# Relational inductive biases, deep learning, and graph networks

**Peter W. Battaglia**[1]*, **Jessica B. Hamrick**[1], **Victor Bapst**[1], **Alvaro Sanchez-Gonzalez**[1], **Vinicius Zambaldi**[1], **Mateusz Malinowski**[1], **Andrea Tacchetti**[1], **David Raposo**[1], **Adam Santoro**[1], **Ryan Faulkner**[1], **Caglar Gulcehre**[1], **Francis Song**[1], **Andrew Ballard**[1], **Justin Gilmer**[2], **George Dahl**[2], **Ashish Vaswani**[2], **Kelsey Allen**[3], **Charles Nash**[4], **Victoria Langston**[1], **Chris Dyer**[1], **Nicolas Heess**[1], **Daan Wierstra**[1], **Pushmeet Kohli**[1], **Matt Botvinick**[1], **Oriol Vinyals**[1], **Yujia Li**[1], **Razvan Pascanu**[1]

[1]DeepMind; [2]Google Brain; [3]MIT; [4]University of Edinburgh

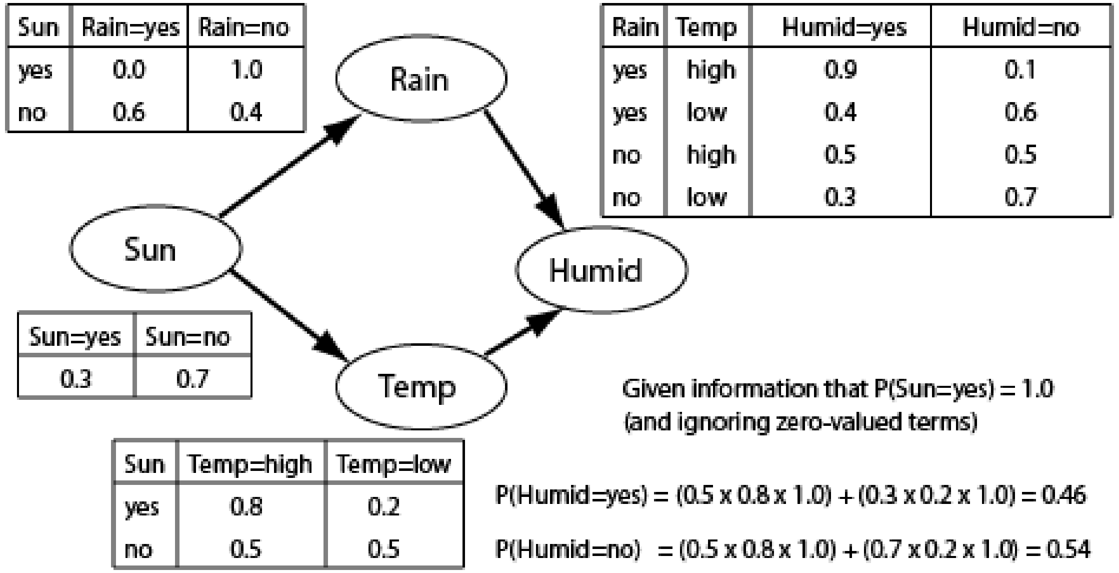
- Combinatorial generalization
- Relational inductive biases
- Graph network
- Discussion & Conclusion

# Combinatorial Generalization

- A key signature of human intelligence is the ability to make infinite use of finite means"
- Constructing new inferences, predictions, and behaviours from known building blocks.

# Modern deep learning methods

- “end-to-end” design philosophy
  - Emphasizes minimal a-priori representational and computational assumptions
  - Avoid explicit structure and “hand-engineering”
  - Cheap data, trade of sample efficiency



| Sun | Rain=yes | Rain=no |
|---|---|---|
| yes | 0.0 | 1.0 |
| no | 0.6 | 0.4 |

| Rain | Temp | Humid=yes | Humid=no |
|---|---|---|---|
| yes | high | 0.9 | 0.1 |
| yes | low | 0.4 | 0.6 |
| no | high | 0.5 | 0.5 |
| no | low | 0.3 | 0.7 |

| Sun=yes | Sun=no |
|---|---|
| 0.3 | 0.7 |

| Sun | Temp=high | Temp=low |
|---|---|---|
| yes | 0.8 | 0.2 |
| no | 0.5 | 0.5 |

Given information that P(Sun=yes) = 1.0
(and ignoring zero-valued terms)

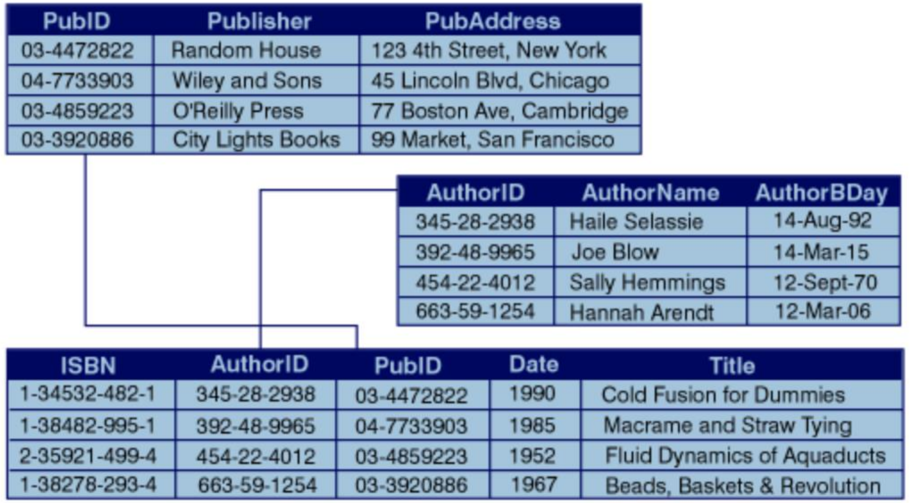
P(Humid=yes) = (0.5 x 0.8 x 1.0) + (0.3 x 0.2 x 1.0) = 0.46

P(Humid=no) = (0.5 x 0.8 x 1.0) + (0.7 x 0.2 x 1.0) = 0.54

# Modern deep learning methods(cont`d)

- Challenges:
  - complex language and scene understanding
  - reasoning about structured data
  - transferring learning beyond the training conditions
  - Learning from small amounts of experience

Uses **nature** and **nurture** jointly



| PubID | Publisher | PubAddress |
|---|---|---|
| 03-4472822 | Random House | 123 4th Street, New York |
| 04-7733903 | Wiley and Sons | 45 Lincoln Blvd, Chicago |
| 03-4859223 | O'Reilly Press | 77 Boston Ave, Cambridge |
| 03-3920886 | City Lights Books | 99 Market, San Francisco |

