Hierarchical Relational Inference

Anonymous Authors[1]

Abstract

Common-sense physical reasoning requires learning about the interactions of objects and their dynamics. The notion of an abstract object, however, encompasses a wide variety of physical objects that differ greatly in terms of the complex behaviors they support. To address this, we propose a novel approach to physical reasoning that models objects as hierarchies of parts that may locally behave separately, but also act more globally as a single whole. Unlike prior approaches, our method learns in an unsupervised fashion directly from raw visual images to discover objects, parts, and their relations. We demonstrate how it improves over a strong baseline at modeling synthetic and real-world physical dynamics.

1 Introduction

Common-sense physical reasoning in the real world involves making predictions from complex high-dimensional observations. Humans somehow discover and represent abstract objects to compactly describe complex visual scenes in terms of 'building blocks' that can be processed separately (Spelke & Kinzler, 2007). They model the complex physical real-world by reasoning about dynamics of high-level objects such as footballs and football players and the consequences of their interactions. It is natural to expect that artificial agents operating in the real world will benefit from a similar approach (Lake et al., 2015).

Real world objects vary greatly in terms of their properties, which complicates modelling their dynamics. Often, these can be viewed as a *hierarchy* of parts that locally behave somewhat independently of each other, but also act more globally as a single whole (Mrowca et al., 2018; Lingelbach et al., 2020). This suggests to simplify models of object dynamics by explicitly distinguishing multiple levels of abstraction, separating hierarchical sources of influence.

Prior approaches to common-sense physical reasoning explicitly consider objects and relations at a representational level, e.g., (Battaglia et al., 2016; Chang et al., 2017; van Steenkiste et al., 2018; Kipf et al., 2018). They decompose complex physical interactions in the environment into pairwise interactions between objects, modelled efficiently by Graph Networks (Battaglia et al., 2018). Here the representation of each object is updated at each time step by propagating 'messages' through the corresponding interaction graph. While recent approaches (specifically) address the challenge of learning object representations from raw visual data (Greff et al., 2017; Kosiorek et al., 2018; van Steenkiste et al., 2018; Burgess et al., 2019; Greff et al., 2019) and of dynamically inferring relationships between objects (van Steenkiste et al., 2018; Kipf et al., 2018; Goyal et al., 2019; Veerapaneni et al., 2019), reasoning about the dynamics and interactions of complex objects remains difficult without incorporating additional structure. On the other hand, approaches that consider part-based representations of objects and hierarchical interaction graphs lack the capacity to learn from raw visual images and dynamically infer relationships (Mrowca et al., 2018; Lingelbach et al., 2020).

Here we propose *Hierarchical Relational Inference* (HRI), a novel approach to common-sense physical reasoning capable of learning to discover objects, parts, and their relations, directly from raw visual images in an unsupervised fashion. HRI extends *Neural Relational Inference* (NRI) (Kipf et al., 2018) in two regards. Firstly, it considers part-based representations of objects and infers *hierarchical* interaction graphs to simplify modelling their dynamics (and interactions). This necessitates a more efficient message-passing approach that leverages the hierarchical structure, which we will also introduce. Secondly, it provides a mechanism for applying NRI (and thereby HRI) to raw visual images that infers part-based object representations spanning multiple levels of abstraction. We evaluate HRI on synthetic and real physical dynamics prediction tasks and demonstrate how it improves over a number of strong baselines.

2 Method

Motivated by how humans learn to perform common-sense physical reasoning, we propose Hierarchical Relational Inference (HRI). It consists of a *visual encoder*, a *relational inference module*, a *dynamics predictor*, and a *visual decoder*. All are trained end-to-end in an unsupervised manner.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Do not distribute.

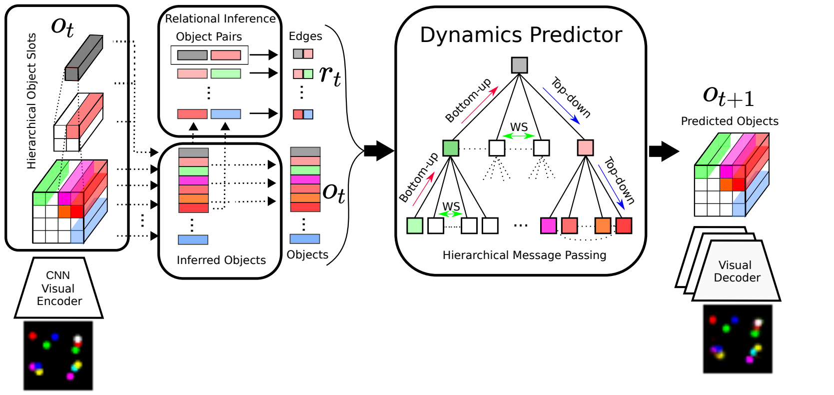


Figure 1: **HRI.** A visual encoder infers part-based object representations, which are fed to a relational inference module to obtain a hierarchical interaction graph. A dynamics predictor, using hierarchical message-passing, makes predictions about future object states. Their ‘rendering’, produced by a visual decoder is compared to the next frame to train the system.

2.1 Inferring Objects, Parts, and their Relations

To make physical predictions about a stream of complex visual observations, we will focus on the underlying *interaction graph*. It distinguishes objects or parts (corresponding to nodes) and the relations that determine interactions between them (the edges), which must be inferred.

Inferring Object/Part Representations The task of the visual encoder is to infer separate representations for each object from the input image. In order to relate and compare these representations efficiently, it is important that they are described in a common format. Moreover, since we are concerned with a hierarchical (i.e. part-based) representation of objects, we also require a mechanism to relate the part representations to the corresponding object representation.

