RECONSTRUCTIVE MEMORY FOR ABSTRACT SELECTIVE RECALL

Catherine Wong
Department of Computer Science
Stanford University
catwong@stanford.edu

ABSTRACT

Biological episodic memory is context-dependent and reconstructive. Despite well-documented resource constraints and inaccuracies, humans can recall richly detailed events at varying levels of abstraction, remember individually salient features and abstract general gists, leverage learned knowledge to form memories in circumstances with sparse or limited information, and reconstruct and combine parts of prior experiences to guide complex future behavior. This work presents a computational mechanism for reconstructive memory that could enable similarly selective, hierarchically feature-addressable recall. We propose an architecture that integrates a recurrent neural network controller with both reconstructive memory and a content-addressable working memory. Finally, we suggest methods for evaluation and avenues for future research using reconstructive memory as a baseline for memory consolidation and compositional future planning.

1 INTRODUCTION

Modern deep neural networks excel when they can learn slowly and act quickly. On tasks that require pattern recognition or short-term decision making, such as image classification (Razavian et al., 2014) or phrase translation (Sutskever et al., 2014), even relatively simple models trained on sufficiently large datasets have proven incredibly (even unreasonably!) effective (LeCun, 2014).

In both biological and artificial neural networks, however, researchers have long suggested that explicit, quickly-writable memory plays an important role in enabling broader intelligence (Eichenbaum, 1993). Machines and organisms that can rapidly encode the world into memory can later draw on those memories to learn new knowledge quickly from a few examples (McClelland et al., 1995). Memories can allow a system to reason in complex ways over variable-like bits of information (Feldman, 2013), or perform tasks that require storing and accessing information over longer timescales, such as writing a computer program, reading and understanding a great but hefty novel, or deciding how to act now, based on the experiences accumulated over a lifetime.

The idea of combining neural networks with a more rapidly writable memory is not new. Early neural network architectures introduced the complementary learning systems framework to combat issues of catastrophic forgetting in connectionist models, inspired by evidence that the hippocampus and neocortex act similarly to support short and longer term learning (McClelland et al., 1995). More recently, newer memory-augmented neural networks (Graves et al., 2016; 2014; Sprechmann et al., 2018; Wayne et al., 2018) have revived and refined methods of building deep learning systems with external memories.

In these architectures, however, the challenges of incorporating a useful external memory system generally fall along two largely distinct axes:

Deciding how to remember. Many recent systems formalize the problem of machine memory using a fully differentiable Von Neumann-like architecture. In these systems, such as the Differentiable Neural Computer and Neural Turing Machine (Graves et al., 2016; 2014), the entire system is optimized end to end to maximize reward obtained from solving a memory-intensive task, while learning how to write and read from its external memory along the way. While successful at some tasks, the complexity and underconstrained nature of end to end optimization can make these systems very difficult to train. Worse, a network that learns to remember only in service of immediate rewards may

be woefully short-sighted, with no incentive to remember latent information that could be useful for metalearning and future tasks (Wayne et al., 2018).

More recent architectures (Sprechmann et al., 2018; Wayne et al., 2018) separate the process of encoding and using memory. Often directly inspired by evidence that the brain is composed of specialized modules optimized towards functionally diverse ends, these architectures generally leverage separately trained encoder networks to form memories and controller networks to select and use them. However, when restricting how information is compressed and stored into memories, these architectures face another challenge:

Deciding what to remember. Architectures that leverage compressive encoding for content-based memory retrieval must balance the demands of efficient and effective content-based lookup with domain specificity. In general, these systems must decide at the time that the memory is encoded which details matter. Inevitably, this means that the system must risk either throwing important details away, or face the difficulty of handling very large memories using content-based lookup. If under-compressed, the system will need to generate more precise lookup keys to retrieve complex memories, and may be hard pressed to generate the correct cues to recall useful information later.

In this paper, we propose combatting both of these challenges through a system capable of both content-based and selectively reconstructive memory. While research into the neurological mechanisms of memory is still ongoing and far from complete, ample evidence suggests that biological memories are both reconstructive and generative (Schacter, 2012). Specifically, we draw on two ideas from cognitive science and neuroscience: 1. *hippocampal indexing theory*, which proposes that memories can be stored as memory traces of activations in encoding neural networks, and restored by reactivating those same neurons (Teyler & Rudy, 2007), and 2. evidence that encoding networks decompose sensory information into hierarchical, discriminative features at different layers of abstraction (Zeiler & Fergus, 2014).