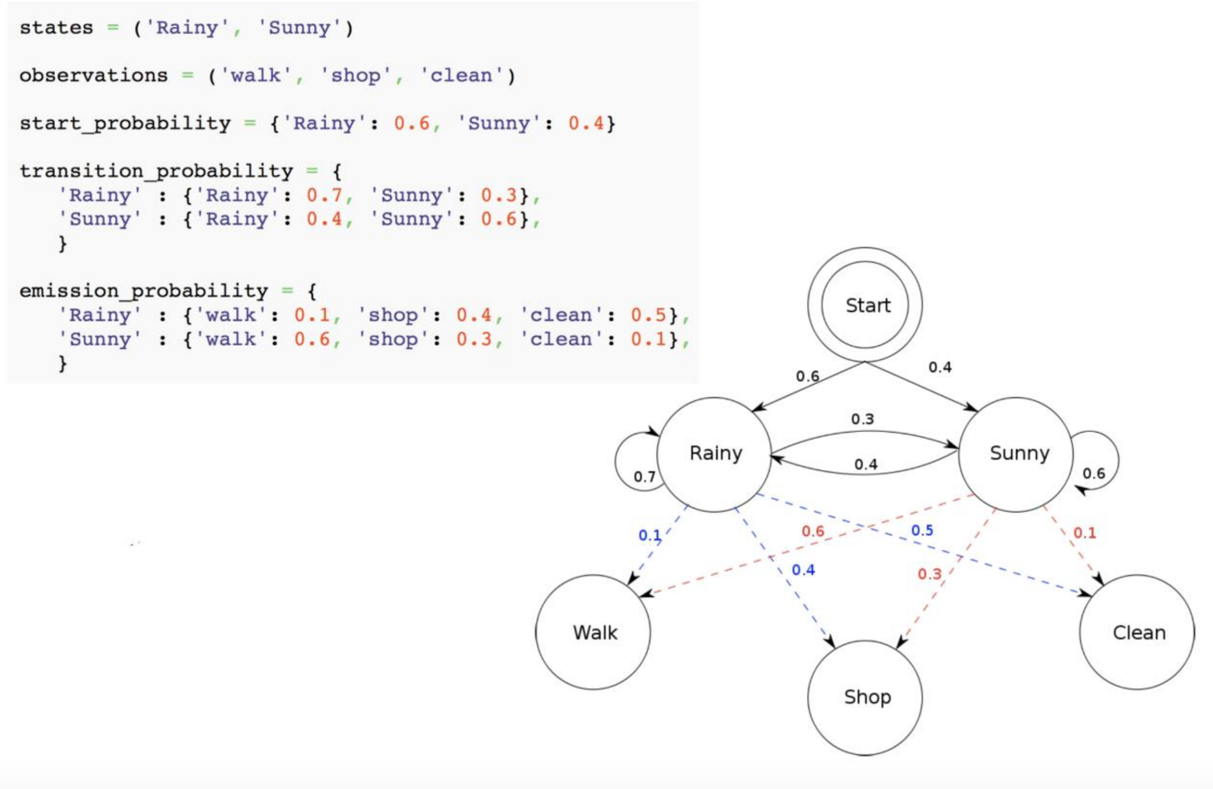
| AuthorID | AuthorName | AuthorBDay |
|---|---|---|
| 345-28-2938 | Haile Selassie | 14-Aug-92 |
| 392-48-9965 | Joe Blow | 14-Mar-15 |
| 454-22-4012 | Sally Hemmings | 12-Sept-70 |
| 663-59-1254 | Hannah Arendt | 12-Mar-06 |

| ISBN | AuthorID | PubID | Date | Title |
|---|---|---|---|---|
| 1-34532-482-1 | 345-28-2938 | 03-4472822 | 1990 | Cold Fusion for Dummies |
| 1-38482-995-1 | 392-48-9965 | 04-7733903 | 1985 | Macrame and Straw Tying |
| 2-35921-499-4 | 454-22-4012 | 03-4859223 | 1952 | Fluid Dynamics of Aquaducts |
| 1-38278-293-4 | 663-59-1254 | 03-3920886 | 1967 | Beads, Baskets & Revolution |

- Combinatorial generalization
- Relational inductive biases
- Graph network
- Discussion & Conclusion

# Relational reasoning

- Structure:
  - Composing a set of known building blocks
    - Structured representation
    - Structured computation



```
states = ('Rainy', 'Sunny')

observations = ('walk', 'shop', 'clean')

start_probability = {'Rainy': 0.6, 'Sunny': 0.4}

transition_probability = {
   'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3},
   'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
   }

emission_probability = {
   'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
   'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},
   }
```

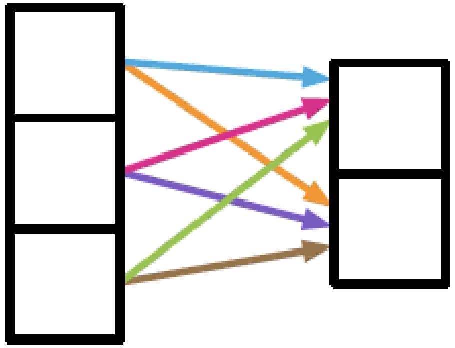
# (Relational) Inductive biases

- Allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data
- Impose constraints on relationships and interactions among entities in a learning process

**L2 regularization**

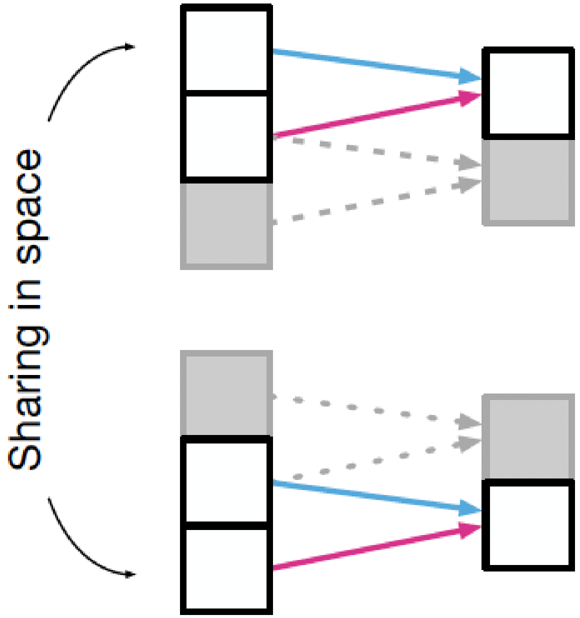
# Fully connected layers

- Entity:
    - Units
- Relation
    - All-to-all
- Relational inductive biases
    - Weak

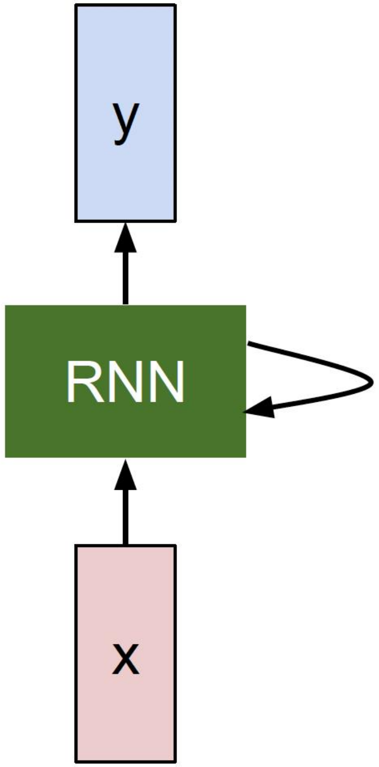


# Convolutional layer

- Entities
  - Grid elements
- Relations:
  - Local
- Relational inductive biases
  - Locality



