Spatiotemporal Sequence Memory for Prediction using Deep Sparse Coding

Edward Kim
Villanova University
Villanova, Pennsylvania
edward.kim@villanova.edu

Edgar Lawson
Naval Research Laboratory
Washington, DC
ed.lawson@nrl.navy.mil

Keith Sullivan
Naval Research Laboratory
Washington, DC
keith.sullivan@nrl.navy.mil

Garrett T. Kenyon
Los Alamos National Laboratory
Los Alamos, New Mexico
gkenyon@lanl.gov

ABSTRACT

Our brains are, "prediction machines", where we are continuously comparing our surroundings with predictions from internal models generated by our brains. This is demonstrated by observing our basic low level sensory systems and how they predict environmental changes as we move through space and time. Indeed, even at higher cognitive levels, we are able to do prediction. We can predict how the laws of physics affect people, places, and things and even predict the end of someone's sentence.

In our work, we sought to create an artificial model that is able to mimic early, low level biological predictive behavior in a computer vision system. Our predictive vision model uses spatiotemporal sequence memories learned from deep sparse coding. This model is implemented using a biologically inspired architecture: one that utilizes sequence memories, lateral inhibition, and top-down feedback in a generative framework. Our model learns the causes of the data in a completely unsupervised manner, by simply observing and learning about the world. Spatiotemporal features are learned by minimizing a reconstruction error convolved over space and time, and can subsequently be used for recognition, classification, and future video prediction. Our experiments show that we are able to accurately predict what will happen in the future; furthermore, we can use our predictions to detect anomalous, unexpected events in both synthetic and real video sequences.

1 INTRODUCTION

Consciously and unconsciously, we are continually predicting what will happen next. In fact, many believe that prediction plays a more central role, or could even be one of the primary roles of the brain. For example, take a moment to look around you. Using a combination of your senses including sight, audition, smell, and touch, you are able understand what is happening in your immediate environment. Now close your eyes and think about what will happen when you open your eyes a second later. Do you believe (predict) your environment will stay the same?

"Prediction is the essence of intelligence" says Yann LeCun - a pioneer of deep neural networks. Jeff Hawkins, founder of the Redwood Center for Neuroscience, proposed that prediction is intelligence framed by understanding. Indeed, we as humans are predicting all the time, ranging from our visual system's ability to expect that our environment will stay relatively constant between blinks, to our auditory system cringing as it awaits a loud bang when a child pushes a pin into a balloon, to our language centers predicting and finishing the end of people's sentences. Perception and understanding are governed by predictions made by the internal models of our brain. Our ability to predict and anticipate the future helps in our understanding of the world around us including understanding intention, planning, and finding anomalies.

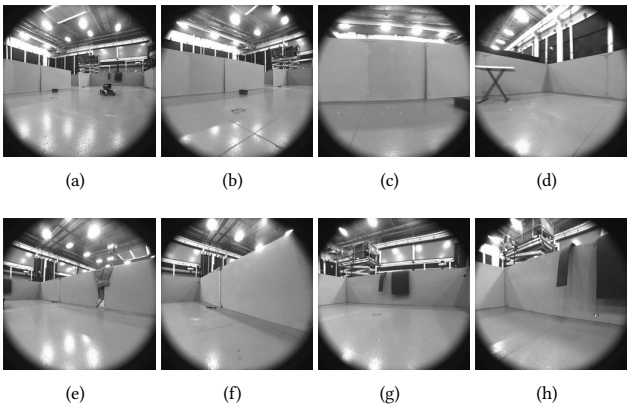


Figure 1: Frames of a video captured from a Endeavor Robotics PackBot 510 at the Naval Research Laboratory. This image sequence is used in our experiments to train the spatiotemporal sequence dictionary used for prediction.

In this work, we explore an artificial neural network model that is able to mimic low level predictive behavior in a computer vision system. Our model representations are generated using sparse coding, a mathematical technique to create a numerical code where many of the elements are not active, e.g. zero. This representation is consistent with evidence that shows the neural code represented in biological perception is also sparse; in fact, a large population of cortical neurons are "silent", e.g. they spike rarely or do not spike at all [10].

