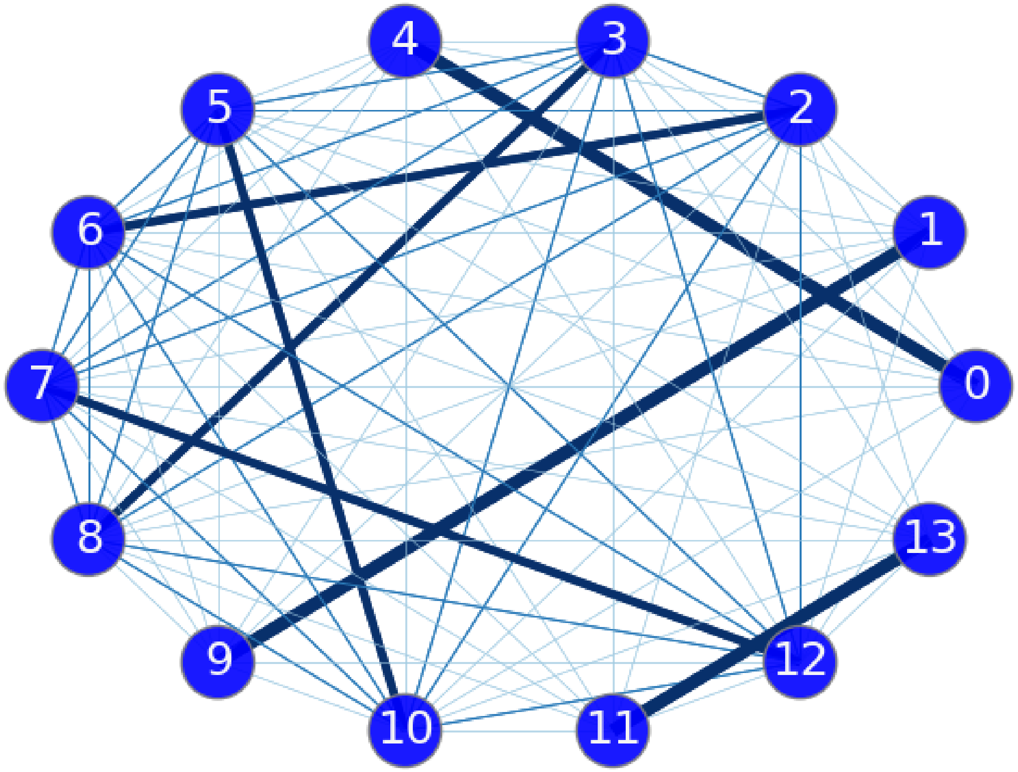
The weight between 2 neurons, A and B, is the extent to which the output of A will contribute to the activation of B, and vice versa.



(1) Hopfield, J.J. (1982) Neural networks and physical systems with emergent collective computational abilities

Hopfield Networks

Set	Pair	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Train	9 --> 1	-1	1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
Train	11 --> 13	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	1
Train	0 --> 4	1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Train	12 --> 7	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	1	-1
Train	5 --> 10	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	1	-1	-1	-1
Train	8 --> 3	-1	-1	-1	1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
Train	2 --> 6	-1	-1	1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1
Train	1 --> 9	-1	1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
Train	13 --> 11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	1
Train	4 --> 0	1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Test	7 --> 12	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1
Test	10 --> 5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1
Test	3 --> 8	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Test	6 --> 2	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1