In our system, we propose augmenting a neural network with both a content-addressable working memory and a longer-term reconstructive memory accessible by specifying specific subsets of encoding neurons that produced a memory trace. That is, the system should mitigate the tradeoff of deciding what and how much to remember by remembering as much as possible, and then leveraging the hierarchically abstract structure of the encoding network to learn where in its memory to look later on. The main contributions of this work are to:

1. Introduce a mechanism for reconstructive memory, using the same sensory network for encoding and decoding memories.
2. Introduce a mechanism for neuronal-unit based recall using reconstructive memory, in which a controller learns to specify which parts of a full memory trace it should reconstruct.
3. Propose a sample integrated architecture that makes use of this memory scheme to combine a reconstructive, neuronal-unit based memory and a smaller, content-based working memory.

This paper is intended as a proposal, and focuses primarily on mechanisms for reconstructive memory storage and recall. However, there is clearly room to both further develop the model and refine these mechanisms for practical use, and we discuss possibilities for future work at the end of the paper.

2 System Overview

At a high level, the model (Figure 1) consists of two trainable components:

1. An encoder, which both encodes perceptual inputs into memory and also functions as a reconstructive decoder, and
2. A controller, which decides how to access and use its memories.

These components interact with two memory banks:

1. A small, content-addressable working memory, and

2. A larger bank of memory traces, accessible by specifying subsets of activations to recall in any memory trace.

We describe these components in detail below.

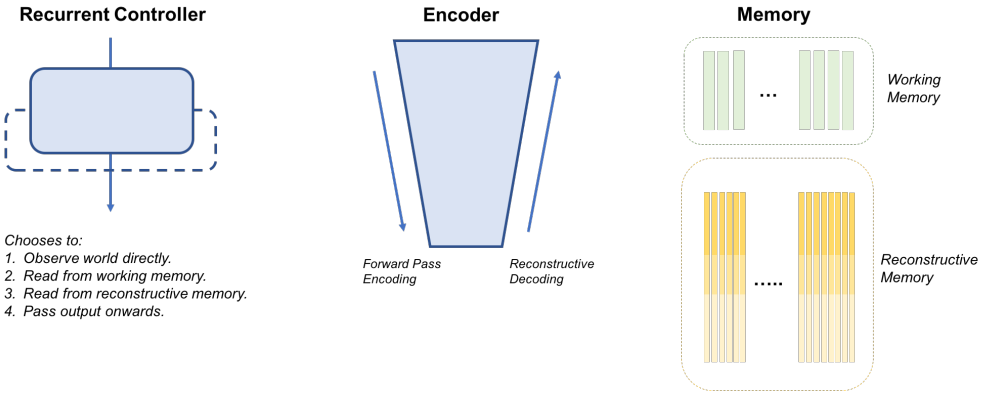

Figure 1: The proposed model architecture, with a recurrent neural network controller, an encoder that encodes perceptual inputs into memory and functions as a reconstructive decoder, a content-addressable working memory, and a reconstructive memory bank.

2.1 Encoder

In this work, as a running example, we will focus on the encoding and reconstructive recall of visual memories. We consider an encoder network as a simple multi-layered convolutional neural network, pre-trained on a simple object classification objective. Convolutional neural networks (CNNs) have consistently produced excellent results on visual tasks even with relatively simple training objectives (Krizhevsky et al., 2012), and evidence suggests that the dynamics of deep CNNs may resemble hierarchical visual encoding in the human visual ventral stream (Yamins et al., 2014). More importantly for this work, a large body of work has demonstrated that neuronal activations at intermediate layers of these networks can produce relatively disentangled, hierarchically abstract, and even semantically interpretable features (Zeiler & Fergus, 2014) - for example, activations at initial layers generally respond to semantically simple visual features such as edges and basic shapes, whereas later layers may respond to increasingly abstract features, such as eyes or dog ears. Additionally, deconvolutional methods exist to reconstructively map activities at intermediate layers back to the input pixel space (Zeiler et al., 2010).

More generally, however, the same idea of reconstructive decoding can be extended to encoders for other applications (such as text decoders (Wayne et al., 2018)), and easily encompasses architectures optimized for different cost functions, such as predictive autoencoding or explicit feature disentanglement.

2.1.1 Forward Pass Encoding

To encode visual inputs, we perform a standard forward pass over a fixed size input image through the trained encoder network. In most standard CNN architectures trained for image classification, models are defined as a series of layers that map an input image x_i to a probability vector y_i over the C different classes, where each layer consists of (i) convolution of either the initial image or the previous layer output with a set of learned filters, (ii) passing the resulting responses through a nonlinear activation function such as a rectified linear function, and (iii) optional additional normalization and regularization functions, such as max pooling over local image patches, batch normalization, or neuronal dropout (Simonyan & Zisserman, 2014).

In many standard encoding procedures, only the output of a single layer, such as the penultimate layer of a network trained towards an image classification objective (Sprechmann et al., 2018) or the final layer of a network trained for predictive autoencoding (Wayne et al., 2018), is stored as

an image embedding. In systems that leverage compressive encoding for content-based memory access, these embeddings are considered sufficiently rich to capture useful extracted features from the original input image in a compact vector embedding. However, in this paper, we consider the full activation trace of the input image through the encoding network - that is, the outputs at each layer of the network - as a hierarchically organized memory trace. For example, if considering an encoder implemented as the well-documented VGG-16 network, passing a single (224x224x3) image tensor through the network would produce a memory trace consisting of 6 embedding tensors of sizes ((112x112x64), (56x56x128), (28x28x256), (14x14x512), (7x7x512), (1x1x1000)), respectively (Simonyan & Zisserman, 2014).

