A Neural Architecture for Representing and Reasoning about Spatial Relationships

Eric Weiss, Brian Cheung & Bruno A. Olshausen
Redwood Center for Theoretical Neuroscience
University of California, Berkeley
Berkeley, CA 94720, USA
{eaweiss,bcheung,baolshausen}@berkeley.edu

Abstract

We explore a new architecture for representing spatial information in neural networks. The method binds object information to position via element-wise multiplication of complex-valued vectors. This approach extends Holographic Reduced Representations by providing additional tools for processing and manipulating spatial information. In many cases these computations can be performed very efficiently through application of the convolution theorem. Experiments demonstrate excellent performance on a visuo-spatial reasoning task as well as on a 2D maze navigation task.

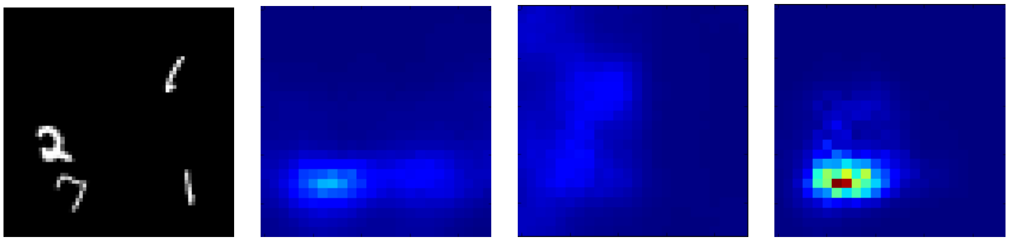
1 Introduction

Many problems faced by intelligent agents involve some degree of spatial reasoning. This work describes a new framework for artificial neural networks that allows them to represent information embedded in space and to efficiently draw inferences using that information. One way to represent a collection of objects embedded in space is to simply form a list of tuples, with each tuple pairing the identity or properties of an object with its position, i.e., (object, location). The goal of this work is to develop a memory system that can model this type of representation and is suitable for use as a component of neural networks, meaning that it must be differentiable and computationally efficient.

The method presented in this work is based on Holographic Reduced Representations or HRRs, described in Plate (2003). These associative memory systems can represent a variety of data structures, including sequences, lists of associated items, and hierarchical structures. They also support various operations for efficient storage and retrieval of information contained within the system. One way to formulate HRRs utilizes complex vectors as the fundamental elements of representation. As was shown by Kanerva (2009) and more recently by Google DeepMind in Danihelka et al. (2016), this approach is suitable for use in neural networks on a variety of learning tasks. Here we present a continuous mapping from positions to complex vectors. This enables us to utilize the framework of HRRs to represent collections of objects in space as complex vectors.

For encoding locations within a space of dimension d as a complex vector of dimension n, with $n >> d$, we use the formula $\ell(r) = e^{iL \cdot r}$, where r is a d-dimensional vector specifying position, L an n-by-d dimensional matrix, and i the imaginary unit. L is chosen such that the vector encodings of two different positions will be nearly orthogonal unless the original positions are very close. In this work, we choose L such that the encoded location vector $\ell(r)$ is the Discreet Fourier Transform of a (discrete) delta function centered at r. This choice allows us to switch between Fourier and spatial representations, so that we may utilize the Convolution Theorem to efficiently compute convolutions and products when needed.

To bind object information, represented as a complex vector c, to a position, we simply take the element-wise product with the corresponding complex location vector. A list of k object-position pairs can then be represented as a single complex vector m by summation: $m = \sum_{j=1}^{k} c_k \odot \ell(r_k)$ with $\odot$ denoting element-wise multiplication. For more information about HRRs based on complex vectors, please refer to Danihelka et al. (2016). The rest of this text describes two experiments that utilize our spatial memory system.



(a) Example image (b) ”below a 2” (c) ”to the left of a 1” (d) Combined

Figure 1: Query constraint satisfaction maps visualized for a training example. For this image, the query is ”below a 2 and to the left of a 1”. The correct answer is therefore ”7”.

1.1 Experiment 1: Representing and Retrieving Image Information

This section describes an experiment that is based on previous work involving recurrent neural networks with selective visual attention, as in Gregor et al. (2015). These networks control a virtual retina which they use to gather image information over a series of glimpses. On each glimpse, image information sampled by the virtual retina is fed as input to a recurrent neural network. The state is updated and the network outputs the coordinates of the next glimpse. The retina is then moved to these new coordinates and another glimpse is processed. Finally, after some predetermined number of glimpses the network must provide an answer to some supervised learning task. While one would hope that such a network would effectively learn to combine all the glimpses into a unified description of the image, due to the difficulty in understanding neural network function it can be hard to tell whether or not this is the case. Additionally, while many recurrent networks such as LSTMs are designed to be quite general, nothing about their design suggests that they are particularly well suited for representing spatial structure.

One possible solution to these issues is to bind visual and spatial information using the framework introduced in section 1. All we need to do is to convert the image information collected during a glimpse as well as the position of that glimpse into complex-valued vectors. Then we multiply these vectors element-wise, effectively encoding the glimpse as a (content, position) tuple. Instead of learning a highly nonlinear recurrence function, as is the norm in recurrent networks, we can simply sum these vectors over the sequence of glimpses to encode a model of the image as a singe complex vector. We will refer to this vector as the scene vector. This simple, fixed recurrence function significantly lowers the number of free parameters to be learned, and thanks to the fact that all of the required operations are just (complex) multiplications or additions, it is fairly cheap to implement.