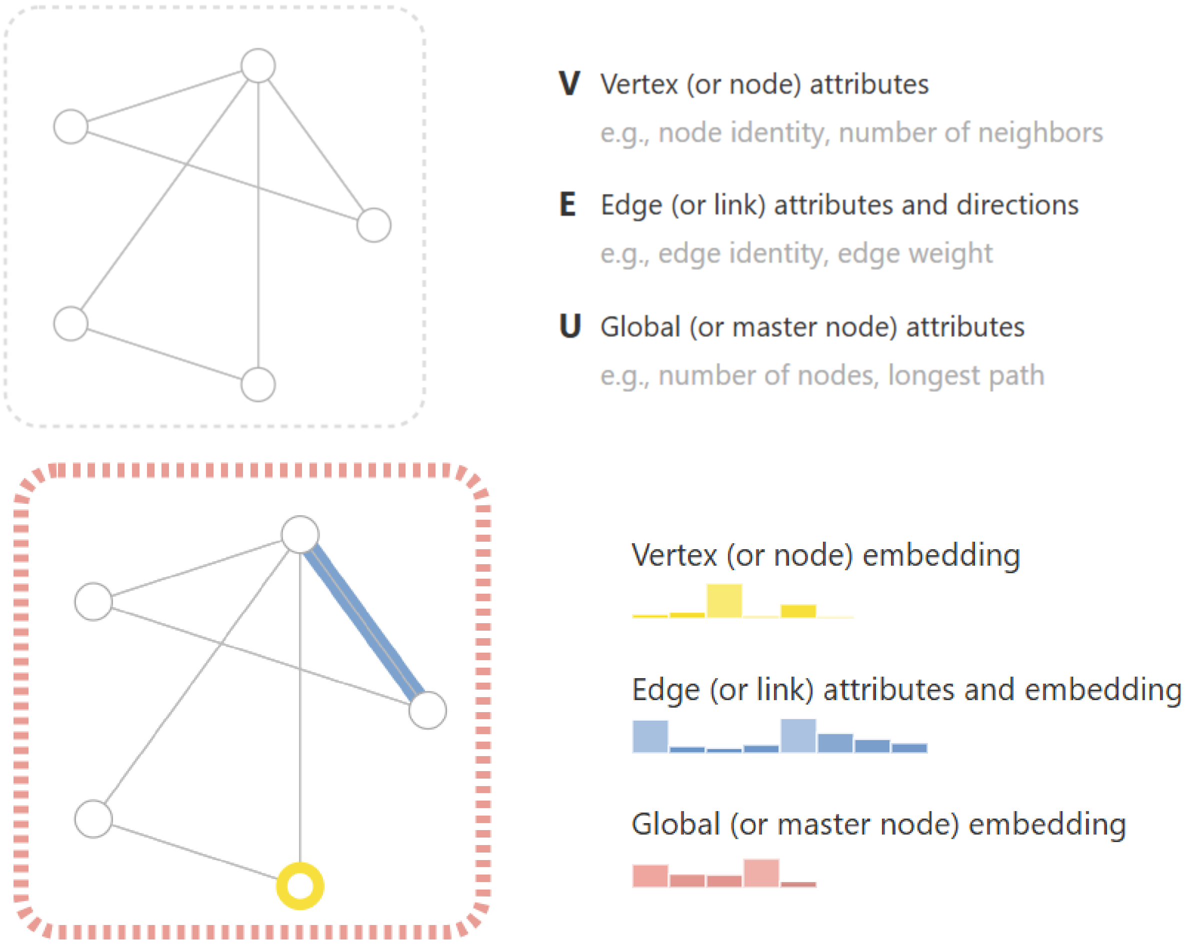
Graph Neural Networks

Graph Neural Networks (GNN)[1] can operate on graph-structured data and are well suited to *relational* learning.

Typical tasks for GNNS:
- Graph – property
- Node – type/class/property
- Edge – relation type/presence

Graph properties are encoded into embeddings (numerical representations) and used to infer missing properties.

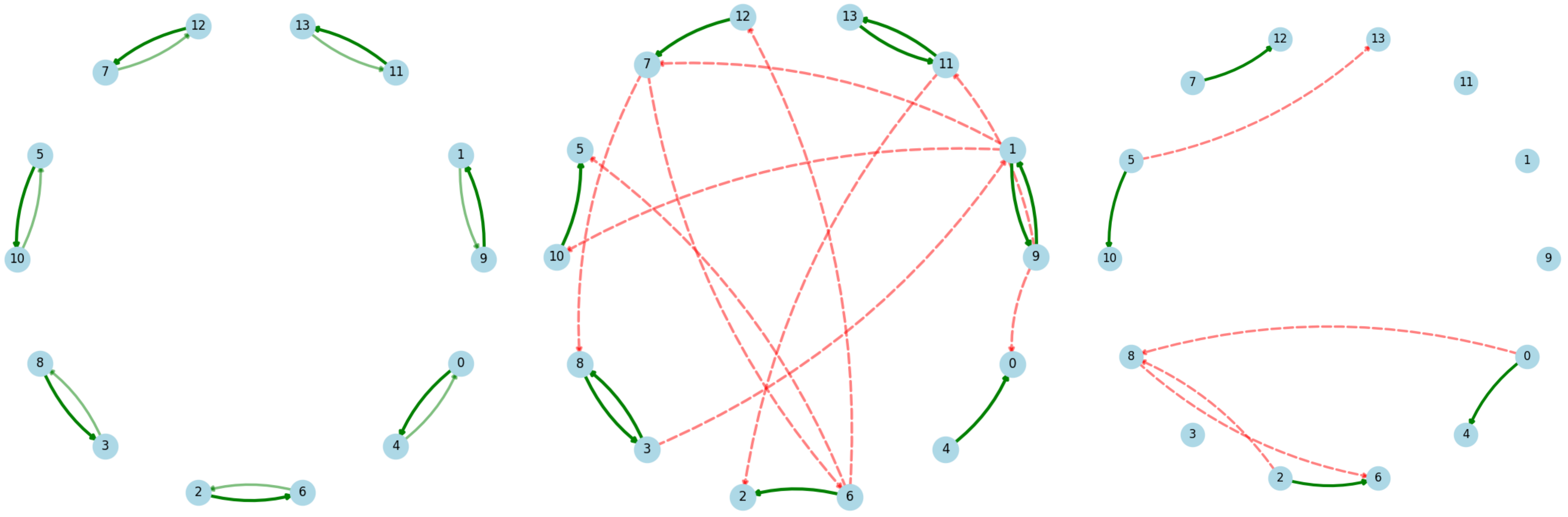
Can consider the problem of relational symmetry as nodes, representing stimuli, and edges, the relations between them.