Much as the hippocampus may store much of the full original neocortical activation pattern as episodic memory (Teyler & Rudy, 2007), we store the entire hierarchical set of encoding layer outputs, allowing the controller to decide later which relevant parts of the trace to reconstruct at recall time by addressing the neuronal units that produced the trace.

Upon completion of a forward pass at time t, the resulting memory trace is stored in two separate memory banks:

1. The full memory trace M_t consisting of a tuple of layer outputs $(m_0, m_1, ...m_n)$ in an n-layer embedding network is stored in a long-term memory bank $M_{reconstructive}$. Implementations of the memory bank can vary, but the simplest implementation could be a naive two dimensional matrix of size $(N_{mem}, |z|)$, where N_{mem} is the total capacity of the memory bank and $|z|$ is the unraveled length of the concatenated layer outputs.

   If full, the memory bank can make use of eviction schemes such as FIFO, LRU, or even a learned eviction or further consolidation scheme, which is clearly an avenue for future research.

2. Optionally, an immediate memory embedding m_t consisting of a standard image embedding (such as the output of the penultimate fully connected layer of a CNN or the output of a network trained specifically for predictive autoencoding) can be stored directly in content-addressable working memory $M_{working}$ of size $N_{working}, |m_t|$. Again, working memory bank implementations can vary, but the simplest implementation is a simple, fixed-size two-dimensional matrix where new memory embeddings are concatenated onto the existing memory bank and old memories are evicted using any eviction scheme.

By storing full memory traces and optionally immediate memory embeddings, we allow the system to form hierarchically detailed latent memories while still paying specific, immediate attention to the world, written into working memory. While in this simple implementation we consider storing the full memory trace as the set of encoding layer outputs, future work can consider refining which parts of the memory trace to store or even running the resulting outputs through an additional compressive or encoding module, as considered in the discussion.

2.1.2 Reconstructive Decoding-Encoding

We now describe a reconstructive decoding and encoding procedure, by which specific parts of any one memory trace can be reconstructed to form a new memory embedding highlighting those salient features. At a high level, this method is inspired by evidence from hippocampal indexing theory suggesting that the hippocampus may store a memory trace as pointers to the original neuron activations that fire during an episodic memory, which can then be selectively reactivated to reconstruct relevant features of the original episode. Additionally, we aim to provide an alternative to strictly content-based memory addressing, instead allowing the controller to learn generally which *neuronal units* may correspond to relevant features for any given task - intuitively, this may both allow the controller to lookup memories by specifying more arbitrarily abstract knowledge (hierarchically ranging from did I see any dog at all to what was the color of the specific dogs nose, and what was in the background, using the same visual memory trace), and allow a kind of variable-based addressing, whereby the controller can lookup memories by specifying neurons corresponding to specific abstract features, and then access the actual values of those features in any given value.

Further, we use the encoder network itself to re-encode the reconstructed memory, capturing the intuition that 1. existing learned knowledge in the encoder network informs the reconstruction of

memory, and 2. future learning, if the encoder network is trained further, can inform the reconstruction of older memories.

Formally, we propose a reconstruction scheme for a given memory trace M_t consisting of a tuple of layer outputs $(m_0, m_1, ...m_n)$ produced by the encoder as follows:

1. *Attentional selection to specify which neurons to reconstruct.* As described in 2.2.3, the controller produces a reconstructive memory read-key vector k of size $|z|$ which will specify as a soft attention which neurons in the memory trace we would like to reconstruct. We can consider this key to in fact be composed of layer-wise read weighting matrices $(k_0, k_1, ...k_n)$, each the size of the corresponding layer outputs in the memory trace. Currently, of course, this method produces an attention over the entire memory trace, which may not be desirable. Future implementations could consider restricting attention to a single or multiple layers in the original encoder, or enforce sparsity constraints on the read key for more efficient and specific reconstruction.
2. *Reconstruction as deconvolution.* We draw on the well-documented deconvolutional method (Zeiler & Fergus, 2014) to reconstruct visual memory traces as weighted mappings of the activations in the stored memory trace back to the input pixel space.