Our experiments were performed on both synthetic datasets and real world datasets gathered in complex environments. Our results show that predictive sequence memories using sparse coding can be learned in a completely unsupervised manner. Furthermore, we demonstrate that the spatiotemporal sequence memories can be

used to predict the immediate future in low level vision systems. By computing and analyzing reconstruction error, e.g. the difference between the predicted outcome and the actual outcome, we are able to detect anomalous events. For our real world data, we collected video data from a small, autonomous robotics platforms and analyzed the robot video around several environments at the Laboratory for Autonomous Systems Research (LASR), see Figure 1. Our quantitative and qualitative results show that predictive modeling gives our vision system the ability to learn what to expect as it perceives both space and time.

2 BACKGROUND

2.1 Background in Spatiotemporal Sequence Memory

There is strong evidence that the spatiotemporal sequence memories exist in both high and low levels of biological perception. For example, Gavornik and Bear [7] discovered that repeated presentations of a visual sequence over a course of days causes evoked response potentiation in mouse V1 that is highly specific for stimulus order and timing. In other words, cortical activity in V1 regenerates a full sequence even when individual stimulus elements are omitted. This explains how the brain is able to make intelligent guesses or fill in details from partial visual information.

In fact, the whole cortex is continuously processing streams of sensory data to build a spatiotemporal model of the environment. Mauk and Buonomano [11] show through neuroimaging studies that there is an intricate link between temporal and spatial information in most sensory and motor tasks. Brosch and Schreiner [4] show that sequence-sensitive neurons in the auditory cortex are common and involved in the cortical representation of spectrotempral patterns of acoustic signals. Additionally, the brain is not a passive pattern matching machine, but rather a prediction machine, constantly generating and approximating the future. Bar [2] suggests that perceptual information activate linked stored representations, a kind of analogy that provides focused predictions.

Research by Jeff Hawkins' group have created a theoretical framework for sequence learning in the cortex called hierarchical temporal memory (HTM). HTMs are models that are able to continuously learn a large number of variable order temporal sequences using an unsupervised Hebbian-like learning rule. The sparse temporal codes formed by the model can robustly handle branching temporal sequences by maintaining multiple predictions until there is sufficient disambiguating evidence [5]. Our work has similar properties to the presented biological and theoretical frameworks; however, we utilize sparse coding to represent the input signals and learn the spatiotemporal sequence memories.

2.2 Background in Sparse Coding

Strong evidence demonstrates that the neural code is both explicit and sparse [6] where neurons fire selectively to specific stimuli. Olshausen [12] has shown that sparsity is a desirable property as our natural environment can be described by a small number of structural primitives. Sparse codes have a high representational capacity in associative memory, far surpassing the number of input-output pairs that can be stored by a more dense code [3], and that biologically, the sparsity of neural codes are more metabolically efficient and reduce the cost of code transmission [1]. Since the evidence shows that the neural code represented in biological perception is sparse, our model is governed by the principle of sparsity.

Mathematically, sparse coding is a reconstruction minimization problem where many of the elements are not active, e.g. zero. Sparse coding falls under the class of unsupervised methods as the objective function minimizes reconstruction error. The goal is to find a set of basis vectors e.g. dictionary elements, such that we can represent some input signal as a linear combination of these vectors. Specifically, we have some input variable $x^{(n)}$ from which we are attempting to find a latent representation $a^{(n)}$ (we refer to as "activations") such that $a^{(n)}$ is sparse, e.g. contains many zeros, and we can reconstruct the original input, $x^{(n)}$ as well as possible. Mathematically, a single layer of sparse coding can be defined as,

$$\min_{D} \sum_{n=1}^{N} \min_{a^{(n)}} \frac{1}{2} \|x^{(n)} - Da^{(n)}\|_2^2 + \lambda \|a^{(n)}\|_1 \tag{1}$$

Where D is the overcomplete dictionary, and $Da^{(n)} = \hat{x}^{(n)}$, or the reconstruction of $x^{(n)}$. The λ term controls the L1 sparsity penalty, balancing the reconstruction versus sparsity term. L1 regularization is used since it drives the coefficients of activity exactly to zero. N is the total training set, where n is one element of training. D represents a dictionary composed of small kernels that share features across the input signal. Sparse coding is a generative model that is overcomplete, performing dimensionality expansion in the hidden layer, while also forcing the coefficients to be sparse. This means the system is underdetermined and there exists an infinite number of possible solutions to the problem and will have to be solved through optimization. The criterion of sparsity resolves the degeneracy introduced by the overcomplete basis. In fact, the process of optimization is required to obtain the coefficients at inference time and is thus, computationally more expensive compared to typical feedforward architectures.