(1) Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M. and Monfardini, G. (2009) The Graph Neural Network Model. *IEEE Transactions on Neural Networks*

Graph Neural Networks

Nodes (stimuli) are encoded as one-hot, e.g. Node ‘3’ = [0, 0, 0, 1, 0, ….]



Data **Train** **Test**

(1) Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M. and Monfardini, G. (2009) The Graph Neural Network Model. *IEEE Transactions on Neural Networks*

Graph Neural Networks

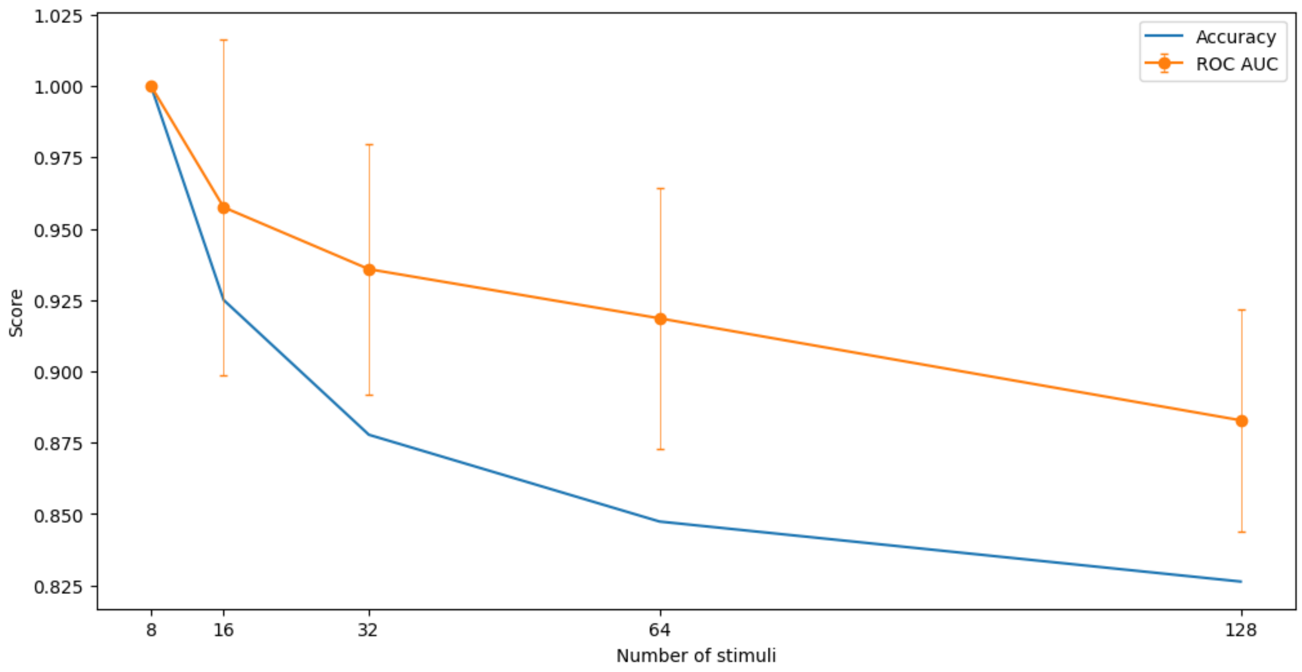
Some results for the link predictions task using the using the **GraphSAGE** (SAmple and aggreGatE) architecture.