Here we address these challenges by partitioning the feature maps learned by a CNN according to their spatial coordinates to obtain object representations. This is a natural choice, since CNNs excel at image processing and because weight-sharing ensures that the resulting object representations are described in a common format. Indeed, several others have proposed to learn object representations in this way (Santoro et al., 2017; Zambaldi et al., 2019). Here, we take this insight a step further and propose to learn hierarchical object representations in a similar way. In particular, we leverage the insight that the parts belonging to an object tend to be spatially close, to apply a sequence of convolutions followed by down-sampling operations to extract object-level representations from part-level representations (left side of Figure 1). We build a 3-level part-based hierarchy like this, which is then fed into the *relational module*.

Neural Relational Inference To infer relations between object representations, we will make use of NRI (Kipf et al., 2018). By default, NRI takes as input a set of object trajectories (states) and infers their pairwise relations (edges) using a Graph Neural Network (GNN) (Battaglia et al., 2018). It assumes a static interaction graph, and performs relational inference by processing the entire input sequence at once. In contrast, we will assume a *dynamic* interaction graph, which is necessary since objects move across the image and may end up in different spatial slots throughout the sequence. This is achieved by inferring edges at each time step based on a history of $k = 10$ most recent object states.

More formally, given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes (objects) $o \in \mathcal{V}$ and edges (relations) $r_{i,j} = (o_i, o_j) \in \mathcal{E}$, NRI defines a single *node-to-node message passing operation* in a GNN similar to Gilmer et al. (2017):

$$\mathbf{e}_{(i,j)} = f_e([\mathbf{o}_i, \mathbf{o}_j, \mathbf{r}_{(i,j)}]), \quad \mathbf{o}'_j = f_o([\textstyle\sum_{i \in \mathcal{N}_{\mathbf{o}_j}} \mathbf{e}_{(i,j)}, \mathbf{o}_j])$$

where $\mathbf{e}_{(i,j)}$ is an embedding (effect) of the relation $\mathbf{r}_{(i,j)}$ between objects $\mathbf{o}_i$ and $\mathbf{o}_j$, $\mathbf{o}'_j$ is the updated object embedding, $\mathcal{N}_j$ the set of indices of nodes connected by an incoming edge to object $\mathbf{o}_j$ and $[\cdot, \cdot]$ indicates concatenation. f_o and f_e are neural networks, typically simple MLPs.

The NRI ‘encoder’ receives as input a sequence of object state trajectories $\mathbf{o} = (\mathbf{o}^1, ..., \mathbf{o}^T)$, which in our case are inferred. It consists of a GNN f_ϕ that defines a probability distribution over edges $q_\phi(\mathbf{r}^t_{ij} | \mathbf{o}^{t-k:t}) = \mathrm{softmax}(f_\phi(\mathbf{o}^{t-k:t})_{ij})$, and relations are one-hot encoded. The GNN performs two stages of message passing to infer relations (details in Appendix A.3), where the initial node representations $\mathbf{o}_i$ are obtained by concatenating the corresponding object states across the window.

2.2 Physical Reasoning

Physical reasoning is performed by the *dynamics predictor*, which leverages the inferred object representations and edges to predict the object states at the next time step. To distinguish between representations at multiple levels of abstractions, HRI makes use of *hierarchical message passing*.

Hierarchical message passing We propose a more efficient message-passing procedure that leverages the hierarchical structure of the interaction graph to propagate all effects across the entire graph in a *single* step, i.e. evaluating each relation only once. Loosely inspired by Mrowca et al. (2018), it distinguishes three phases.

Starting from the leaf nodes, the *bottom-up phase* computes the effect on parent nodes $\mathbf{o}_p$ based on messages from its children, $\mathbf{e}_p^1 = \mathbf{e}_p^0 + f_{MP}^{bu}(\{\mathbf{e}_c^0\}_{c\in\mathcal{C}_p}, \mathbf{e}_p^0, \{\mathbf{r}_{cp}\}_{c\in\mathcal{C}_p})$ where $\mathcal{C}_p$ is the set of children indices of object $\mathbf{o}_p$ and the initial effects $\mathbf{e}^0$ are simply the object embeddings. In this way, global information is propagated from every node in the hierarchy to the root node. Afterwards, the bottom-up effect $\mathbf{e}_i^1$ on node $\mathbf{o}_i$ is combined with effects from its siblings (*within-sibling* phase) $\mathbf{e}_i^2 = \mathbf{e}_i^1 + f_{MP}^{ws}(\{\mathbf{e}_s^1\}_{s\in\mathcal{S}_i}, \mathbf{e}_i^1, \{\mathbf{r}_{si}\}_{s\in\mathcal{C}_i})$, where $\mathcal{S}_i$ is the set of sibling indices of object $\mathbf{o}_i$. Starting from the root node, the *top-down* phase then propagates top-down effects that are incorporated by computing $\mathbf{e}_c^3 = \mathbf{e}_c^2 + f_{MP}^{td}(\mathbf{e}_p^2, \mathbf{e}_c^2, \mathbf{r}_{pc})$ for all children $\mathbf{o}_c$ based on its parent $\mathbf{o}_p$. Functions f_{MP}^{bu}, f_{MP}^{ws}, and f_{MP}^{td} perform a single node-to-node message passing step and share weights.

Dynamics predictor Physical reasoning is performed by the dynamics predictor, which predicts future object states $p_\theta(\mathbf{o}^{t+1}|\mathbf{o}^{1:t}, \mathbf{r}^{1:t})$ from the sequence of object states and interactions. We implement this as in the NRI 'decoder', i.e. using a GNN that passes messages between objects, but with two notable differences. Firstly, we will pass messages *only* if an edge is inferred between two nodes. Secondly, we will leverage the hierarchical structure of the inferred interaction graph to perform *hierarchical message-passing*.