3 METHODOLOGY

For our framework, we use the Locally Competitive Algorithm (LCA) [13] to solve the sparse coding problem. The LCA algorithm is a biologically informed sparse solver that utilizes thresholding, excitatory, and inhibitory connections between neurons. LCA minimizes the mean-squared error (MSE) with sparsity cost function as described in Equation 1. The LCA model is governed by dynamics that evolve the neuron's internal state when presented with some input image. The internal state, i.e. "membrane potential", charges up like a leaky integrator and when it exceeds a certain threshold, will activate that neuron. This activation will then send out inhibitory responses to units within the layer to prevent them from firing. The input potential to the state is proportional to how well the image matches the neuron's dictionary element, while the inhibitory strength is proportional to the activation and the similarity of the current neuron and competing neuron's convolutional patches, forcing the neurons to be decorrelated.

The full model we developed is a highly recurrent multimodal network based upon deep sparse coding [9, 14]. The deep sparse model was implemented using OpenPV[1]. OpenPV is an open source,

[1] https://github.com/PetaVision/OpenPV

object oriented neural simulation toolbox optimized for high-performance multi-core computer architectures. In the following sections, we describe and illustrate how we extended the framework to create spatiotemporal sequence memories. The underlying principles of sparsity are maintained within a highly recurrent network that integrates forward, lateral, and feedback connections.

3.1 Creating "Deep" Sparse Coding Networks

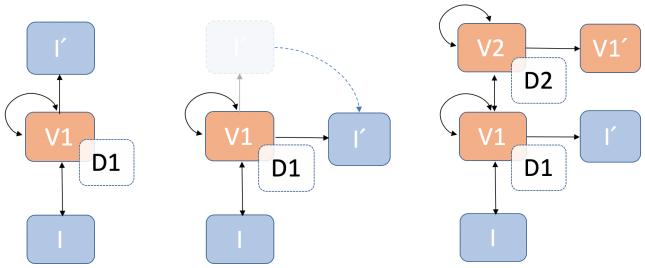
The term, "deep", is borrowed from the machine learning literature referencing deep learning. At a simplistic level, the depth refers to the fact that a network has multiple hidden layers between the input and output. The idea of depth has underpinnings in cortical function, where neural circuits and networks are hierarchical in nature. For example, in the visual cortex, we know that visual information passes through the retina, onward to the LGN, and then to the visual cortex. The visual cortex is structured into a hierarchy of visual areas V1, V2, V4, etc., where information does not simply move in a feedforward manner, but is highly recurrent, and integrating both feedforward and feedback information. At the low levels of the visual cortex, the neurons are tuned to simple visual characteristics such as orientation, spatial frequency, size, color, and shape [8]. At higher levels of the visual cortex, the neurons are more invariant to simple visual differences and tuned to more complex visual stimuli, often integrating combinations of low level features.

Our sparse coding model is structured to mimic the principle form and function of biological networks. We illustrate the process by which we can make each sparse coding level deeper and hierarchical in Figure 2. In Figure 2(a), we show a single sparse coding layer, where the input, I, is sparse coded by V1 using a learned dictionary, D1. In this example, I′ is the reconstruction of I. We "bend" the network (mathematically, this has no impact), Figure 2(b) in order to visually accommodate another layer of sparse coding on top of V1, see Figure 2(c). V2 performs a reconstruction of the membrane potential of V1 using a learned dictionary, D2. Since the entire hierarchy is a dynamic system solved iteratively, the top layers can inform the activity of the lower layers with top-down feedback.

3.2 Creating "Wide" Sparse Coding Networks

While the idea of depth is internal to the structure of the brain, the idea of width is much more observable. As we can clearly see, our perception system consists of multiple input streams. Thus our brains have to reconcile multiple information streams at once, including binocular vision, audition, somatosensory information, etc. We metaphorically represent different input streams as multiple inputs into a sparse coding layer. In this way, we are making a sparse coding network "wide" as illustrated in Figure 3.