# RNN review



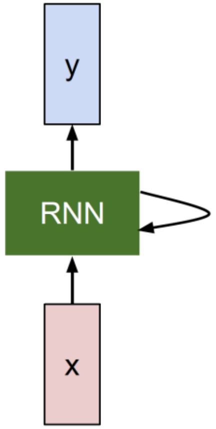
$$h_t = f_W(h_{t-1}, x_t)$$

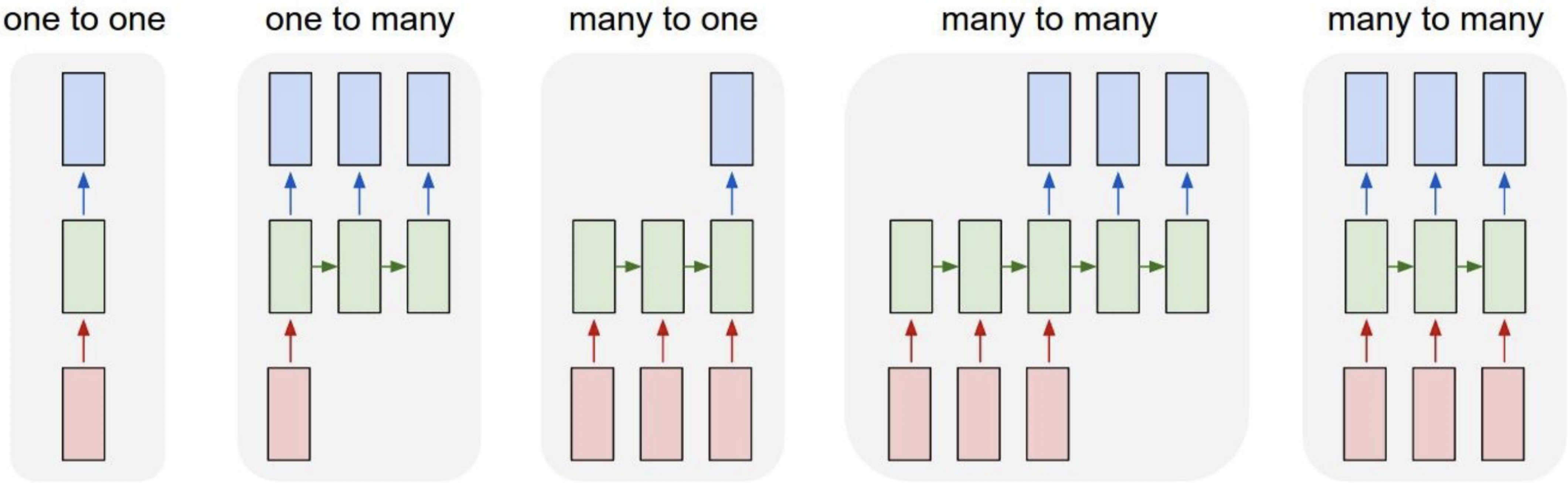
$$\downarrow$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

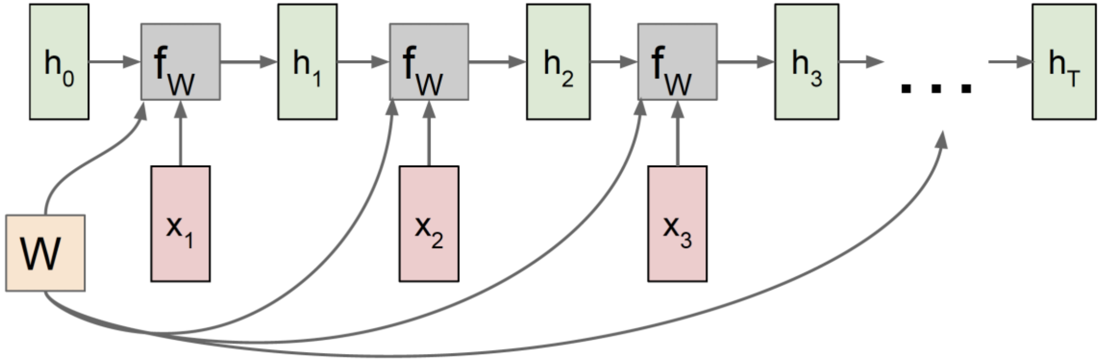
$$y_t = W_{hy}h_t$$

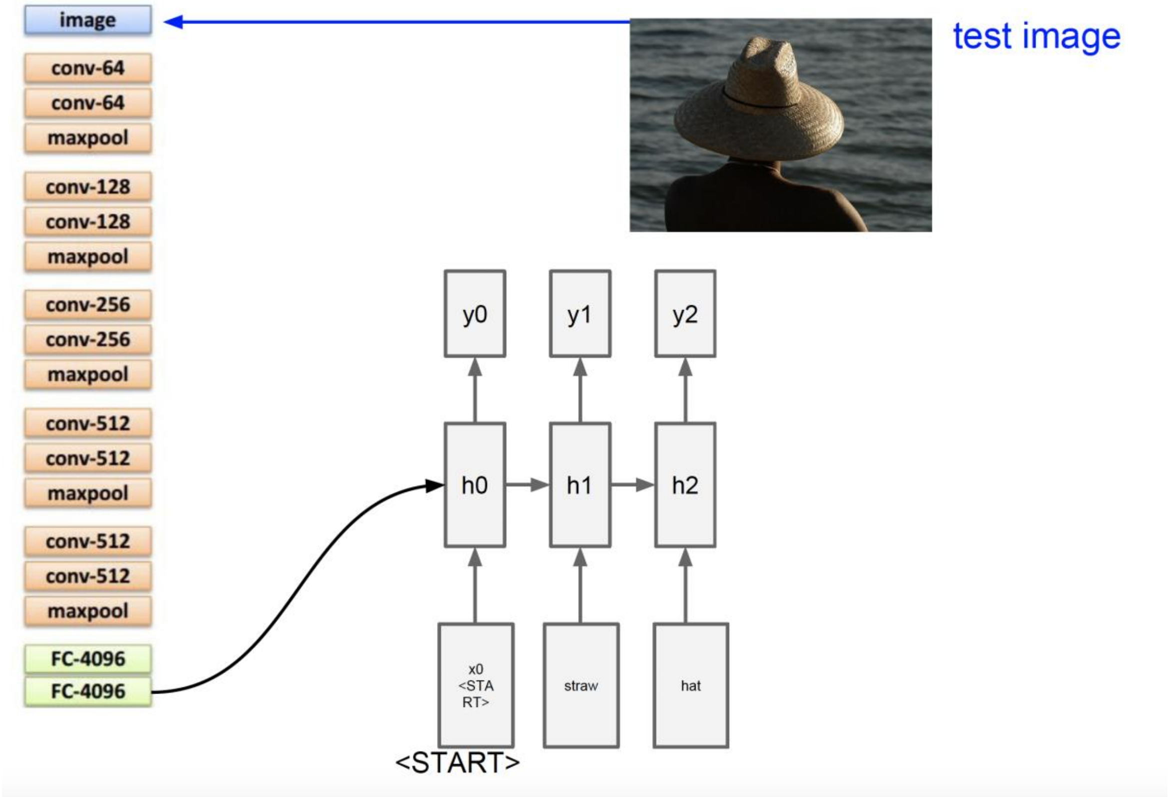
# RNN review (cont`d)