Since, when encoding individual images, no velocity information can be inferred to form the object state, we will consider the following recurrent update rule to predict object states at the next step:

$$\mathbf{e}^t = f_H(\mathbf{h}^t, \mathbf{r}^t), \quad \mathbf{h}^{t+1}, \mathbf{c}^{t+1} = f_{LSTM}(\mathbf{o}^t, \mathbf{e}^t, \mathbf{c}^t),$$
$$\mathbf{o}^{t+1} = f_O(\mathbf{h}^{t+1}), \quad p(\mathbf{o}_j^{t+1}|\mathbf{o}^{1:t}, \mathbf{r}^{1:t}) = \mathcal{N}(\mathbf{o}^{t+1}, \sigma^2\mathbf{I}),$$

where $\mathbf{c}$ and $\mathbf{h}$ are LSTM's cell and hidden state respectively (Hochreiter & Schmidhuber, 1997), σ^2 is a fixed variance, $\mathbf{e}^t$ is the effect computed by the hierarchical message passing module f_H, and f_O is an output MLP.

2.3 Learning

We will use a prediction objective in pixel space to learn about physical interactions. This necessitates a mechanism to 'render' the updated object representations. In this case, HRI can be viewed as a type of Variational Auto-Encoder (VAE) (Kingma & Welling, 2013), where the inferred edges are treated as the latent variables, and maximize the standard ELBO objective for the predicted frames:

$$\mathcal{L} = \mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{r}, \mathbf{o})] - D_{\mathrm{KL}}[q_\phi(\mathbf{r}|\mathbf{x})||p_\theta(\mathbf{r})].$$

The relational module $q_\phi(\mathbf{r}|\mathbf{x})$ outputs a factorized distribution over $\mathbf{r}_{ij}$, which in our case is a categorical variable that can take on two values (one-hot encoded) that indicate the presence of an edge between $\mathbf{o}_i$ and $\mathbf{o}_j$. The edge prior $p_\theta(\mathbf{r}) = \prod_{i\neq j} p_\theta(\mathbf{r}_{ij})$ is a factorized uniform distribution. Given the inferred interaction graph, the dynamics predictor and visual decoder define $p_\theta(\mathbf{x}|\mathbf{r}, \mathbf{o})$.

Visual Decoder The visual decoder renders the updated object states. It ensures compositionality in pixel space by decoding objects separately followed by a summation to produce the final image. This implements a stronger inductive bias that encourages slots to capture a particular object (since images are composed of objects) and makes it easier to inspect the representational content of each slot.

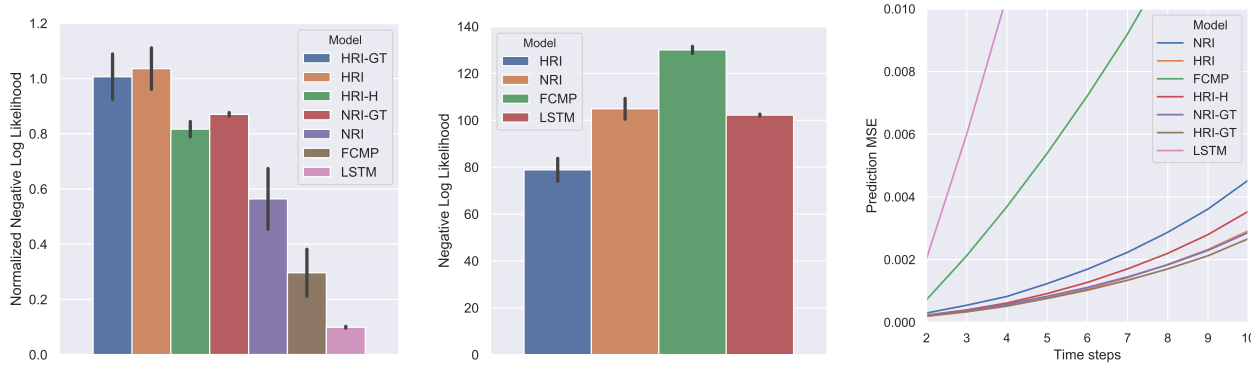
3 Experiments

We evaluate HRI on three different dynamics modelling tasks: state trajectories of objects connected via finite-length springs in a hierarchical structure (*state-springs*); corresponding rendered videos (*visual-springs*); and videos of human moving bodies (*Human3.6M*) (Ionescu et al., 2013). We compare HRI to NRI, which performs relational inference but lacks a hierarchical inductive bias, and to an LSTM that concatenates representations from all objects and predicts them jointly, but lacks a relational inference mechanism all together. All experimental details (including architectures) can be found in Appendix A.

3.1 State Springs Dataset

We consider synthetic physical systems containing simple objects connected via finite-length springs that can be organized according to a hierarchical interaction graph (Figure 3, middle row). We experiment with hierarchies containing 4 intermediate nodes, each having 3 or 4 leaf nodes, denoted as *4-3-state-springs* and *3-3-state-springs*, respectively (results for the latter are available in Appendix B). Inputs are 4-dimensional state trajectories: $x(t), y(t), \Delta x(t), \Delta y(t)$.

Comparison to baselines We compare HRI to NRI and LSTM on *4-3-state-springs* (Figure 2a), in terms of the negative log likelihood inversely proportional to a version of HRI that operates on the ground-truth interaction graph (HRI-GT). In this case, values closer to 1.0 are better, although we also provide raw negative log likelihoods in Figure 4a, which offer the same conclusions. It can be observed that HRI markedly outperforms NRI on this task, and that both significantly improve over the LSTM (which was expected).



(a) (b) (c)

Figure 2: Performance by HRI and baselines on the 4-3-state-springs (a) and 4-3-visual-springs (b). For (a) we report the "normalized" negative log likelihood which is inversely proportional to HRI-GT (higher is better). For (b) we report negative log likelihood (lower is better). (c) MSE when predicting into the future on 4-3-state-springs (prediction rollouts).