14 Stimuli – Same split:
- `Acc = 95% ROC = 0.975`

14 Stimuli – Different splits*:
- `Acc = 95% ROC = 1.0`

Increasing number of stimuli:

Range 8 to 128 - doubling has relatively small impact on performance.



*Currently data split is random, which can lead to ‘unfair’ training/test sets.

(1) Hamilton, W.L., Ying, R. and Leskovec, J. (2017)“Inductive Representation Learning on Large Graphs.”

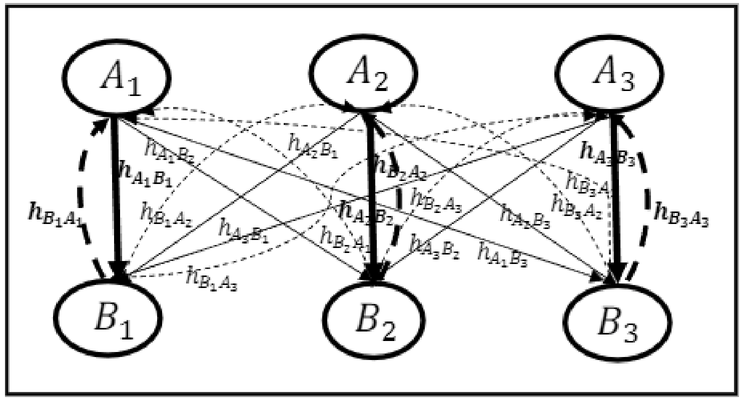
GNN Final Thoughts

GNNs have potential to overcome some of the limitations in other NN architectures:

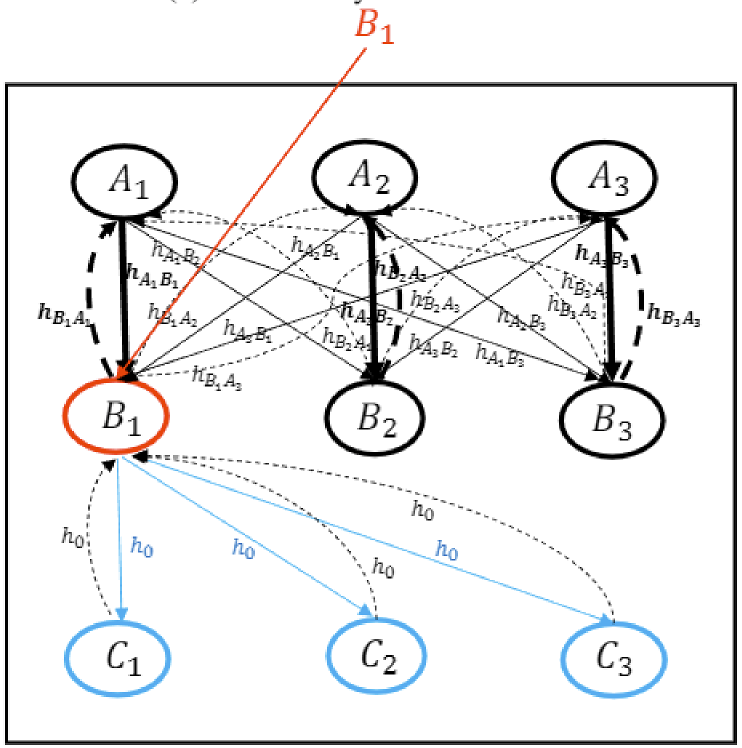
- Representing more complex stimuli or relations (e.g. transitive).
- Infers the relationship *between* stimuli, instead of the data examples themselves (inductive vs transductive).
- Could be included in a more 'dynamic' system e.g Equivalence Projective Simulation (EPS)[1].

BUT..
Link prediction is not the full solution – only predicts if there is a relation between pairs of stimuli.

Most likely a 'hybrid' combination of link and/or node classification is also required.



(a) The mastery of AB relation.



(1) Mofrad, A.A., Yazidi, A., Hammer, H.L. and Arntzen, E. (2020) Equivalence Projective Simulation as a Framework for Modeling Formation of Stimulus Equivalence Classes.