# RNN review(cont'd)





# Long Short Term Memory (LSTM)

## Vanilla RNN

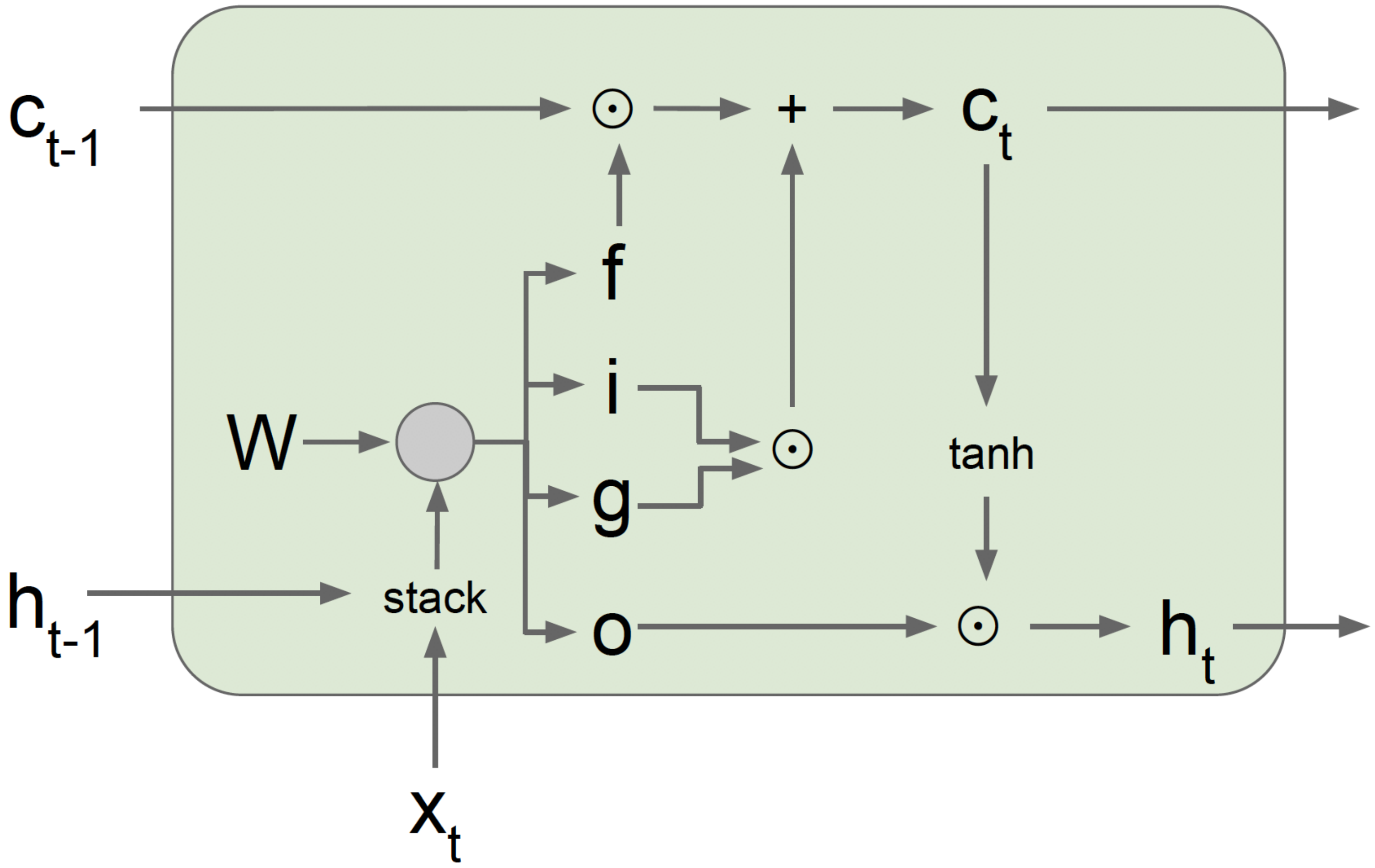
$$\boxed{h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)}$$

## LSTM

$$\boxed{\begin{gathered} \begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \\ c_t = f \odot c_{t-1} + i \odot g \\ h_t = o \odot \tanh(c_t) \end{gathered}}$$

# Long Short Term Memory (LSTM)

*[Hochreiter et al., 1997]*



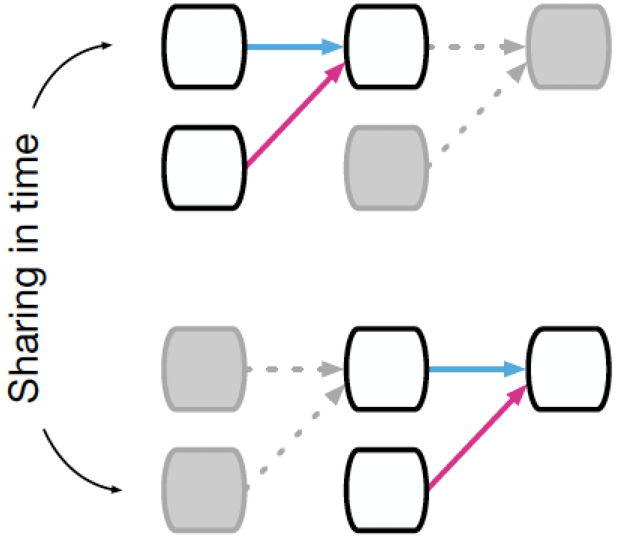
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

# Recurrent layer

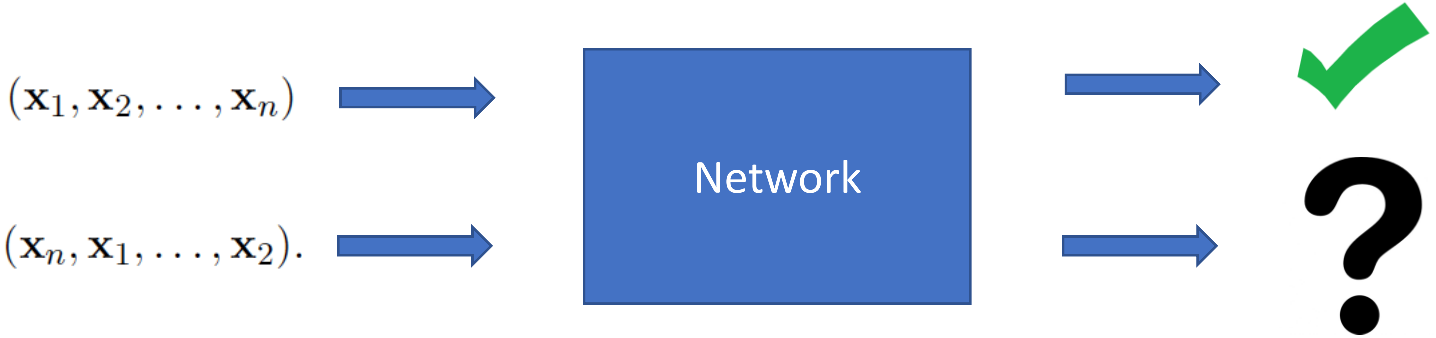
- Entities
  - Timesteps
- Relations:
  - Sequential
- Relational inductive biases
  - Sequentiality



- Combinatorial generalization
- Relational inductive biases
- Graph network
- Discussion & Conclusion

# Some shortcomings of previous layers

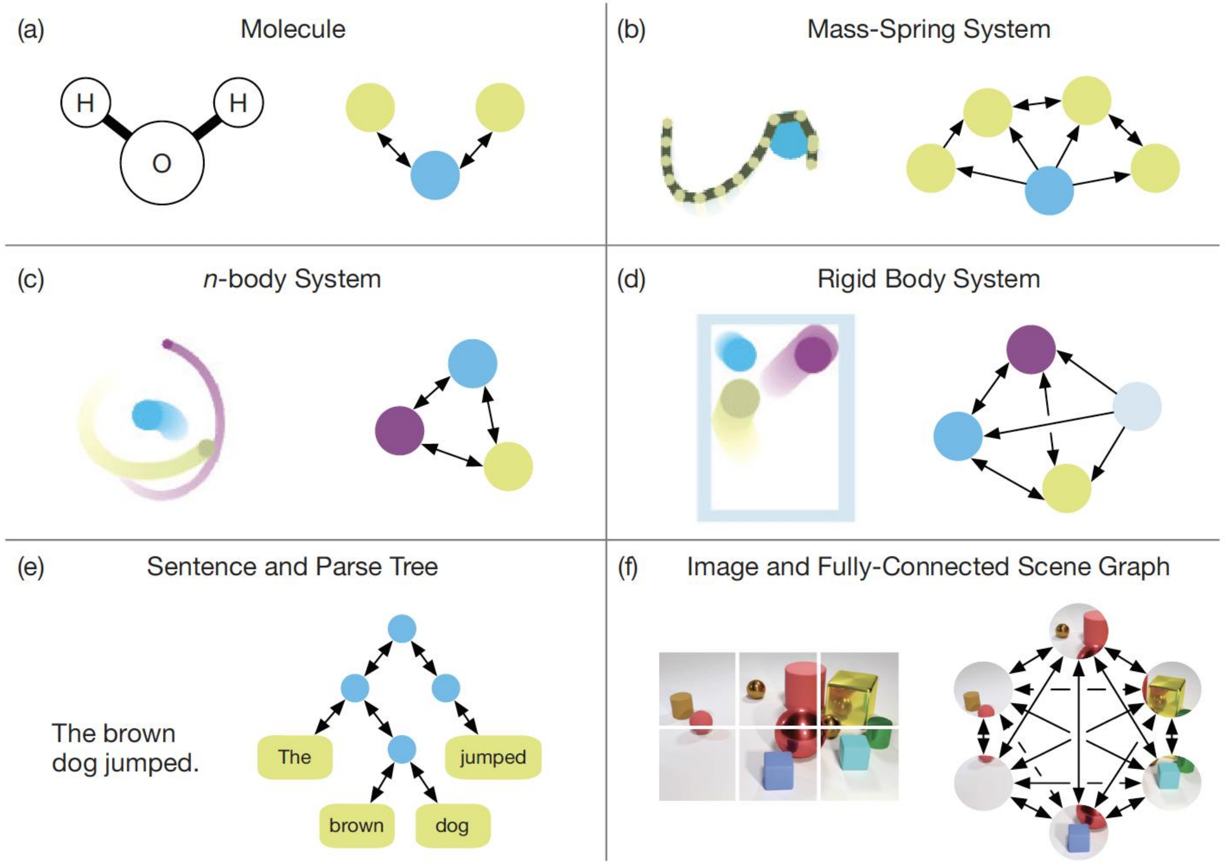
- Entities in the world (such as objects and agents) do not have a natural order