These findings indicate that the hierarchical inductive bias in HRI is indeed highly beneficial for this task.

Analysis We consider FCMP, a variation of NRI that assumes a fixed fully connected graph, and HRI-H, which is given knowledge about the 'valid' edges in the ground-truth hierarchical graph to be inferred and performs relational inference only on those. In Figure 2a it can be observed how the lack of a relational inference module further harms performance. It can also be observed how HRI outperforms HRI-H (but see also Figure 5a), which is surprising since the latter essentially solves a simpler task. We speculate that interactions during training could be the reason for this gap.

We also consider the benefit of hierarchical message-passing in isolation by comparing HRI to NRI-GT, which receives the ground-truth interaction graph. It can be seen how the lack of hierarchical message-passing explains part of the gap between HRI and NRI, but not all of it. It suggests that by explicitly considering multiple levels of abstraction (as in HRI), conducting relational inference becomes easier.

We investigate if HRI is able to perform long-term physical predictions by increasing the number of rollout steps at test-time. In Figure 2c we report the MSE between the predicted and ground truth trajectories and compare to previous baselines and variations. It can be observed that HRI outperforms all other models, sometimes even HRI-GT, and this gap increases as we predict deeper into the future. Indeed, in Figure 3 (bottom row) it can be seen that predictions and inferred relations by HRI closely matches the ground-truth.

3.2 Visual Datasets

To generate visual data for springs we rendered only the *leaf nodes* of *4-3-state-springs* and *3-3-state-springs* as in Figure 6, top row. We also consider *Human3.6M* (Ionescu et al., 2013), for which we use the provided 2D pose projections to render 12 joints in total (3 per limb) as input.

Visual Springs We compare HRI in Figure 2b and are able to observe similar results. HRI is the best performing model, although the added complexity of inferring object states results in smaller differences (in Figure 7a we verify the correspondence between spatial slots and objects). Finally, in Figure 6 we visualize how well future predictions by HRI (10 steps) match the ground-truth. We observe that they match quite well, although compared to *spring-states* the performance has degraded slightly, especially when the number of prediction steps increases.

Human 3.6M On this more complex dataset with varying dynamics, we find that HRI is the best performing model although the margins are smaller compared to before (Figure 4c). This can be explained by the fact that many samples involve relatively little motion or only motion in a single joint (and thereby lack hierarchical interactions). Example future predictions by HRI (10 steps) for this dataset can be seen in Figure 8. Their quality is similar to visual springs.

4 Conclusion

We introduced Hierarchical Relational Inference (HRI), a novel approach to common-sense physical reasoning capable of learning to discover objects, parts, and their relations, directly from raw visual images in an unsupervised fashion. It is builds on the idea that the dynamics of complex objects are best modeled as hierarchies of parts that separates different sources of influence. We compared the physical predictions made by HRI to a strong baseline on synthetic and more real-world physics prediction tasks and were consistently able to observe HRI outperforming. Using a detailed analysis we were able to validate our design choices.

References

Battaglia, P., Pascanu, R., Lai, M., Rezende, D. J., et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pp. 4502–4510, 2016.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Chang, M., Ullman, T., Torralba, A., and Tenenbaum, J. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Bkab5dqxe.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1263–1272. JMLR. org, 2017.

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

Greff, K., van Steenkiste, S., and Schmidhuber, J. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pp. 6691–6701, 2017.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433, 2019.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2013.

Kipf, T. N., Fetaya, E., Wang, K., Welling, M., and Zemel, R. S. Neural relational inference for interacting systems. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 2693–2702, 2018. URL http://proceedings.mlr.press/v80/kipf18a.html.

Kosiorek, A. R., Kim, H., Posner, I., and Teh, Y. W. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 8615–8625, USA, 2018. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=3327757.3327951.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Lingelbach, M. J., Mrowca, D., Haber, N., Fei-Fei, L., and Yamins, D. L. K. Towards curiosity-driven learning of physical dynamics. *ICLR 2020 Bridging AI and Cognitive Science Workshop*, pp. 6, 2020.

Liu, R., Lehman, J., Molino, P., Such, F. P., Frank, E., Sergeev, A., and Yosinski, J. An intriguing failing of convolutional neural networks and the coordconv solution. In *Advances in Neural Information Processing Systems*, pp. 9605–9616, 2018.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Mrowca, D., Zhuang, C., Wang, E., Haber, N., Fei-Fei, L. F., Tenenbaum, J., and Yamins, D. L. Flexible neural representation for physics prediction. In *Advances in neural information processing systems*, pp. 8799–8810, 2018.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. A simple neural network module for relational reasoning. In *Advances* *in neural information processing systems*, pp. 4967–4976, 2017.

Spelke, E. S. and Kinzler, K. D. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

van Steenkiste, S., Chang, M., Greff, K., and Schmidhuber, J. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryH20GbRW.

Veerapaneni, R., Co-Reyes, J. D., Chang, M., Janner, M., Finn, C., Wu, J., Tenenbaum, J. B., and Levine, S. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, 2019.

Watters, N., Matthey, L., Burgess, C. P., and Lerchner, A. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. *arXiv preprint arXiv:1901.07017*, 2019.

Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D., Lillicrap, T., Lockhart, E., Shanahan, M., Langston, V., Pascanu, R., Botvinick, M., Vinyals, O., and Battaglia, P. Deep reinforcement learning with relational inductive biases. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkxaFoC9KQ.