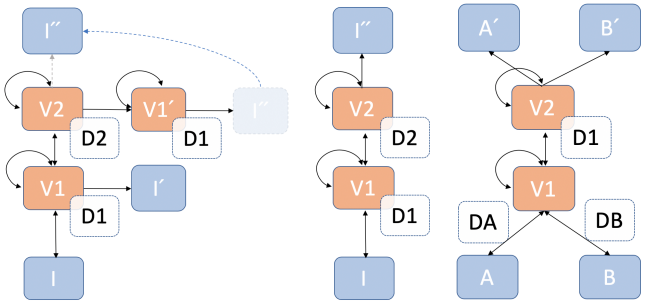
In Figure 3 (a), we show a complete two layer sparse coding network with full reconstructions, I′, V1′, and I′′. Because V2 is reconstructing V1′, the reconstruction in I′′ through V1′ is not exactly the same as I′, as there is inevitably some small error. In order to simplify the visualization of multi-layer sparse coding networks, we hide the reconstructions as show in Figure 3(b), even though they are still there and being used for the error computation. Finally, in Figure 3(c), we illustrate the addition of another input stream.



(a) Single sparse coding layer (b) Single sparse coding layer bend transition (c) Two layer sparse coding network, V2 stacked on top of V1

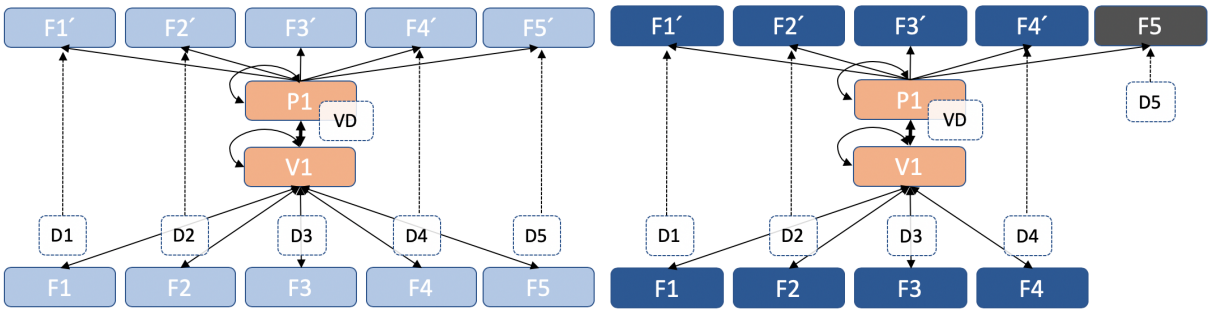
Figure 2: An illustration of how we transition from a single sparse coding layer (a), to a multi layer, hierarchical sparse coding network (c). The transition in (b) is only visual and has no computational effect on the model. Note that in (c), V2 is reconstructing the membrane potential of V1 using a learned dictionary D2.

The new input stream, B, utilizes its own learned dictionary, DB, but the sparse coding layer V1 must use the same sparse activation vector to represent both input streams, A and B. Because of this architecture, we can say that the dictionaries, DA and DB, are linked.



(a) Complete two layer sparse coding network, with a reverse bend transition (b) Equivalent representation to (a) (c) Addition of another input stream to the network

Figure 3: An illustration of how we transition from a deep sparse coding network into a multimodal deep sparse coding network. In (a) we show all of the components of the network and visually bend the reconstruction, I′′ to the top of the network (even though it is technically being passed through V1′). In (b) we show a simplified visualization where the reconstruction components are hidden. Finally, in (c) we show the addition of another input stream into the network. Each input stream has its own dictionary, but the sparse coding layer, V1, must share the same activation between input streams. This will enable the two input dictionaries to be linked.



(a) Sequential Train Process (b) Prediction Process

Figure 4: Configurations of the spatiotemporal deep sparse coding architecture. (a) is the training architecture and (b) is used for inference. The sequence memory is stored in the dictionary elements, D1-D5, and learned through a Hebbian like update from a residual reconstruction error. During inference, a truncated input is presented to the network which then optimizes for a sparse activation. Since the dictionaries D1-D5 are linked, we can utilize the sparse activations and D5 to predict the next frame in the sequence, F5.

3.3 The Spatiotemporal Model

Using the concepts of deep sparse coding networks and multimodality in a layer, we construct our final spatiotemporal model as shown in Figure 4. The model is structured as one input layer followed by two sparse coding layers. The input layer supports five streams of data that we map to image frames to support video. This introduces a conceptual difference from the traditional view of multimodality, since we treat time e.g. the frames of a video, as separate input streams. In specific terms, an input to our system is snapshot of five image frames, initialized t=1..5. The next input to the system would be another sequence of video frames, t=2..6, and would continue until the end of a video sequence. Presenting the data in this fashion will result in a learned spatial and temporal sequence within the model.