   Considering the full series of layer outputs $(m_0, m_1, ...m_n)$, where m_0 is the first layer output and m_n is the output of the final layer, we begin with the final layer m_n and work backwards to m_0, performing the following operation at each layer m_i to produce intermediate reconstruction r_i:

   (a) Compute the attentionally weighted layer output at this layer $m_{i,weighted}$ as the element-wise product of m_i and k_i.
   (b) Combine this with the reconstruction from the previous layer r_{i+1} to produce $m_i = m_{i,weighted} + r_{i+1}$.
   (c) Unpooling and rectification: in networks that use pooling, we follow an unpooling procedure as described in (Zeiler & Fergus, 2014). In networks that use RELU nonlinearities, to obtain valid feature reconstructions at each layer, we pass the reconstructed signal thus far through a RELU nonlinearity.
   (d) Filtering: we apply transposed versions of the feature maps at layer i to the rectified maps, producing an upsampled, intermediate layer output r_i.

   At the highest level, the resulting reconstruction r_0 will be a selectively reconstructed image mapped back to input pixel space, with the selected salient features amplified as specified by the controller and nonsalient features masked.
3. *Re-encoding.* Finally, we map the reconstructed image back down to embedding space by running a forward pass on the *reconstructed image* r_0 to produce a new reconstructed memory embedding $m_{reconstructed}$. Optionally, we could consider storing the memory trace produced by this forward pass in our reconstructive memory bank, and the resulting memory embedding in working memory, so that we produce an updated, recent record of our reconstructive recall for future use.

2.2 CONTROLLER

The second trainable component of this system is a controller network, which determines how memories are accessed and used at each timestep of operation.

Specifically, we consider a recurrent controller implemented as a deep LSTM (Hochreiter & Schmidhuber, 1997) of two hidden layers, which we describe here for completeness. At each timestep t, the network receives an input x_t of size —m_t—, the size of the working memory embedding vectors produced by the encoder network. Each layer l maintains a recurrent hidden state h_t^l and cell state s_t^l, which are updated recursively as follows (using $\sigma(x) = (1+exp(x)^1)$):

$$i_t^l = \sigma(W_i^l[x_t, h_{t1}^l, h_t^{l1}] + b_i^l)$$
$$f_t^l = \sigma(W_f^l[x_t, h_{t1}^l, h_t^{l1}] + b_f^l)$$
$$s_t^l = f_t^l s_{t1}^l + i_t^l tanh(W_s^l[x_t, h_{t1}^l, h_t^{l1}] + b_s^l)$$

$$o_t^l = \sigma(W_o^l[x_t, h_{t1}^l, h_t^{l1}] + b_o^l)$$
$$h_t^l = o_t^l tanh(s_t^l)$$

At each timestep, this network produces a vector h_t produced by concatenating hidden state vectors from each layer. This vector which is finally passed through a single linear layer to produce an output $LSTM_{out}$, which can be used for action and memory selection.

We segment $LSTM_{out}$ into two vectors $[a_t, y_t]$, where a_t is an action selection vector passed to a softmax and max function to produce a one-hot vector used to select over the following four actions, and y_t is an output vector that will be used in each of these actions. The controller can decide to perform one of the following actions at each timestep:

1. Look at the world: read a world observation from the encoder network directly into working memory.
2. Working memory read: select memories using content-based reading from its working memory bank.
3. Reconstructive memory read: select memories using unit-based reading from its reconstructive memory.
4. Output redirection: pass the output vector onwards to a task-specific output network, where it can be used to produce a final output.

We now describe each of these actions in detail.

2.2.1 Direct World Observation

This action allows the controller to directly receive a world observation as input, and read this observation into working memory. In the simplest implementation, we run a single forward pass through the encoder as described in 2.1.1, producing a memory embedding that is written directly into working memory and a memory trace written into reconstructive memory. Additionally, we pass the resulting working memory embedding directly to the controller as input x_{t+1} on the next timestep.

Optionally, future implementations could consider allowing the controller to choose how it observes the world, even when receiving a direct world observation, using the same reconstructive memory mechanism described later in 2.2.3. In this implementation, the output vector would be used as described to directly control which features of the current world observation are used to produce the final world observation embedding vector.

2.2.2 Working Memory Read

This action allows the controller to select and read memories from its content-based working memory. As in other related work (Graves et al., 2016), we use an attentional, read-key based algorithm to read from memory, based on the similarity between a read key and the memories stored in the working memory bank.

To produce a read key, we pass the controller output y_t through a multilayer perceptron to produce read key k_t of size $|m_i|$, where m_i is a memory embedding vector in the working memory bank. We compute the cosine similarity between the read key and each memory m_i as $c_i = cos(k, m_i) = \frac{k \cdot m_i}{|k||m_i|}$, and then produce a normalized weighting vector of length $N_{working}$ where each element is calculated as $w[i] = \frac{exp(c_i)}{\Sigma_i exp(c_i)}$.

We finally produce the working memory read as a weighted average over the entire working memory $M_{working}w$, which is passed as input to the controller at the next timestep.