A Experiment Details

A.1 Datasets

A.1.1 Springs

We simulate a system of moving objects connected via finite-length springs, by starting from the open source simulator implementation of Kipf et al. (2018), with the following modifications: objects are connected via finite length springs (instead of ideal springs) and the sampled graphs have a hierarchical structure in terms of connectivity, initial spatial positions and the spring constants (which reflects in the speed by which objects in different layers of the hierarchy move). Objects are connected via finite springs, which makes them act according to a modified Hooke's law:

$$F_{ij} = -k_F(r_i - r_j - l \cdot \frac{r_i - r_j}{|r_i - r_j|}), \qquad (1)$$

where r_i and r_j are (x, y)-coordinates of objects i and j, k_F is the spring constant and l is its length. The objects are connected in a hierarchical graph, and they move inside a unit box (bounce elastically from the walls). To simulate the system, we initialize the root node position randomly in a neighborhood around the center of the image by sampling its (x, y) coordinates from $\mathcal{N}(0, 0.25)$. We then initialize the intermediate and the leaf nodes randomly inside each of the four image quadrants to ensure an initial bias towards spatial grouping. In particular, we use random distributions with variance 0.25 and the means (for (x, y)-coordinates) being the centers of the four quadrants: $(-0.25, 0.25)$, $(0.25, 0.25)$, $(-0.25, -0.25)$ and $(0.25, -0.25)$. For each sample in the dataset, we sample a graph with *random* connectivity: we start from a full tree graph, where sibling nodes are fully connected, then drop edges at random with a probability of 0.5, but ensure that the resulting graph is connected. The spring lengths are: 0.4 between the root and intermediate nodes, 0.1 between intermediate and leaf nodes, 0.65 within intermediate node siblings and 0.2 within leaf node siblings. All springs have the same constant, except for springs between leaf node siblings, which have a value that is half the value of other constants. In total we generate a dataset of $5 \cdot 10^5$ training, 10^5 validation, and 10^5 test sequences, each 50 frames long.

A.1.2 Human3.6M

This dataset (Ionescu et al., 2013) consists of 3.6 million 3D human poses composed of 32 joints and corresponding images taken from 11 professional actors in 17 scenarios. For training, we use subjects number 1, 5, 6, 7, and 8, and test on subjects number 9 and 11. In total we create 10k training and 3.5k test sequences of 50 frames. We use the provided 2D pose projections to render 12 joints in total (3 of each limb).

A.2 Training Details

All models are trained with Adam (Kingma & Ba, 2015) using default parameters and a learning rate of $5 \cdot 10^{-4}$. We use a batch size of 32 and train models for 100 epochs. On the visual datasets we train each model in two stages, which acts as a curriculum: first we train the visual encoder and decoder on a reconstruction task, afterwards we optimize the dynamics parameters on the prediction task. This acts as a curriculum for the visual modules, where they can first focus on inferring a separate representation for parts and objects in the provided (hierarchical) spatial slots. Once the visual modules converged, it made almost no difference whether we also fine-tuned their parameters on the prediction task.

Optimizing only for the next step prediction task can lead to a predictor which ignores the inferred relational graph (also noted in Kipf et al. (2018)). To avoid this, we have a "burn-in-phase" at the beginning of the sequence, where we feed the ground truth input, and predict the last 10 steps of the sequence by feeding the model's prediction as next step input. To train the models we optimize the ELBO for these longer prediction sequences, whereas we evaluate the models only on *next step* prediction. For the output distribution we use a fixed variance $\sigma^2 = 5 \cdot 10^{-5}$.

Reported results in the bar plots are the mean and standard deviation obtained for each model using 3 different random seeds. The reported negative log likelihood loss is averaged over the number of objects for states or pixels in the visual case. The "normalized" negative log likelihood in Figures 2a and 4a is inversely proportional to a version of HRI that operates on the ground-truth interaction graph (HRI-GT) (higher value is better). This allows us to factor out the complexity of the task and make it easier to compare results *between* tasks. All models were stable during training and runs with different seeds produced results with low variance.

A.3 Architecture Details

A high-level overview of the HRI model is presented in Figure 1, with a high-level summary of all components. Below we describe each component in detail.

Visual Encoder The visual encoder takes as input the concatenation of $32 \times 32 \times 3$ RGB frame and a $32 \times 32 \times 2$ (x, y)-fixed coordinate channels (as in Liu et al. (2018); Watters et al. (2019)) and outputs a hierarchy of object representations. First it infers the 4×4 leaf objects, from which 4 intermediate nodes and 1 root node are inferred. This results in 16 leaf objects, 4 intermediate objects, and one root object, each 48-dimensional. They are all mapped with a FC layer (with shared weights) to 16-dimensional vectors. Detailed architecture is presented in Table 1 (but note a different variant used for Human 3.6M experiments in Table 6).

Table 1: Visual encoder architectures

8×8 conv, 48 ReLU units, stride 8, batch norm
2×2 conv, 48 ReLU units, max pool 2, batch norm
2×2 conv, 48 ReLU units, max pool 2, batch norm
FC, 16 ReLU units. (applied slot-wise)

Relational Inference Module For relational inference (for architecture details see Table 2) we use the 'encoder' of NRI (Kipf et al., 2018), that takes as input a set of object trajectories (states) and infers their pairwise relations (edges) using a Graph Neural Network (GNN) (Battaglia et al., 2018). In contrast to NRI (which assumes a static graph), we assume a *dynamic* interaction graph, which is necessary since objects move across the image and may end up in different spatial slots throughout the sequence. This is achieved by inferring edges at each time step based on a history of $k = 10$ most recent object states. The NRI 'encoder' receives as input a sequence of object state trajectories $\mathbf{o} = (\mathbf{o}^1, ..., \mathbf{o}^T)$, which in our case are inferred. It uses GNN that performs two stages of message passing to infer relations, where the initial node representations $\mathbf{o}_i$ are obtained by concatenating the corresponding object states across a window of size $k = 10$.

$$\begin{aligned} \mathbf{o}'_j &= f_o^1(\mathbf{o}_j), \\ \mathbf{e}'_{(i,j)} &= f_e^1([\mathbf{o}'_i, \mathbf{o}'_j]), \\ \mathbf{o}''_j &= f_o^2(\textstyle\sum_{i \neq j} \mathbf{e}'_{(i,j)}), \\ \mathbf{e}''_{(i,j)} &= f_e^2([\mathbf{o}''_i, \mathbf{o}''_j]), \\ f_\phi(\mathbf{o}^{t-k:t})_{ij} &= \mathbf{e}''_{(i,j)} \end{aligned}$$

where ϕ contains the parameters of the message-passing functions, which are simple MLPs, and o', e' and o'', e'' are node- and edge-embeddings after first and second message passing operations respectively.