Dictionary Learning

In order to update the linked dictionaries, D1 - D5, we formulate the sparse coding problem as an energy minimization,

$$E = \frac{1}{2}\|x - Da\|_2^2 + \frac{1}{2}\lambda\|a\|_1 \tag{2}$$

Where the update to the dictionary, D, can be obtained by taking the gradient of the energy function with respect to D,

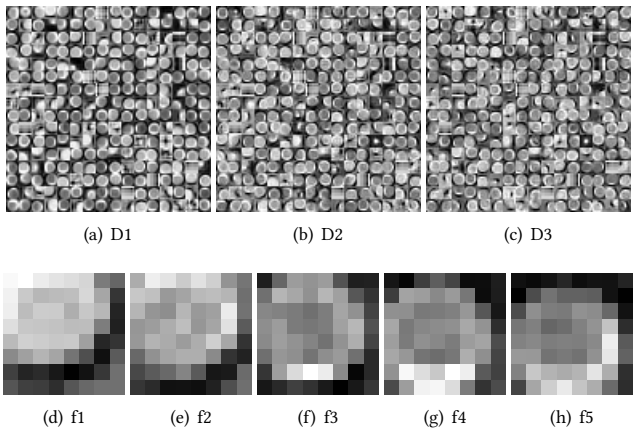
$$\Delta D \propto -\frac{\partial E}{\partial D} = a \otimes (x - Da) \tag{3}$$

This process can be achieved by first computing a sparse representation for a given input, x, and then computing the ΔD to reduce the reconstruction error given the sparse representation of the current input. The weight update is akin to a local Hebbian learning rule with pre and post-synaptic activities. For more details on the exact implementation of dictionary learning, see [14].

Prediction Process

The model is trained with the full structure shown in Figure 4 (a). During training, the weights are plastic and can be updated through dictionary learning. However, at prediction time, we fix the dictionary weights and make another minor adjustment. We omit the last input frame, F5, and perform inference on the truncated input. After optimizing for the sparse activations, e.g. dictionary coefficients, we can reconstruct input frames F1-F5, essentially predicting the next frame of the input sequence as shown in Figure 4(b).

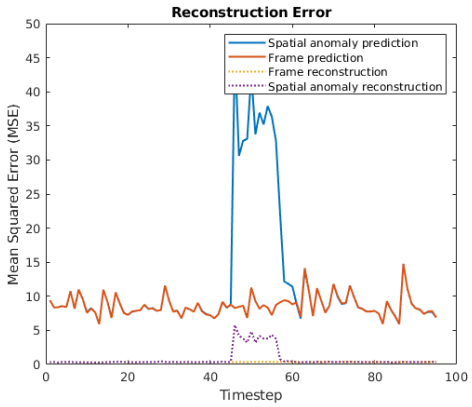
The key component to the prediction task is the fact that the dictionaries, D1-D5, are linked. During the training process, a single activation was used in Equation 3, shared across all input streams. Therefore the same activation represents sequential input frames in our model. We can use the activations that best match the truncated input and use this same activation vector to reconstruct the missing input. This is reminiscent to the previous work mentioned by Gavornik and Bear [7] where cortical activity in V1 regenerates a full sequence even when individual stimulus elements are omitted.



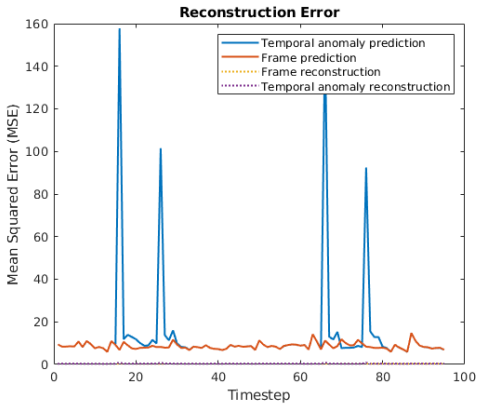
(a) D1 (b) D2 (c) D3

(d) f1 (e) f2 (f) f3 (g) f4 (h) f5

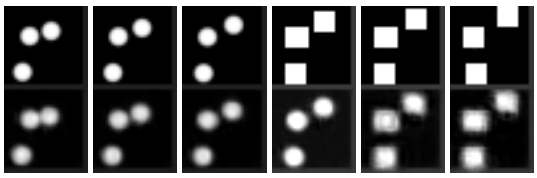
Figure 5: (a), (b), and (c) Show the dictionary D1, D2, and D3 learned from observing the synthetic video sequences. We expand on one of the sequences in (d)-(h) from a single dictionary element contained in D1-D5. This particular sequence memory shows an ball moving from the top left to the bottom right.