2.2.3 Reconstructive Memory Read

This action leverages the reconstructive decoding-encoding procedure described in 2.1.2, allowing the controller to read from reconstructive memory by specifying - on the level of neuronal units - which features it wants to reconstruct from memory. This read method is one of the main novelties

of the system, and aims to capture the intuition that a controller may want to access memories not only based on their content, but also based on the *neuronal activations that may have produced the original memory trace*, leveraging the discriminative, hierarchical, and functionally specific nature of the encoding network itself to decide what in the original memory it is looking for. Intuitively, this also aims to give the controller control over the salient aspects of the original memory trace that it wants to remember, instead of relying on a single, fixed encoding of the original memory. Rather, the controller specifies what of the original memory it wishes to reconstruct, and then can access encodings of the reconstructed memories to attend to specifically relevant details at the appropriate level of hierarchical granularity.

We describe the simplest implementation of this method below, in which the controller may choose to specify which over all possible encoder activations in the entire memory trace should be reconstructed. In practice, this may prove overly general, and future work should aim to further refine the controller reconstructive memory read process - for example, by perhaps restricting attention to specific layers or hierarchical subsets of activations in the memory trace, by further preprocessing the memory trace to compress the trace further, or even by using other cached read vectors (say, other memories) to parameterize in advance which subsets of activations to reconstruct, so that over time, the system can remember how to remember.

Even in the most naive case, where the controller can read over the full memory trace, additional experimentation remains to determine at what stage memories should be read and reconstructed. For example, we could imagine producing a read key over the memory trace in which we preselect memories that already showed high activation in the desired units, and then only reconstructing these most salient memories. Alternatively, we could imagine introducing a recency heuristic, in which the controller instead searches over its most recent memories, reconstructing the specified units and then using another metric, like a second content-based read key, to decide whether the reconstructed memory is relevant. In either case, we produce an attentional, reconstructive read key over all activations in a full memory trace by passing the controller output y_t through a multilayer perceptron to produce reconstructive key k_t^r of size $|z|$, where $|z|$ is the size of the unraveled memory trace as described in 2.1.1. In the former preselection case, we can either produce a weight vector over the full reconstructive memory bank used to then reconstruct and combine memories (or simply the top n memories) based on cosine similarity to the read key as described in 2.2.2, or alternatively, we can imagine iteratively considering reconstructed memories based on their cosine similarity, in which we sequentially choose the most relevant memory to reconstruct, produce a reconstructive embedding $m_{reconstructed}$ as described in 2.1.2, and then pass this memory to the controller as the input at the next timestep. This latter method could reduce the number of relatively expensive memory reconstructions, and allow the controller to decide, based on the memory, whether additional memory reconstructions were necessary.

2.2.4 Output Redirection

This action allows the controller to produce a domain-specific output, such as a movement in the world. We consider the simplest case where the entire system is being trained for a single task at a time, and there is only one domain-specific output network; of course, this can easily be extended to allow the controller to choose from any number of separate domain-specific output networks.

We simply pass the output vector y_t to the domain-specific output network, which produces the appropriate task output as part of downstream processing. For example, in the case where the task is to produce some movement in an environment, this network may be a two layer MLP, which projects to the logits of a multinomial softmax with the dimensionality of the movement space where we can then sample an output movement.

At the next time step, we also default to making another direct world observation as described in 2.2.1, allowing the controller to observe the impact of its output in the world. Depending on the domain, we may also wish to pass the results of the output network itself to the controller network at the next timestep, which we leave as a task-specific implementation detail.

3 EXPERIMENTS AND EVALUATION

This paper focuses on mechanisms for reconstructive memory storage and recall, and is intended primarily as a computational architecture proposal. However, the recent explosion of effective machine learning models has only intensified the need for challenging, specific evaluation metrics to test comprehension. In artificial intelligence, as in human psychology, testing on insufficiently narrow or poorly controlled tasks can be doubly susceptible to failure - we can wind up both unclear what the system has actually learned (Yang et al., 2018), and unsuspecting of future corner cases (Papernot et al., 2016). Good task design can highlight the difference between human learning and Clever Hans (Samhita & Gross, 2013).

In this section, we therefore draw on the rich history of tasks in computer vision and cognitive science to suggest several evaluation benchmarks that could be appropriate to test aspects of a hierarchically feature-accessible, detailed, reconstructive memory system. These tasks are intended to highlight desirable features of any complex visual memory system, and may provide a useful baseline for future models.

Long term visual memory tasks. While human memory is imprecise and error prone, our ability to retain detailed and discriminative long-term visual memories can be remarkably high. Brady et. al demonstrated that under certain conditions, humans can recall a massive number of objects in surprising visual detail (Brady et al., 2008). We propose variations of the same tasks to compare against baseline systems in which straightforward content-based lookup of nonreconstructive visual embeddings should fail - a system that overnormalizes in advance to correct against invariances should be insufficiently discriminative, and a system that did not normalize in advance should be unable to perform more abstract recall. Instead, the following battery of tasks should reward salient, feature-based lookup: after exposure to a large quantity of visual images, Brady et. al evaluate human ability to discriminate between previously-seen objects and unseen objects under:

1. A novel condition: an old item paired with a new, categorically distinct item.
2. An exemplar condition: an old item paired with a physically similar new item from the same basic level category.
3. A state condition: an old item paired with a visually identical new item in a different state or pose.