To backpropagate through the sampling from $q_\phi(\mathbf{r}_{ij}|\mathbf{o})$, NRI uses a continuous approximation of the discrete distribution to obtain gradients via the reparameterization trick (Maddison et al., 2017; Jang et al., 2017). In this case, samples are drawn as $\mathbf{r}_{ij} = \text{softmax}((\mathbf{e}_{(i,j)} + \mathbf{g})/\tau)$ where $\mathbf{g}$ is drawn from a $\text{Gumbel}(0, 1)$ distribution and $\tau = 0.5$ is the temperature parameter.

Dynamics Predictor The dynamics predictor takes as input inferred object states of dimensionality d and the inferred pairwise edges and predicts the object states at the next time step via message passing (Table 3). The hierarchical message passing functions f_{MP}^{bu}, f_{MP}^{ws}, and f_{MP}^{td} perform a single node-to-edge and edge-to-node message passing operation, where their node-to-edge and edge-to-node MLPs all share the same set of weights.

Table 2: Relational inference module architecture

Node-embeding MLP
Concatenate K object states in a slot-wise manner FC, 64 ELU FC, 64 ELU, batch norm
Concatenate object pairs slot-wise $o_ij = [o_i, o_j]$
Node-to-edge MLP f_{n2e}
FC, 64 ELU FC, 64 ELU, batch norm
Edge-to-node MLP f_{e2n}
FC, 64 ELU FC, 64 ELU, batch norm
Append slot-wise the skip connection of o_ij
Node-to-edge MLP (shared weights with f_{n2e}**)**
FC, 64 ELU FC, 64 ELU, batch norm
Output MLP f_o
FC, 64 ELU FC, 64 ELU, batch norm FC, 2 output units

Table 3: Dynamics predictor architecture

Bottom-up message passing round f_{MP}^{bu} (on child-parent edges)
Node-to-edge MLP f_{n2e} FC, 64 ReLU FC, 64 ReLU, batch norm **Edge-to-node MLP** f_{e2n} FC, 64 ReLU FC, 64 ReLU, batch norm
Within-siblings message passing round f_{MP}^{ws} (on sibling edges) Shared weights of f_{n2e} and f_{e2n} MLPs with f_{MP}^{bu}
Top-down message passing round f_{MP}^{td} (on parent-child edges) Shared weights of f_{n2e} and f_{e2n} MLPs with f_{MP}^{bu}
LSTM
LSTM, 64 hidden units
Output MLP f_o
FC, 64 ReLU FC, d output units.

Visual Decoder The Visual decoder takes as input a set of N $d = 16$-dimensional object states and produces the output image according to the architecture in Table 4. A unique float index $i \in [0, 1]$ is appended to each object state, which helps learning the visual object colors, as they are decoded separately as a set and then summed (permutation invariant). Note a different decoder variant we used for Human 3.6M where all states are decoded together as in a standard convolutional decoder (detailed architecture in Table 6).

Table 4: Visual decoder architecture

FC, $4 \times d$ ReLU
4×4 convTranspose, 64 ReLU, stride 2
4×4 convTranspose, 64 ReLU, stride 2
4×4 convTranspose, 64 ReLU, stride 2
4×4 convTranspose, 3 ReLU, stride 2

A.4 Ablations

Following are the ablation-specific configurations:

- HRI-GT: HRI model that receives the ground truth graph as the input to the dynamics predictor (no relational inference).
- HRI-H: HRI model that performs relational inference on a smaller subset of edges (other edges are excluded), by considering the convolutional and pooling operations that infer the hierarchical object slots. Let $o1$ be the root object, $o2, o3, o4, o5$ intermediate objects, and $o6 - o21$ leaf objects. The subset of edges HRI-H considers is: parent-child (and vice-versa child-parent) edges $(1{-}2, 1{-}3, 1{-}4, 1{-}5)$, $(2{-}6, 2{-}7, 2{-}8, 2{-}9)$, $(3 - 10, 3 - 11, 3 - 12, 3 - 13)$, $(4 - 14, 4 - 15, 4 - 16, 4 - 17)$ and $(5 - 18, 5 - 19, 5 - 20, 5 - 21)$ and all within-sibling edges.
- NRI-GT: NRI model that receives ground truth graph as the input to the dynamics predictor (no relational inference).
- FCMP: NRI model that performs message passing in the dynamics predictor on a fully connected graph (no relational inference).

A.5 NRI baseline

To infer the object states on which NRI performs relational inference we use the visual encoder and decoder of the HRI architecture. This ensures a fair comparison between NRI and HRI in the visual setting. We emphasise that standard NRI as presented in Kipf et al. (2018) did not support learning from visual images.

The dynamics predictor ('decoder' in NRI (Kipf et al., 2018)) is presented in Table 5, which uses an LSTM (Hochreiter & Schmidhuber, 1997) instead of the GRU (Cho et al., 2014) cell.

Table 5: NRI dynamics predictor

Node-to-edge MLP f_{n2e}
FC, 64 ReLU FC, 64 ReLU, batch norm
Edge-to-node MLP f_{e2n}
FC, 64 ReLU FC, 64 ReLU, batch norm
LSTM
LSTM, 64 hidden units
Output MLP f_o
FC, 64 ReLU FC, d output units.

