Interaction Networks for Learning about Objects , Relations and Physics
(2016)

Battaglia Matthew, Peter W Lai,
Resume

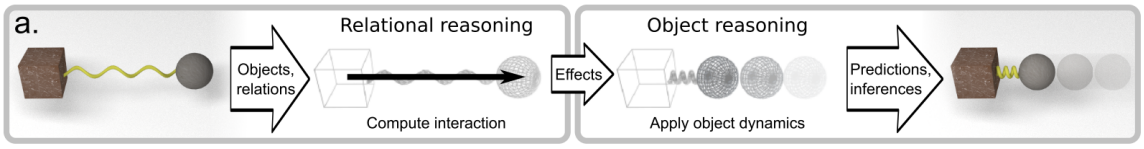
January 8, 2019

Abstract

Reasoning about objects, relations, and physics is central to human intelligence, and a key goal of artificial intelligence. Here they introduce an interaction network, takes graphs as inputs (the graph expresses the objects and the relations), and external effect, and then performs object-and relation-centric reasoning in a way that is analogous to a simulation, and is implemented using deep neural networks.

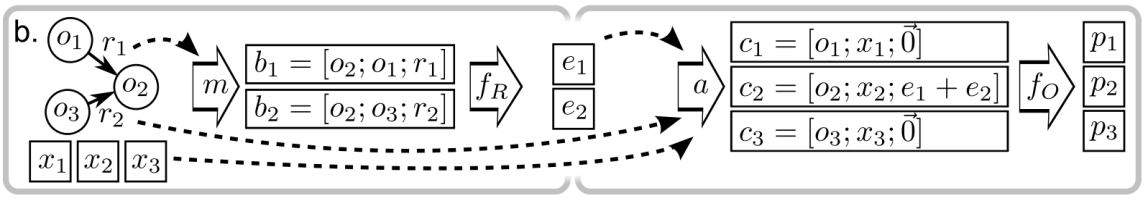
1 Introduction

Predicting what will happen next in physical environments or inferring underlying properties of complex scenes, can be challenging because their elements can be composed in combinatorially many possible arrangements. People can nevertheless solve such problems by decomposing the scenario into distinct objects and relations, and reasoning about the consequences of their interactions and dynamics. Here



Let's take the example above, the network takes as inputs the objects (with a set of features for each object, such as their positions, velocities, masses, forms ...), and the relations (say the ball will be in a relation to the cube - force applied - , in this case the cube is the sender and the ball is the receiver), the network first uses the objects and the relations to contruct the effects of the relations to each objects, and then uses the local and external effects to predict the next state of the objects.

2 Model



The model takes as input a graph that represents a system of objects, o_j, and relations, $\langle i, j, r_k \rangle_k$, instantiates the pairwise interaction terms, b_k, and computes their effects, e_k, via a relational model, $f_R()$. The ek are then aggregated and combined with the o_j and external effects, x_j, to generate input $(a_s c_j)$, for an object model, $f_O()$, which predicts how the interactions and dynamics influence the objects, p.

2.1 Model components

- Inputs The inputs to the model are 1) a graph $G =< O, R >$, The nodes O are the objects and the edges R are the realtions. and 2) the external effects X

- The Objects: $O = \{o_j\}_{j=1...N_O}$, representing the states of each object, O is a matrix of size $D_S \times N_O$, where each object (out of N_O) is represented as a vector of features/states of size D_S.
- The relations : $R = \langle R_r, R_s, R_a \rangle$, where R_r and R_s are two binary matrices of size $(N_O \times N_R)$ to specify which objects are the receivers and the senders. Example from the Figure above :

$$R_r = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \text{ And } R_s = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

- The external effects X; a matrix of size $(D_X \times N_O)$

- Functions There are mainly four functions,

- The marshalling function $m(G) = [OR_r; OR_s; R_a] = B$, taking the input graph and rearranging the objects and relations into interaction terms (triplets, two objects, one sender and the other is the receiver, and the relation attribute in between) $b_k = \langle o_i, o_j, r_k \rangle \in B$. This is done by computing the matrix products OR_r (maintaining only the receivers) and OR_s (only the senders), and concatenates them with R_a (to add the third term to the triplet [Receiver Sender Rela]). The resulting matrix is a $(2D_S + D_R) \times N_R$.
- Relation-centric function $f_R(b_k) = e_k$, implemented as a MLP to each colums (here we have a local connectivity across the triplets, similar to the CNN), the ouputs is D_E length vector e_k, so the function / model concatenates the objects (per pairs of sender and receiver = per colums), and ouputs a new vector describing the effect (say a change in the position), so the resulting matrix is of size : $D_E \times N_R$, called the effect matrix.
- $a(G, X, E) = C = \{c_j\}_{j=1...N_O}$ the function a applies the effect to each object-receiver, the effects applied are only the local ones, the ones inferenced by the function f_R. This is dones computing the product $\bar{E} = ER_r^T$ of size $D_E \times N_R$, now we concatenate the Objects O, the external effects we got as inputs X. $a(G, X, E) = [O; X; E] = C$ and the resulting C is a $(D_S + D_X + D_E) \times N_O$ matrix.
- Not that we have the local and external effects, with the states of the object (on the time step t) We can compute the new state of the object at the new time step o_{t+1}, this is done using object-centric function $f_O(c_j) = p_j$. f_O applied another MLP which return D_P length vector that are concatenated to form the ouput matrix.
- Another function g can be added to estimate the abstract propreties of the system

3 Experiments

To test the model disscussed above, the authors apply it to various problems, one of the expriments explored is predicting future states of a system, in the n-body domain, such as solar systems, all n bodies extert distance and mass dependent gravitaional forces on each other, so there is $n(n-1)$ relations input to the models (the features of the objects are their mass, position, velocity ...), with

external effects (gravitationnal accelation) and the relations (gravitationnal atraction, collisions ...), and the model outputs say, the position and velocity at time step $t+1$, which will be the inputs of the model in the next state.