$(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
$(\mathbf{x}_n, \mathbf{x}_1, \ldots, \mathbf{x}_2).$
Network
✓
?

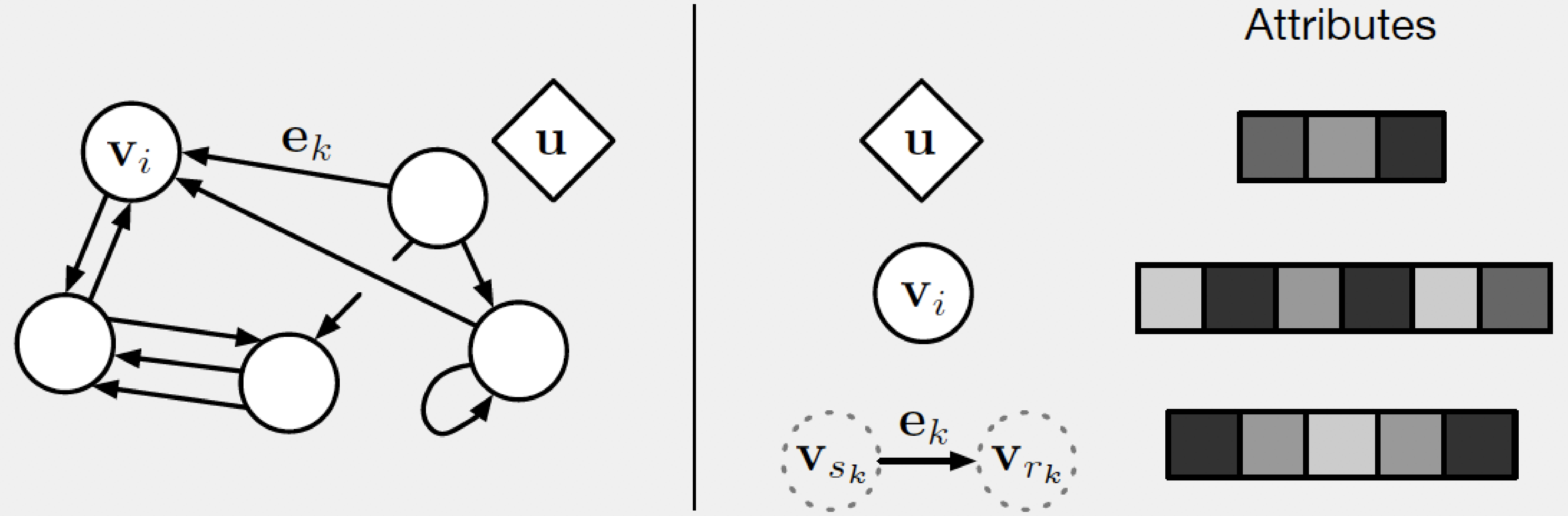
- Cannot operate on arbitrary relational structure

# Graph network

- Entities
    - Nodes
- Relations:
    - Edges
- Relational inductive biases
    - Node, edge permutations

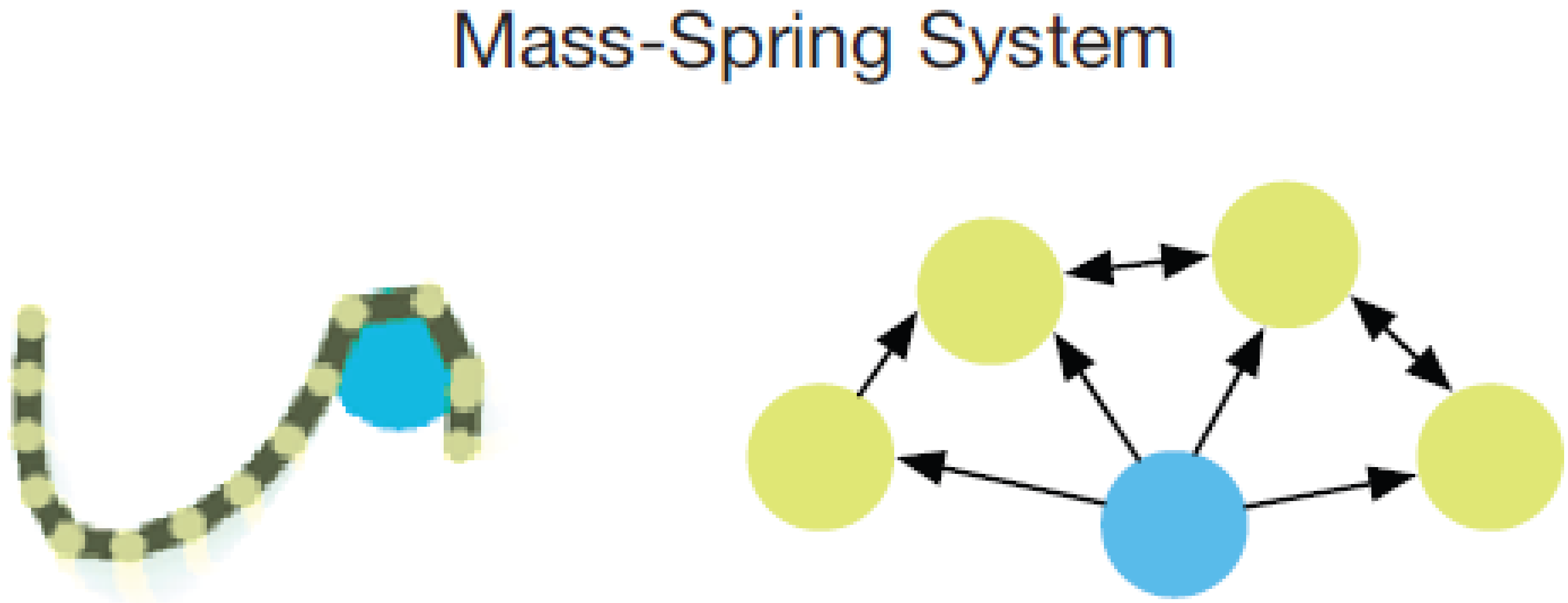


## Box 3: Our definition of “graph”



Directed : one-way edges, from a “sender” node to a “receiver” node.
Attribute : properties that can be encoded as a vector, set, or even another graph.
Attributed : edges and vertices have attributes associated with them.
Global attribute : a graph-level attribute.
Multi-graph : there can be more than one edge between vertices, including self-edges.