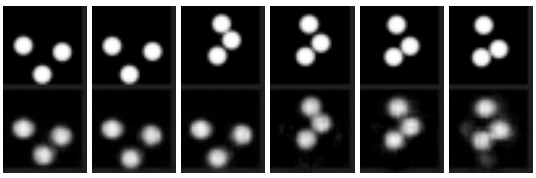
(a) Reconstruction error of spatial anomalies



(b) Reconstruction error of temporal anomalies



(c) (top) Input image sequence (bottom) Predicted reconstruction



(d) (top) Input image sequence (bottom) Predicted reconstruction

Figure 6: (a-b) Plots of reconstruction errors using prediction and reconstruction on normal data and anomalous data. (c-d) Visualization of the spatial and temporal anomalies. (c) Frames 47-53 of the bouncing ball dataset showing the change from a ball bouncing to a square bouncing. (d) Frames 64-69 in the temporal domain that display an abrupt change in ball position.

4 EXPERIMENTS AND RESULTS

For our experiments and results, we tested our spatiotemporal model on both synthetic and real world data. For both sets of experiments, we used the same architecture as shown in Figure 4. All of the experiments were run using the open source neural simulation toolbox, OpenPV. With the prediction model, we are interested in answering several questions. How well does the model predict the next frame? And to what extent do the predictions help us in other tasks, such as anomaly detection?

4.1 Synthetic Data

For our synthetic dataset, we selected a simple and well structured dataset called the bouncing balls dataset [15]. The data consists of videos where 3 white balls bouncing in a black box. The balls are constant in size, never merge, or disappear off the visual field. The videos are of length 100 frames with a resolution 32x32 pixels. Each training example is synthetically generated, so no training sequence is seen twice by the model.

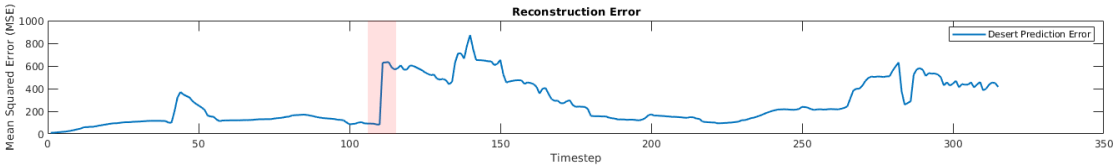
Using the synthetic dataset, we learned dictionary elements of size 8x8 with a dictionary size of 256 elements. Training was performed over 3000 image sequences and 25 epochs through the data. The learned dictionary elements are shown in Figure 5. We can compute the prediction error by MSE between the predicted frame and actual frame. Furthermore, because of the structured nature of this dataset, we can perform an ablation study of how the model responds to spatial and temporal anomalies independent of each other.

Anomalies in the Spatial Domain

In the bouncing ball dataset, the vision system has only ever seen three white balls. As one can see in the learned dictionary, the structure primitives match circular objects. To test the ability of the system to detect spatial anomalies, we introduce a square shape to frames 50-60 as shown in Figure 6 (c) and display the reconstruction error in Figure 6 (a). We plot the spatial prediction error in blue, and the normal prediction error in orange. There is a large jump in the frames where the square anomaly presents itself in the data. We also are able to see a significant jump in reconstruction error (dotted purple) when the model is presented the full (non-truncated) input. This indicates that even when not predicting, the model still has a difficult time reconstructing objects that never presented themselves in the training data. In summary for the spatial domain, it is clear that both the reconstruction error and in predicted error can detect the existence of a spatial anomaly.