A.6 LSTM baseline

Similarly to NRI baseline, the LSTM baseline uses the same (pretrained) visual encoder and decoder to map from image to object states, and vice-versa. We use an LSTM with 64 hidden units that concatenates representations from all objects and predicts their future state jointly. Essentially, the NRI baseline dynamics predictor can be viewed as extending the LSTM by adding the message passing part (functions f_{n2e} and f_{e2n}) based on the inferred interaction graph. In contrast, the LSTM baseline only explicitly considers the nodes of the graph, but not its edges (relations).

B Additional Results

B.1 Springs datasets

In order to test whether HRI can handle different amounts of objects, we experiment with 3-3-state-springs and 3-3-visual-springs datasets. The results in Figure 5 and Figure 4b for 3-3-state-springs provide the same conclusions as for 4-3-state-springs, with the exception of Figure 5a where HRI and HRI-H perform the same (for 4-3-state-springs HRI outperforms HRI-H, see Figure 2a). We also provide results and visualizations for 3-3-visual-springs. From inspecting the decoded images corresponding to each of the learned slot-based representations (Figure 7b), we are able to observe that they frequently correspond to individual objects. When too many slots are provided some of them become empty, which demonstrates the flexibility of this type of encoder. Similar to before, in Figure 5b we find that HRI outperforms all baselines as for 4-3-visual-springs.

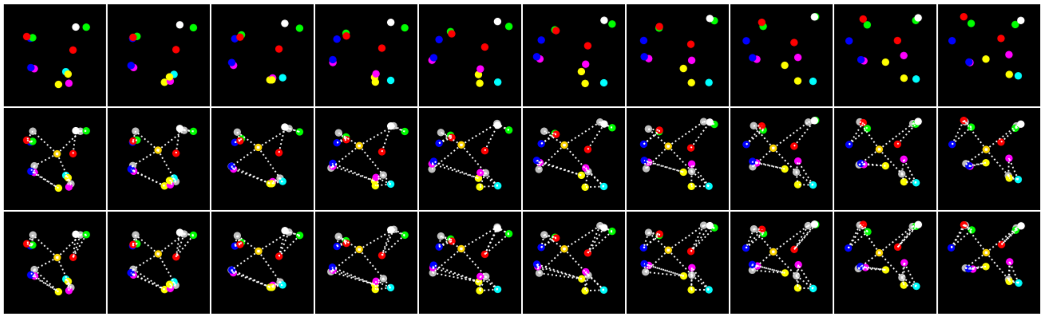
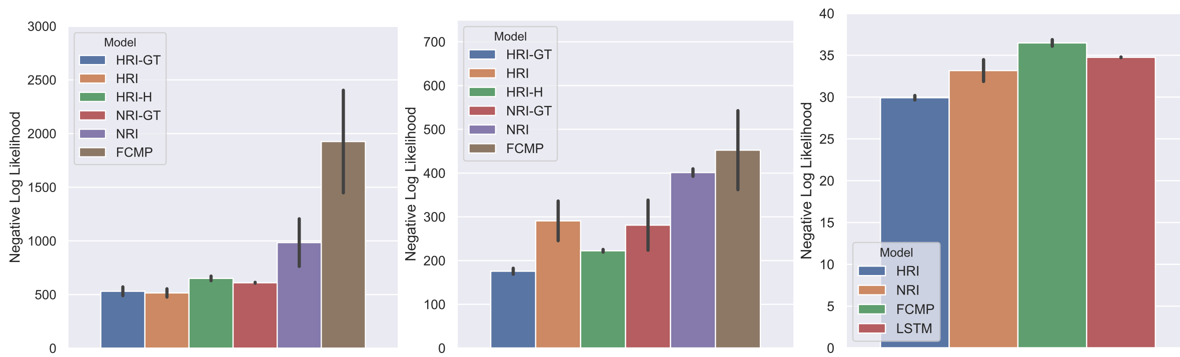


Figure 3: Rendered input sequence of 4-3-state springs, showing leaf objects only (top); showing all objects and the interaction graph (not observed by the model) (middle); Predictions and inferred edges by HRI (bottom).



(a) (b) (c)

Figure 4: Performance by HRI and baselines in terms of the negative log likelihood (lower is better) on the 4-3-state-springs (a), 3-3-state-springs (b) and on the Human 3.6M dataset (c).

Finally, we report unnormalized scores (i.e. plain negative log likelihoods) for 4-3-state-springs and 3-3-state-springs in Figures 4a and 4b.