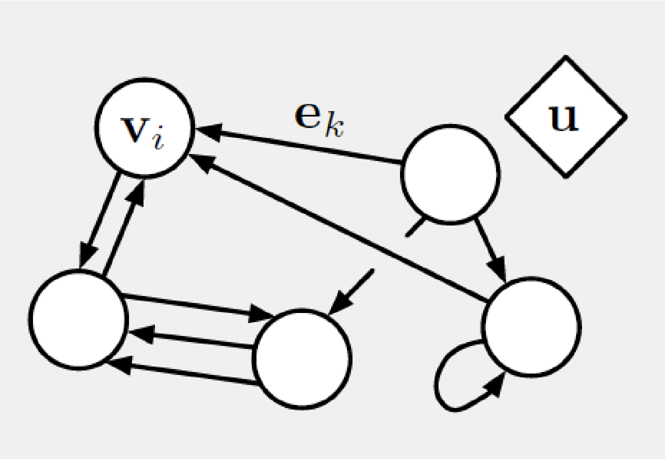
# Internal structure of a GN block



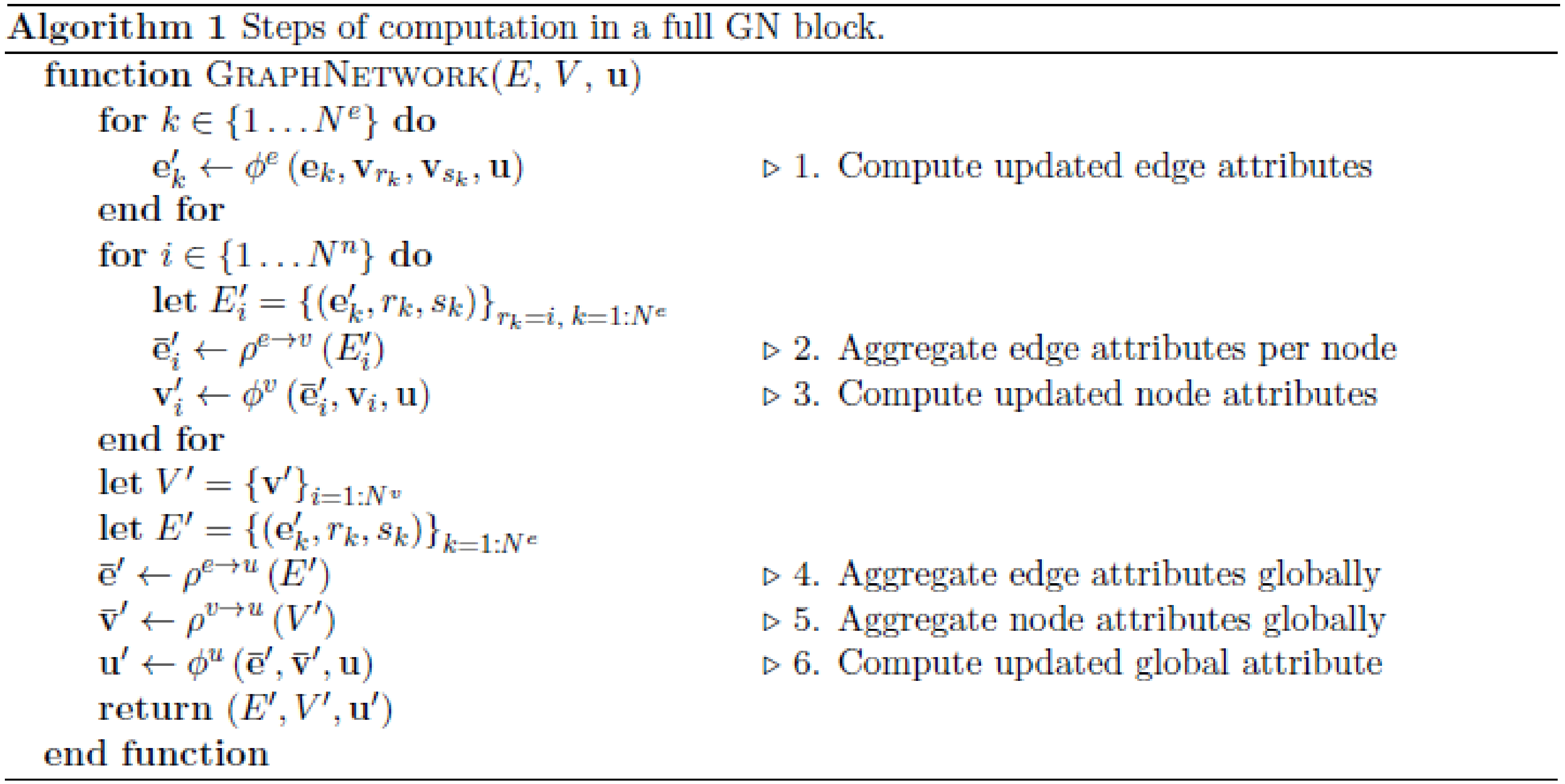
# Internal structure of a GN block

$$\begin{aligned} \mathbf{e}'_k &= \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right) & \bar{\mathbf{e}}'_i &= \rho^{e\rightarrow v}\left(E'_i\right) \\ \mathbf{v}'_i &= \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right) & \bar{\mathbf{e}}' &= \rho^{e\rightarrow u}\left(E'\right) \\ \mathbf{u}' &= \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right) & \bar{\mathbf{v}}' &= \rho^{v\rightarrow u}\left(V'\right) \end{aligned} \tag{1}$$

where $E'_i = \left\{(\mathbf{e}'_k, r_k, s_k)\right\}_{r_k=i,\; k=1:N^e}$, $V' = \left\{\mathbf{v}'_i\right\}_{i=1:N^v}$, and $E' = \bigcup_i E'_i = \left\{(\mathbf{e}'_k, r_k, s_k)\right\}_{k=1:N^e}$.



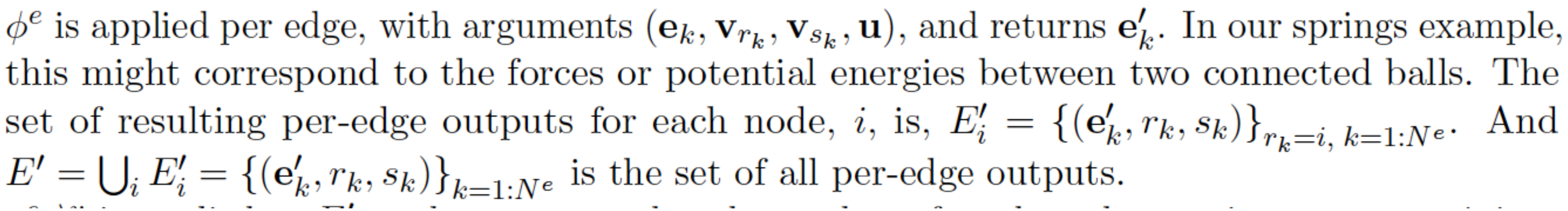
# Computational steps within a GN block



**Algorithm 1** Steps of computation in a full GN block.

```
function GraphNetwork($E$, $V$, $\mathbf{u}$)
    for $k \in \{1 \ldots N^e\}$ do
        $\mathbf{e}'_k \leftarrow \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right)$        ▷ 1. Compute updated edge attributes
    end for
    for $i \in \{1 \ldots N^n\}$ do
        let $E'_i = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{r_k = i,\; k=1:N^e}$
        $\bar{\mathbf{e}}'_i \leftarrow \rho^{e \rightarrow v}\left(E'_i\right)$        ▷ 2. Aggregate edge attributes per node
        $\mathbf{v}'_i \leftarrow \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right)$        ▷ 3. Compute updated node attributes
    end for
    let $V' = \left\{\mathbf{v}'\right\}_{i=1:N^v}$
    let $E' = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{k=1:N^e}$
    $\bar{\mathbf{e}}' \leftarrow \rho^{e \rightarrow u}\left(E'\right)$        ▷ 4. Aggregate edge attributes globally
    $\bar{\mathbf{v}}' \leftarrow \rho^{v \rightarrow u}\left(V'\right)$        ▷ 5. Aggregate node attributes globally
    $\mathbf{u}' \leftarrow \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right)$        ▷ 6. Compute updated global attribute
    return $(E', V', \mathbf{u}')$
end function
```