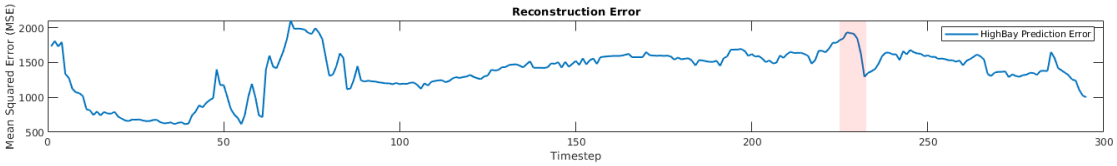
Anomalies in the Temporal Domain

In the temporal domain, we shuffled some of the image sequence data to create discontinuous motion. Specifically, we took frames 20-29 and swapped them with 70-79. In Figure 6 (b) we see the predicted error peaks during the onset of the switch, and again at the offset of the frame switch. In Figure 6 (d) we can visualize why the error peaks in this fashion. As the image sequence progresses, the third image frame in on the (top row) original image sequence shows the discontinuity. In the respective (bottom) predicted image



(a) Predictive reconstruction error in the Desert environment at the Naval Research Laboratory

(b) Sequence that corresponds to the discontinuity in MSE error (a) highlighted in red.



(c) Predictive reconstruction error in the High Bay environment at the Naval Research Laboratory

(d) Sequence that corresponds to the high MSE error (c) highlighted in red

Figure 7: Plot of reconstruction error using our spatiotemporal predictive model in the (a) Desert scene and in (b) a reconfigurable High Bay. (b)(d) The video frame sequence that corresponds to respective areas highlighted in red in the reconstruction error graph. (b) The anomaly presented here is due to a bright light entering the image frame. (d) The anomaly presented here is due to fast moving vehicle hitting the PackBot.

sequence, the model expects the ball motion to continue with the current trajectory. This error is captured in the reconstruction error plot.

What is especially interesting in the temporal domain is that this discontinuous motion is not well detected with a non-predictive model. In fact, one can see that the reconstruction error with non-truncated input is nearly zero with the original data and with the temporal anomalous data. Thus, for temporal anomalies, simply relying on reconstruction error is not sufficient, and warrants the utilization of a predictive sparse coding model.

4.2 Real Data

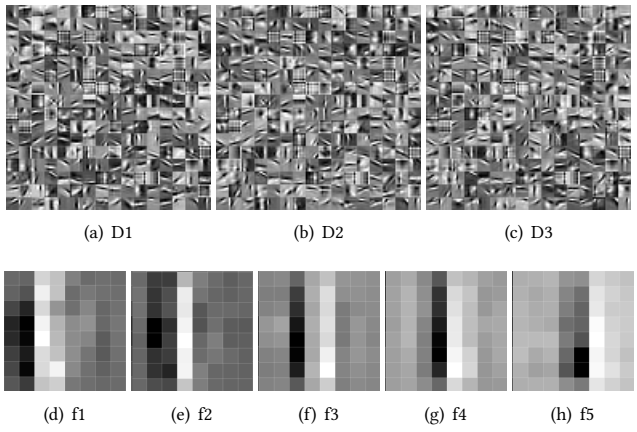
For our real data, we used video data obtained from a small robotics platform (an Endeavor Robotics PackBot 510) with a Carnegie Robotics S7 sensor. This robot was tele-operated around several environments in the Laboratory for Autonomous Systems Research (LASR) at the Naval Research Lab, see Figure 1.

The model is trained over a video sequence of 1,088 video frames sized at 256x256 pixels. Groups of five frames are sparse coded simultaneously (as labeled f1-f5 in Figure 4(a)). The layer labeled V1 defines the joint activation of the input frames in a single sparse activation vector of size 128x128x256. The dictionary elements used to create the sparse code are labelled D1-D5, are of size 8x8x256. These begin as random noise and eventually become structural primitives as shown in Figure 8 (a)-(c).

The layer P1 is higher layer in the architecture that is of size 32x32x512 and has a larger receptive than its V1 counterpart. The neurons within this layer have a larger receptive field and see the image input as an entire sequence. Metaphorically, P1 is at a higher level of cognition in the "brain" and can provide top-down feedback to guide the lower layers. In this case, the sequence memory has been stored in the dictionary elements D1-D5 and are more edge-like representing the structural primitives of the real world. An example of such a sequence can be seen in Figure 8(d)-(h).