Challenging visual comprehension tasks. A system that can reconstruct and recall memories with varying levels of visual detail may be better equipped to handle complex visual comprehension tasks, including tasks that require tracking complex scenes over time, and answering arbitrarily detailed queries. A number of emerging datasets attempt to test complex visual reasoning: the CLEVR dataset (Johnson et al., 2017) tests compositional language and visual reasoning in highly-controlled, computer generated scenes, and the COG dataset (Yang et al., 2018) tests visual reasoning about image sequences over time, with distractors present. Converted into memory tasks, either dataset may provide useful benchmarks to test visual recall over time. While both datasets use computer generated images to control for superficial statistical biases, we also suggest developing more controlled real-world scene comprehension benchmarks to test richer visual recall.

Additionally, reconstructive memory may boost performance on video comprehension tasks, such as the Moments in Time (Monfort et al., 2018) or MovieQA story understanding benchmarks (Tapaswi et al., 2016), or on video summarization.

Sparse information and new information reconstructive memory tasks. More generally, a visual system augmented with reconstructive memory should be able to recall and make sense of visual memories given limited information (reconstructed memories should leverage prior and general knowledge to describe what was probably present given sparse visual details), and to incorporate information learned later when recalling past events.

Castelhano and Henderson (Castelhano & Henderson, 2003) describe a flash preview and moving window experiment to test the ability of human subjects to use abstract cues learned after initially previewing a scene, such as a target word, to guide visual search with a narrow field of view. More primitively, reconstructive memory could improve performance on visual question answering tasks that require integrating multiple perspectives of the same scene, in which accessing abstract information later could aid comprehension and recall. The classic Heider-Simmel illusion (Heider &

Simmel, 1944) demonstrates human ability to tell rich stories about simple scenes; a more limited task inspired by this test could test the ability of computer systems to generate detailed captions of scenes with missing or limited details.

Finally, we suggest that computational models with reconstructive memory might offer a better model of human biological memory. Prior work in psychology and cognitive science has leveraged advances in computer vision to track human decay of visual representations (Võ et al., 2017), predict image memorability (Khosla et al., 2015), and study dynamics of scene representation in the human brain (Cichy et al., 2016). Models augmented with more human-like recall systems could provide a better baseline architecture to model and evaluate human memory.

4 Foundational Research and Related Work

4.1 Biological Memory

Biological memory is no mere record of the past. It is both more and less; evolving and sometimes conflicting research suggests that human memory can be both detailed and error-prone, specific and context-dependent, reconstructive and fundamentally constructive. At both short and longer timescales, biological memory may be critical to human-like behavior, enabling rich and complex reasoning.

Over shorter timescales, psychology and neuroscience has developed the concept of fast-writable working memory as a capacity-limited (Miller, 1956) but critical component of logical reasoning, fundamental to tasks that require short-term manipulation of prior information (Kyllonen & Christal, 1990). Under the influential central executive paradigm, a central coordinating system maintains and updates information stored across discrete and interacting systems, enabling complex cognitive abilities such as symbol-like and variable-binding behavior (Baddeley, 1992).

More generally, modern evidence depicts human episodic memory as fundamentally constructive (Bartlett, 1920), subject both to rich and context-dependent recall, but also various errors and detailed but highly-inaccurate illusions. Despite these imperfections and obvious capacity limitations, humans can selectively recall a massive number of objects, events, and concepts at hierarchical and arbitrary levels of abstraction and detail (Brady et al., 2008). Biological episodic memory clearly does not entail a simple replay of the past. Instead, its constructive nature may allow both compressive storage and play a critical role in adaptive, selectively useful recall - the future does not precisely replay the past, and composing bits and pieces of prior episodes may allow more flexible and even recombinatory memory that allows humans to imagine, simulate, or pre-experience episodes in the future (Schacter & Addis, 2007). Indeed, the ability to reconstruct arbitrary and specific information at varying levels of detail may be critical for broader comprehension and adaptive transfer learning, and generalization; our ability to remember both individual details and a more general gist of past events may be critical to fast recall and abstract generalization (McClelland, 1995; Oliva, 2005).