# Computational steps within a GN block(1)

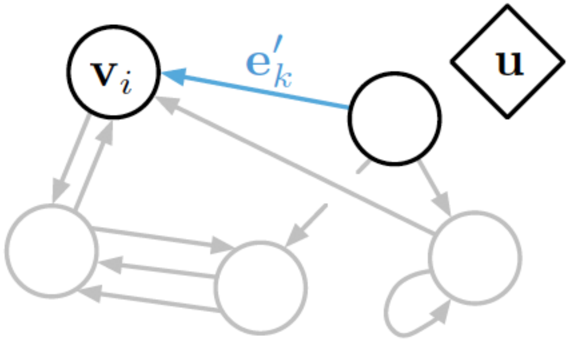


$\phi^e$ is applied per edge, with arguments $(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u})$, and returns $\mathbf{e}'_k$. In our springs example, this might correspond to the forces or potential energies between two connected balls. The set of resulting per-edge outputs for each node, $i$, is, $E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{r_k=i,\; k=1:N^e}$. And $E' = \bigcup_i E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{k=1:N^e}$ is the set of all per-edge outputs.

$$\mathbf{e}'_k = \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right)$$

$$E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{r_k=i,\; k=1:N^e},$$

$$E' = \textstyle\bigcup_i E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{k=1:N^e}$$



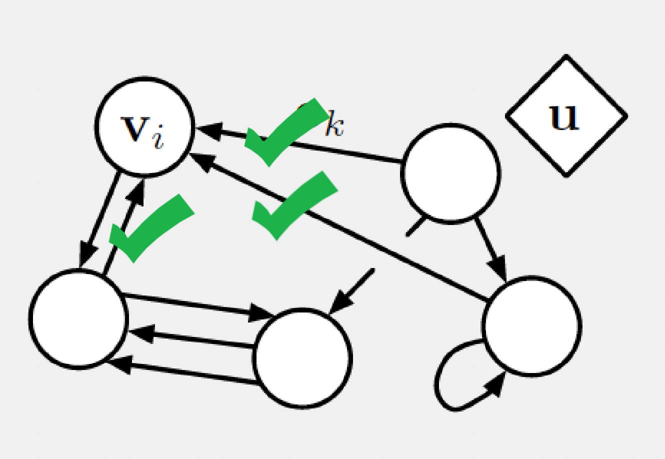
(a) Edge update

# Computational steps within a GN block(2)

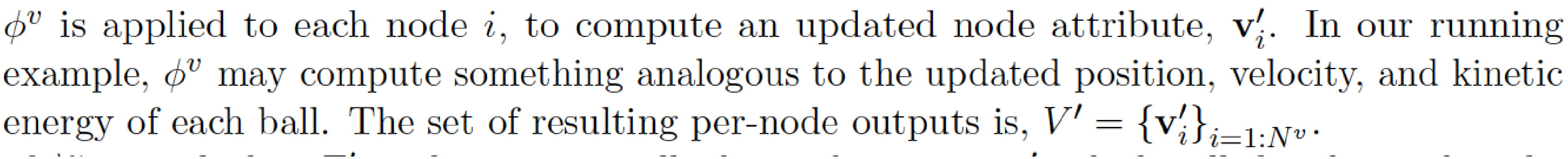
$\rho^{e \to v}$ is applied to $E'_i$, and aggregates the edge updates for edges that project to vertex $i$, into $\bar{\mathbf{e}}'_i$, which will be used in the next step's node update. In our running example, this might correspond to summing all the forces or potential energies acting on the $i^{\text{th}}$ ball.

$$\bar{\mathbf{e}}'_i = \rho^{e \to v}\left(E'_i\right)$$

$$E'_i = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{r_k = i,\; k = 1:N^e},$$



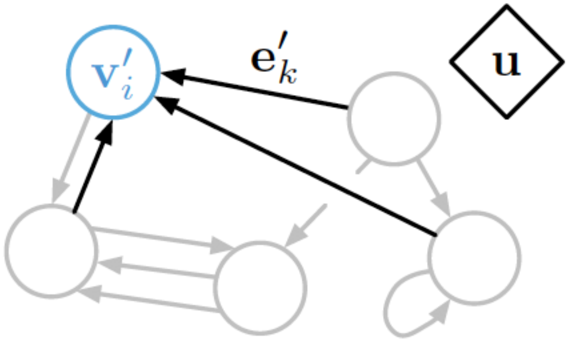
# Computational steps within a GN block(3)



$\phi^v$ is applied to each node $i$, to compute an updated node attribute, $\mathbf{v}'_i$. In our running example, $\phi^v$ may compute something analogous to the updated position, velocity, and kinetic energy of each ball. The set of resulting per-node outputs is, $V' = \{\mathbf{v}'_i\}_{i=1:N^v}$.

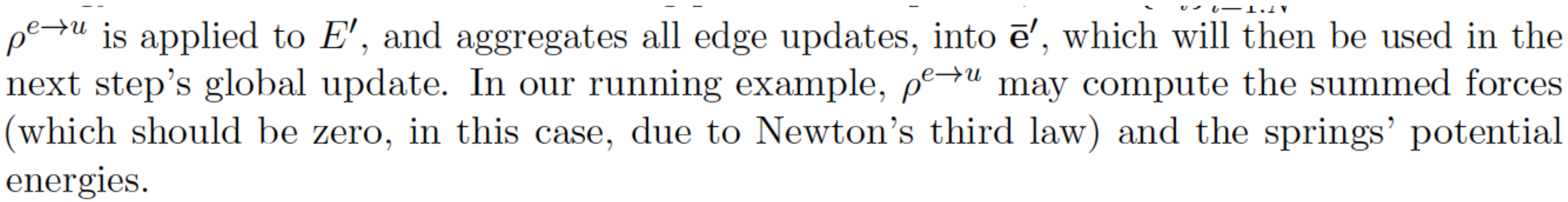
$$\mathbf{v}'_i = \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right)$$

$$V' = \{\mathbf{v}'_i\}_{i=1:N^v}$$

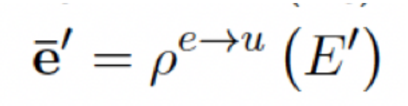


(b) Node update

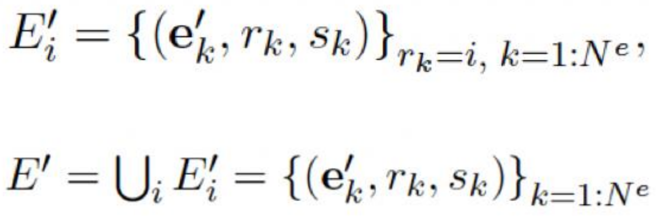
# Computational steps within a GN block(4)



$\rho^{e\to u}$ is applied to $E'$, and aggregates all edge updates, into $\bar{\mathbf{e}}'$, which will then be used in the next step's global update. In our running example, $\rho^{e\to u}$ may compute the summed forces (which should be zero, in this case, due to Newton's third law) and the springs' potential energies.