We present the predictive reconstruction error of two image sequences captured in the LASR environments, one is a desert scene and one is in a reconfigurable mechanical bay, see Figure 7. As expected, in these real world environments, the predictive reconstruction error has much more variance than the synthetic data. However, as illustrated in the plots in Figure 7 (a)(c), there are relative peaks in the error that map to semantic events in the video sequences. For example, there is a very high discontinuity in the Desert MSE around frame 110. When looking back at the image data, we see that the robot moves around the corner and a blinding light suddenly appears. In the High Bay MSE, there are strong peaks visible when the robot rotates about its axis. In frames 60-80,

the robot performs nearly a 180 degree turn. We show a different relative peak in MSE through Figure 7(d). This is anomalous event where another robot in the scene appears and collides with the robot at high speed.



(a) D1 (b) D2 (c) D3

(d) f1 (e) f2 (f) f3 (g) f4 (h) f5

Figure 8: (a), (b), and (c) Show the dictionary D1, D2, and D3 learned from observing video sequences. We expand on one of the sequences in (d)-(h) from a single dictionary element contained in D1-D5. This particular sequence memory shows an edge moving from left to right and would be a generator for local edge movements in an input video sequence.

5 CONCLUSION

In conclusion, we created an artificial model that is able to mimic early, low level biological predictive behavior in a computer vision system. We utilize a biologically plausible algorithm based upon sparse coding to learn spatiotemporal sequence memories. Without supervision, our sparse coding framework learns the causes of the data by observation. We experimented on both synthetic data and real world data to illustrate the benefits of our model. Our ablation experiments on synthetic data show that we are able to accurately predict what will happen in the future; and that our model can detect anomalous events in space and time. Our experiments on real world data show more variance in predictive error, and also show that we are able to map semantic events in the video sequence to predictive reconstruction error.

6 ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1846023.

REFERENCES

[1] Roland Baddeley. 1996. Visual-Perception-An Efficient Code in V1. *Nature* 381, 6583 (1996), 560–561.
[2] Moshe Bar. 2007. The proactive brain: using analogies and associations to generate predictions. *Trends in cognitive sciences* 11, 7 (2007), 280–289.
[3] Eric B Baum, John Moody, and Frank Wilczek. 1988. Internal representations for associative memory. *Biological Cybernetics* 59, 4-5 (1988), 217–228.
[4] Michael Brosch and Christoph E Schreiner. 2000. Sequence sensitivity of neurons in cat primary auditory cortex. *Cerebral Cortex* 10, 12 (2000), 1155–1167.
[5] Yuwei Cui, Subutai Ahmad, and Jeff Hawkins. 2016. Continuous online sequence learning with an unsupervised neural network model. *Neural computation* 28, 11 (2016), 2474–2504.
[6] Peter Foldiak. 2003. Sparse coding in the primate cortex. *The handbook of brain theory and neural networks* (2003).
[7] Jeffrey P Gavornik and Mark F Bear. 2014. Learned spatiotemporal sequence recognition and prediction in primary visual cortex. *Nature neuroscience* 17, 5 (2014), 732.
[8] David H Hubel and Torsten N Wiesel. 1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology* 160, 1 (1962), 106–154.
[9] Edward Kim, Darryl Hannan, and Garrett Kenyon. 2018. Deep Sparse Coding for Invariant Multimodal Halle Berry Neurons. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)* (2018).
[10] Peter Lennie. 2003. The cost of cortical computation. *Current biology* 13, 6 (2003), 493–497.
[11] Michael D Mauk and Dean V Buonomano. 2004. The neural basis of temporal processing. *Annu. Rev. Neurosci.* 27 (2004), 307–340.
[12] Bruno A Olshausen and David J Field. 1997. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research* 37, 23 (1997), 3311–3325.
[13] Christopher Rozell, Don Johnson, Richard Baraniuk, and Bruno Olshausen. 2007. Locally competitive algorithms for sparse approximation. In *IEEE International Conference on Image Processing*, Vol. 4. IEEE, IV–169.
[14] Peter F Schultz, Dylan M Paiton, Wei Lu, and Garrett T Kenyon. 2014. Replicating kernels with a short stride allows sparse reconstructions with fewer independent kernels. *arXiv preprint arXiv:1406.4205* (2014).
[15] Ilya Sutskever, Geoffrey E Hinton, and Graham W Taylor. 2009. The recurrent temporal restricted boltzmann machine. In *Advances in neural information processing systems*. 1601–1608.