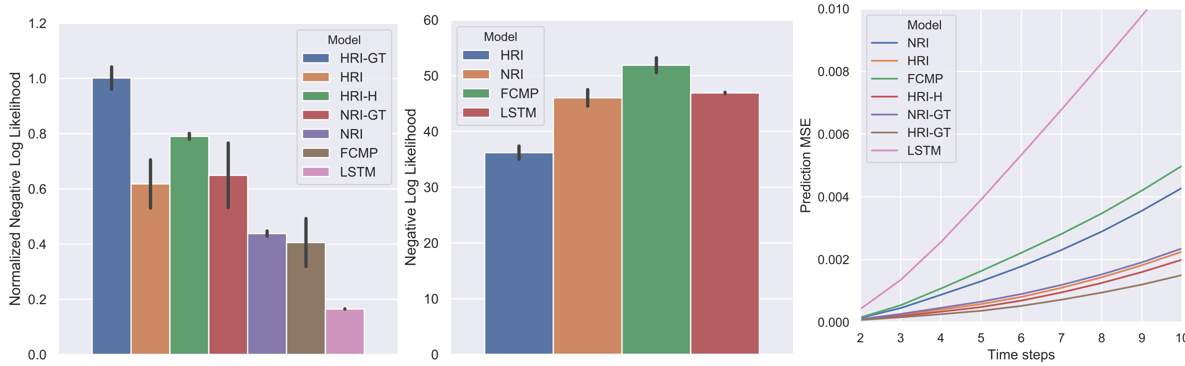
B.2 Human 3.6M

As a final benchmark, we consider the rendered joints of the *Human3.6M* dataset. This task is significantly more complex, since the underlying system dynamics are expected to vary over time (i.e. they are non-stationary). For this reason we adapted the visual decoder and correspondingly the encoder as well (architectures in Table 6). In this decoder variant, which we call *ParDec*, all states are decoded together as in a standard convolutional decoder. As a result of decoding all object slots together *ParDec* is less interpretable than *SlotDec*, but computationally more efficient and potentially more scalable to real-world datasets since it does not make strong assumptions about how information about objects should be combined. This may also make it easier to handle background, although this is not explored.

Figure 4c demonstrates the performance of HRI and several baselines on this task. Note that HRI is the best performing model, although the gap to NRI and LSTM is further reduced. This can be explained by the fact that many of the human motions, such as sitting, eating, taking a photo, and waiting involve relatively little motion or only motion in a single joint (and thereby lack hierarchical interactions). Example future predictions by HRI (10 steps) for this dataset can be seen in Figure 8. Their quality is similar to the one of our results for visual springs.

Table 6: Human 3.6M visual modules architecture

Encoder
8×8 conv, 48 ReLU units, max pool 2, batch norm
8×8 conv, 48 ReLU units, max pool 2, batch norm
8×8 conv, 48 ReLU units, max pool 2, batch norm
2×2 conv, 48 ReLU units, max pool 2, batch norm
2×2 conv, 48 ReLU units, max pool 2, batch norm
FC, 16 ReLU units.
ParDec decoder
4×4 convTranspose, 64 ReLU, stride 2
4×4 convTranspose, 64 ReLU, stride 2
4×4 convTranspose, 3 ReLU, stride 2



(a) (b) (c)

Figure 5: Performance by HRI and baselines on the 3-3-state-springs (a) and 3-3-visual-springs (b). For (a) we report the "normalized" negative log likelihood which is inversely proportional to HRI-GT (higher is better). For (b) we report negative log likelihood (lower is better). (c) MSE when predicting into the future on 3-3-state-springs (prediction rollouts).

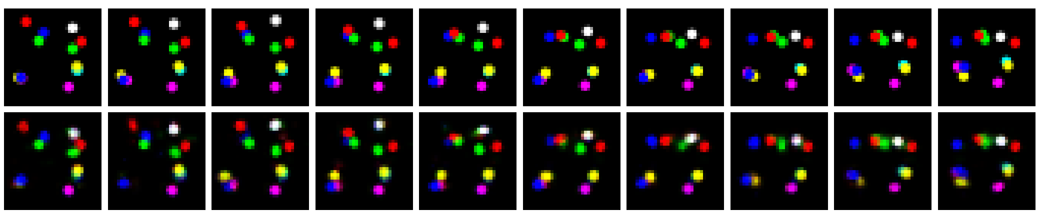
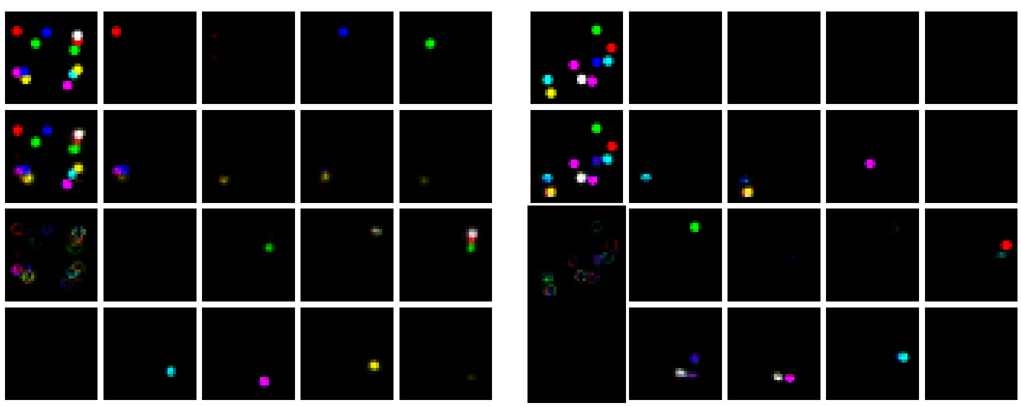


Figure 6: Ground truth (top) and predicted (bottom) 10 time steps rollout of HRI on 4-3-visual-springs.



(a) (b)

Figure 7: HRI introspection for 4-3-visual springs (a) and 3-3-visual-springs (b). The first column of each plot contains the input image, predicted image and the difference between both. In the other 4 columns we visualize 16 object slots decoded separately.

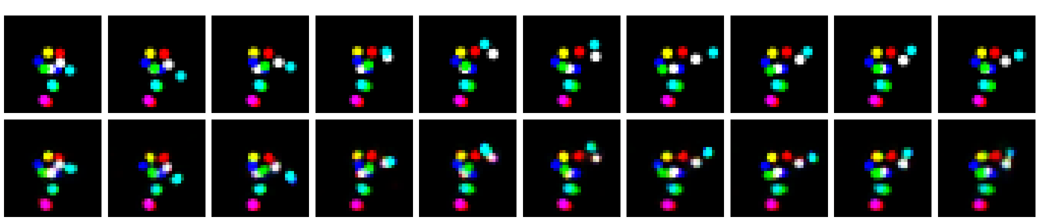


Figure 8: Ground truth (top) and predicted (bottom) 10 time steps rollout of HRI on Human3.6M dataset.