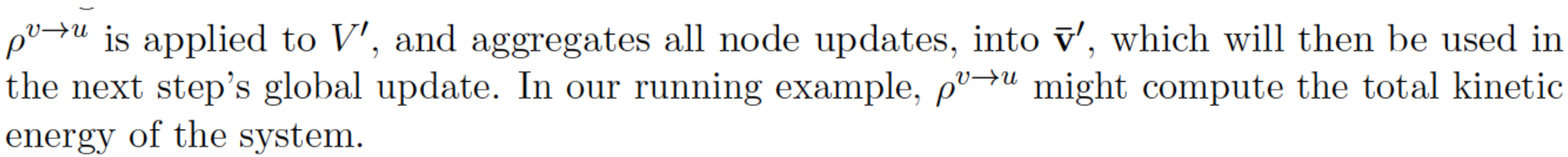
$$\bar{\mathbf{e}}' = \rho^{e\to u}\left(E'\right)$$



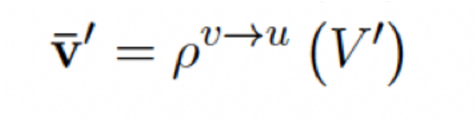
$$E'_i = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{r_k=i,\ k=1:N^e},$$

$$E' = \bigcup_i E'_i = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{k=1:N^e}$$

# Computational steps within a GN block(5)



$\rho^{v \rightarrow u}$ is applied to $V'$, and aggregates all node updates, into $\bar{\mathbf{v}}'$, which will then be used in the next step's global update. In our running example, $\rho^{v \rightarrow u}$ might compute the total kinetic energy of the system.

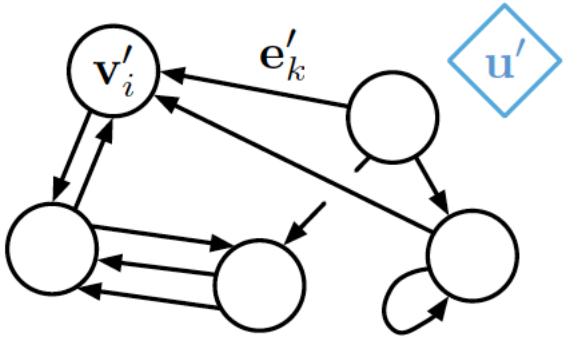


$$\bar{\mathbf{v}}' = \rho^{v \rightarrow u}\left(V'\right)$$

# Computational steps within a GN block(6)

. $\phi^u$ is applied once per graph, and computes an update for the global attribute, $\mathbf{u}'$. In our running example, $\phi^u$ might compute something analogous to the net forces and total energy of the physical system.

$$\mathbf{u}' = \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right)$$



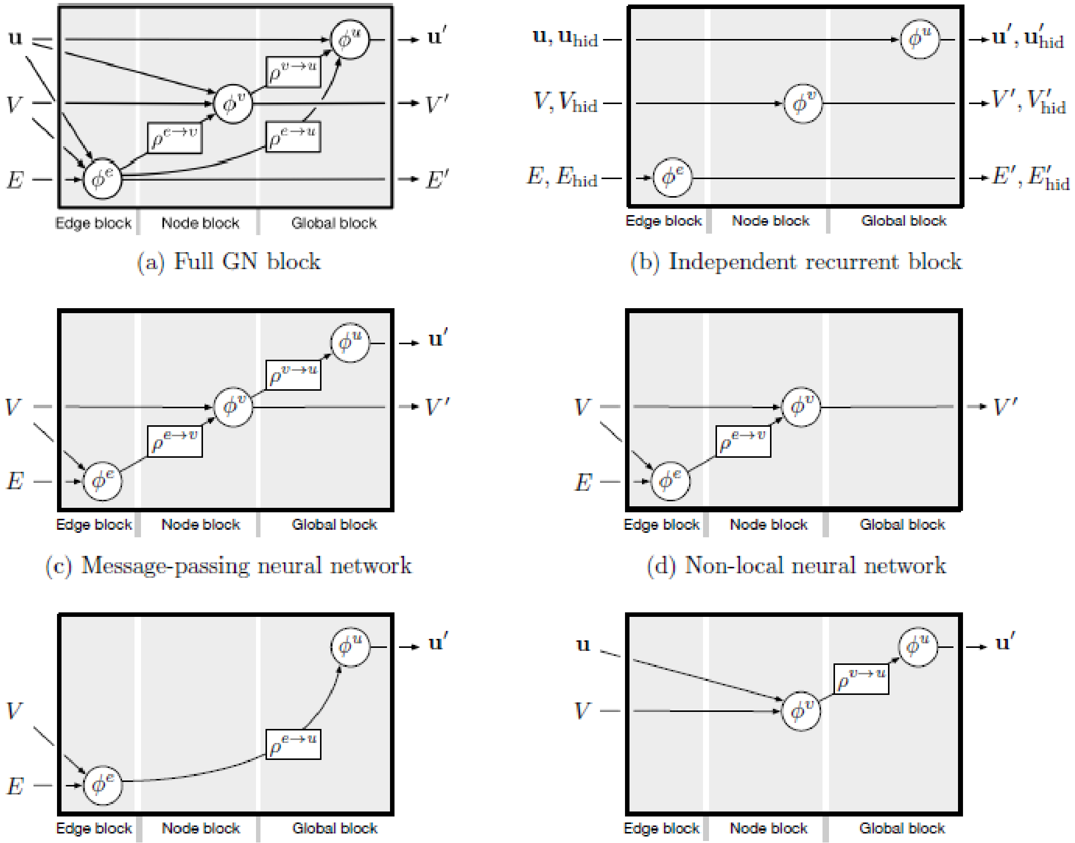
(c) Global update

# Flexible representations

- Attributes -- a GN's output to be passed to other deep learning building blocks such as MLPs, CNNs, and RNNs

- Graph structure – 1 the input explicitly specify the relational structure; 2 the relational structure must be inferred or assumed

# Configurable within-block structure



(a) Full GN block

(b) Independent recurrent block

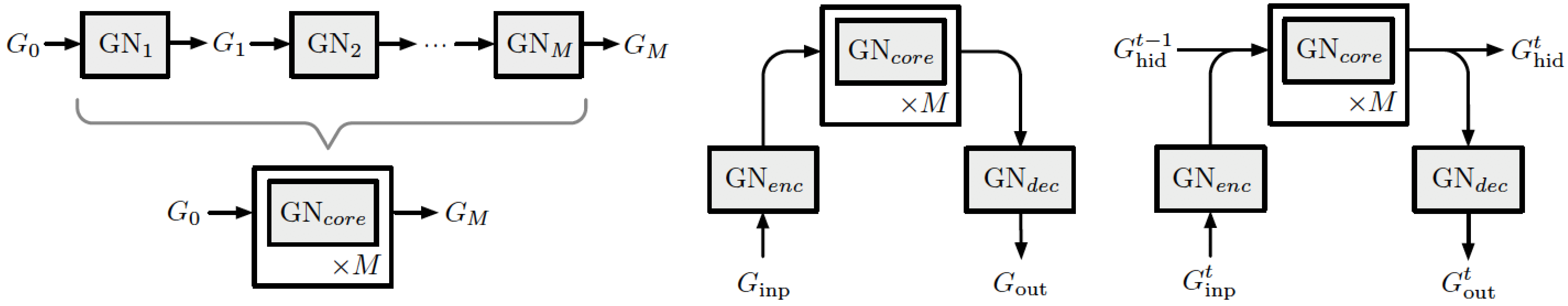
(c) Message-passing neural network

(d) Non-local neural network

(e) Relation network

(f) Deep set

# Composable multi-block architectures



- Combinatorial generalization
- Relational inductive biases
- Graph network
- Discussion & Conclusion

- CNNs and RNNs do contain relational inductive biases, they cannot naturally handle more structured representations such as sets or graphs
- The structure of GNs naturally supports combinatorial generalization because they do not perform computations strictly at the system level, but also apply shared computations across the entities and across the relations as well.
- Notions like recursion, control flow, and conditional iteration are not straightforward to represent with graphs, and, minimally, require additional assumptions