This paper also draws on a rich history of neuroscientific theory and evidence as to the specifics of memory representation, storage, update, and retrieval mechanisms that might enable both fast-writable working memory and reconstructive recall. Hippocampal indexing theory (Teyler & Rudy, 2007) suggests that the hippocampus captures memories as an index of neocortical activity at the time of experience, and can serve as an index to project and reactivate patterns in the neocortex based on selective and partial cues. Indeed, theories of experience replay suggest that reactivation of memory traces can reinstate cortical circuit activity during wake and sleep (ONeill et al., 2010). These episodic memories may be subject to ongoing transformations that reflect dynamic updates in the hippocampus and in the substrates that produced those original memories (Moscovitch et al., 2016), as well as consolidatory processes over short and lifelong timescales (Dudai et al., 2015).

4.2 Memory-Augmented Computer Architectures

This proposal draws on many prior computer architectures that combine neural networks and external memories stored and optimized under different conditions than a centrally-executing controller network. Many of these architectures are explicitly informed by theories of biological memory, and attempt to combat issues of learning in neural networks, capture critical functions of complex reasoning and logical behaviors, and model aspects of human episodic and working memory. We

draw inspiration from computational models of complementary learning systems (McClelland et al., 1995) in the neocortex and hippocampus to interleave fast and slow learning of information, the related Memory-based Parameter Adaptation model of a context-dependent recall and dynamically trainable adaptive controller (Sprechmann et al., 2018), reinforcement learning systems that directly make use of episodic reactivation through a stored experience replay buffer (Mnih et al., 2013), and the classic LSTM model of gated long and short term memory (Hochreiter & Schmidhuber, 1997). Additionally, we build off recent attempts to build fully differentiable, trainable Von Neumann-like architectures such as Neural Turing Machine (Graves et al., 2014) and related Differential Neural Computer (Graves et al., 2016), and models that store compressed episode representations using a predictive encoder (Wayne et al., 2018). These architectures present concrete memory-based implementations and intriguing evidence of how memory can enable complex learning, in a rapidly developing area that continues to evolve alongside our understanding of biological and computational memory.

5 DISCUSSION

This paper introduces a mechanism for reconstructive memory in artificial neural networks, and a preliminary architecture to integrate both unit-addressable reconstructive memory and content-addressable working memory. We draw on both biological and computational precedent, and propose that reconstructive memories may allow more complex, abstract recall.

To conclude, however, we turn briefly to the question of whether augmenting computer systems with a reconstructive memory is necessarily even desirable, and suggest both avenues for caution and intriguing future work. Human memory is notoriously fallible, and the same aspects that make reconstructive memory powerful - its ability to draw on prior expectation to reconstruct rich memories from sparse information, and to contextually control which parts of a memory are salient for recall - can make it prone to bias, manipulation, and catastrophic error (Schacter & Addis, 2007). When introducing a reconstructive aspect to computational recall, we must take enormous care to weigh potential tradeoffs in the interpretability and accuracy of reconstructed memory, ensure that systems do not reinforce their own biases endlessly in restricted learned lookup procedures, and can check their own memory against truth to ensure that a reconstructed memory does not introduce problematic errors. As neuroscientists and engineers alike point out, we should not handicap our electronic machines by blindly emulating biology (Russell & Norvig, 2016). A computer scientist who can hardly remember what she ate for breakfast yesterday may not wish to build a machine that cannot remember either. Perhaps reconstructive memory has no place in a system with vast or unbounded storage and lookup, and is simply an evolutionary trick to hide the limited capacity of a resource-constrained mind.

As described earlier, however, this work is motivated by the hope that a selectively reconstructive memory may be more compositional, contextually relevant, and flexible, leveraging the power of a controller to remember arbitrarily abstract concepts that leverage prior knowledge and learned information. Additionally, along with the immediate avenues for future refinement described throughout the paper to improve encoding and decoding, the same basic mechanism may lay the groundwork for more interesting innovations, such as modeling lifelong memory consolidation and imagination. While prior work on computational and biological memory consolidation focuses heavily on gradual integration of hippocampal information into a neural network or the neocortex (McClelland et al., 1995), the proposed memory trace structure could also model memory consolidation through compression and sparsification of the memory traces themselves, perhaps through a process of re-encoding similar to learned weight distillation (Hinton et al., 2015). On the opposite end of the spectrum, future work could leverage the generative nature of reconstructive memory more fully as a form of forward-thinking imagination - a controller that learns to selectively choose parts of memories to reconstruct could instead choose to compose features from disparate memories to construct a wholly new imagined concept, or even to choose to selectively pass features from one memory to be reconstructed in a different decoding network. Clearly, there is plenty of room for future research. We hope this proposal serves as a preliminary but useful building block for computational models of memory whose richness, complexity, and flexibility can parallel and shed light on our own.

ACKNOWLEDGMENTS

Thanks to Nikhil Bhattasali for the reminder that it may be hard for a neuroscientist to understand a microprocessor, but harder still to understand the neuroscientist herself; to Alex Tamkin, for turning around during an inspiring lecture on paying attention and whispering "attention is all you need"; and to the entire CS379C class and lecturers for their insightful and thought-provoking discussion. Special thanks to Tom Dean for his invaluable advice on building programmers apprentices and intelligent systems, and more importantly, for the necessary reminder that in real life, the only truly catastrophic forgetting is forgetting to chase what brings you joy.