In this experiment we test a simple recurrent network based on this idea on a toy dataset. The algorithm’s goal is to find and identify a target object among several objects in an image. The input data consists of images along with questions about those images, which we will call queries. Each query specifies the position of the target object by listing the identities of some nearby objects, as well as their relative spatial relationships to the target object. The spatial relationships are specified only approximately, as would be the case in common language. Thus, a query might read ”an object above the table and to the left of the plate”, to which the correct answer might be ”cup”.

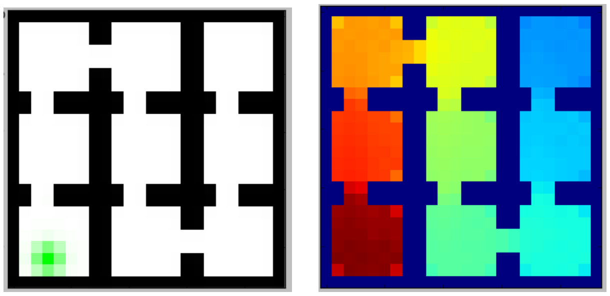
Using the framework outlined in the previous section as well as some basic operations from HRRs it is possible to efficiently combine a query with a scene vector in order to obtain an answer, also in the form of a complex vector. The method applies the convolution theorem in order to compute a product of functions - in this case, each function represents the degree of satisfaction for one of the constraints specified in the query. The product of all of these functions then provides an estimate of the degree of satisfaction of all of the constraints in the query.

Each example in our toy dataset is constructed by selecting several digits from the MNIST dataset at random and placing them at random locations in a 64-by-64 image. Queries are then generated automatically by selecting a target digit and two of its closest neighbors. The spatial relationships of the target digit to its neighbors are then computed as a function of their relative positions. These are each then binned into one of eight possible cases, corresponding to ”above”, ”above and to the left”, ”to the left”, and so on. A representative example image from this dataset can be found in figure 1a. The query for this particular image is ”below a 2 and to the left of a 1”. So, the answer is ”7”.

After encoding the image as a scene vector through multiple glimpses, we derive multiple constraint satisfaction maps from scene vector using the components of the query. In this case there are two of them. Each of these returns a 2D spatial map (corresponding to image space) representing which locations satisfy the constraint in question. The conversion into the spatial domain is implemented through multiplication of the Inverse DFT matrix corresponding to the chosen location encoding parameters L. Figure 1b shows the resulting map for ”below a 2”, and figure 1c shows the map for ”to the left of a 1”. We then take the point-wise product of these two maps and normalize, resulting in the map visualized in figure 1d. As hoped, the majority of the weight is located at the position of the 7 in the image, which is the correct answer for this example. The final step is to retrieve the object identity information contained in the scene vector at the location specified by this map. To do this we convert this final map into a complex vector encoding location. This is achieved through multiplication by the DFT matrix. Then we multiply the scene vector by the inverse of this location vector (for location vectors this happens to be its complex conjugate). The resulting vector should then contain an ”unbound” vector representing ”7”, allowing the network to provide the correct answer. After training has converged, the algorithm achieves about 95 percent accuracy on the artificial multi-MNIST dataset described in this experiment.

1.2 Experiment 2: Path Planning

Our framework can also be used to solve a simple navigation problem. In this experiment, a 2D map of obstacles and a reward function are stored within a complex vector. Actions, also represented as complex vectors, act on these maps to produce translations. It is possible to map the concepts discussed previously onto the value iteration algorithm from reinforcement learning, providing a way to calculate the optimal action given the current state and reward function. The operations are very similar to those outlined in the first experiment, making use of the convolution theorem to reduce computational cost. The only difference is that instead of reducing over spatial positions using a product, we reduce using a max function. The maze, reward function, and computed value function are shown in figure 2. The optimal action while in a state is to simply move to the neighboring state with the highest value. It is clear in figure 2b that following such a strategy will lead to the reward state.



(a) Environment map (b) Value function

Figure 2: Obstacle/reward map and calculated value function described in the path planning experiment. In (a), black indicates walls and green indicates high reward. In (b), red indicates high reward, while blue indicates low reward.

2 Conclusion

We have introduced a novel way to bind positions to object information within an HRR using complex vectors. The method allows various types of computations to be performed efficiently by leveraging some properties of DFTs, and can be easiliy integrated into existing neural net architectures as it is differentiable, computationally efficient, and contains relatively few free parameters. The utility of these properties is verified on a visual reasoning task, as well as on a simple navigation task.

ACKNOWLEDGMENTS

We would like to acknowledge everyone at the Redwood Center for their helpful discussion and comments. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPUs used for this research.

REFERENCES

Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. Associative long short-term memory, 2016.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation, 2015.

Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, 2009. doi: 10.1007/s12559-009-9009-8. URL http://dx.doi.org/10.1007/s12559-009-9009-8.

Tony A. Plate. *Holographic Reduced Representation: Distributed Representation for Cognitive Structures (Lecture Notes)*. Center for the Study of Language and Inf, 2003. ISBN 1575864304.