REFERENCES

Alan Baddeley. Working memory. *Science*, 255(5044):556–559, 1992.

Frederic C Bartlett. Some experiments on the reproduction of folk-stories. *Folklore*, 31(1):30–47, 1920.

Timothy F Brady, Talia Konkle, George A Alvarez, and Aude Oliva. Visual long-term memory has a massive storage capacity for object details. *Proceedings of the National Academy of Sciences*, 105(38):14325–14329, 2008.

Monica S Castelhano and John M Henderson. Flashing scenes and moving windows: An effect of initial scene gist on eye movements. *Journal of Vision*, 3(9):67–67, 2003.

Radoslaw M. Cichy, Aditya Khosla, Dimitrios Pantazis, and Aude Oliva. Dynamics of scene representations in the human brain revealed by meg and deep neural networks. *Neuroimage*, 2016.

Yadin Dudai, Avi Karni, and Jan Born. The consolidation and transformation of memory. *Neuron*, 88(1):20–32, 2015.

Howard Eichenbaum. *Memory, amnesia, and the hippocampal system*. MIT press, 1993.

Jerome Feldman. The neural binding problem (s). *Cognitive neurodynamics*, 7(1):1–11, 2013.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538 (7626):471, 2016.

Fritz Heider and Marianne Simmel. An experimental study of apparent behavior. *The American journal of psychology*, 57(2):243–259, 1944.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pp. 1988–1997. IEEE, 2017.

Aditya Khosla, Akhil S. Raju, Antonio Torralba, and Aude Oliva. Understanding and predicting image memorability at a large scale. In *International Conference on Computer Vision (ICCV)*, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Patrick C Kyllonen and Raymond E Christal. Reasoning ability is (little more than) working-memory capacity?! *Intelligence*, 14(4):389–433, 1990.

Yann LeCun. The unreasonable effectiveness of deep learning. In *Seminar. Johns Hopkins University*, 2014.

James L McClelland. Constructive memory and memory distortions: A parallel-distributed processing approach. *Memory distortions: How minds, brains, and societies reconstruct the past*, pp. 69–90, 1995.

James L McClelland, Bruce L McNaughton, and Randall C O'reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Mathew Monfort, Bolei Zhou, Sarah Adel Bargal, Alex Andonian, Tom Yan, Kandan Ramakrishnan, Lisa Brown, Quanfu Fan, Dan Gutfruend, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *arXiv preprint arXiv:1801.03150*, 2018.

Morris Moscovitch, Roberto Cabeza, Gordon Winocur, and Lynn Nadel. Episodic memory and beyond: the hippocampus and neocortex in transformation. *Annual review of psychology*, 67: 105–134, 2016.

Aude Oliva. Gist of the scene. In *Neurobiology of attention*, pp. 251–256. Elsevier, 2005.

Joseph ONeill, Barty Pleydell-Bouverie, David Dupret, and Jozsef Csicsvari. Play it again: reactivation of waking experience and memory. *Trends in neurosciences*, 33(5):220–229, 2010.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014. URL `http://arxiv.org/abs/1403.6382`.

Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

Laasya Samhita and Hans J Gross. The clever hans phenomenon revisited. *Communicative & integrative biology*, 6(6):e27122, 2013.

Daniel L Schacter. Constructive memory: past and future. *Dialogues in clinical neuroscience*, 14 (1):7, 2012.

Daniel L Schacter and Donna Rose Addis. The cognitive neuroscience of constructive memory: remembering the past and imagining the future. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1481):773–786, 2007.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Pablo Sprechmann, Siddhant M Jayakumar, Jack W Rae, Alexander Pritzel, Adrià Puigdomènech Badia, Benigno Uria, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. Memory-based parameter adaptation. *arXiv preprint arXiv:1802.10542*, 2018.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. MovieQA: Understanding Stories in Movies through Question-Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Timothy J Teyler and Jerry W Rudy. The hippocampal indexing theory and episodic memory: updating the index. *Hippocampus*, 17(12):1158–1169, 2007.

Melissa Le-Hoa Võ, Zoya Bylinskii, and Aude Oliva. Image memorability in the eye of the beholder: Tracking the decay of visual scene representations. *bioRxiv*, pp. 141044, 2017.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack Rae, Piotr Mirowski, Joel Z Leibo, Adam Santoro, et al. Unsupervised predictive memory in a goal-directed agent. *arXiv preprint arXiv:1803.10760*, 2018.

Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

Guangyu Robert Yang, Igor Ganichev, Xiao-Jing Wang, Jonathon Shlens, and David Sussillo. A dataset and architecture for visual reasoning with a working memory. *arXiv preprint arXiv:1803.06092*, 2018.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2528–2535. IEEE, 2010.