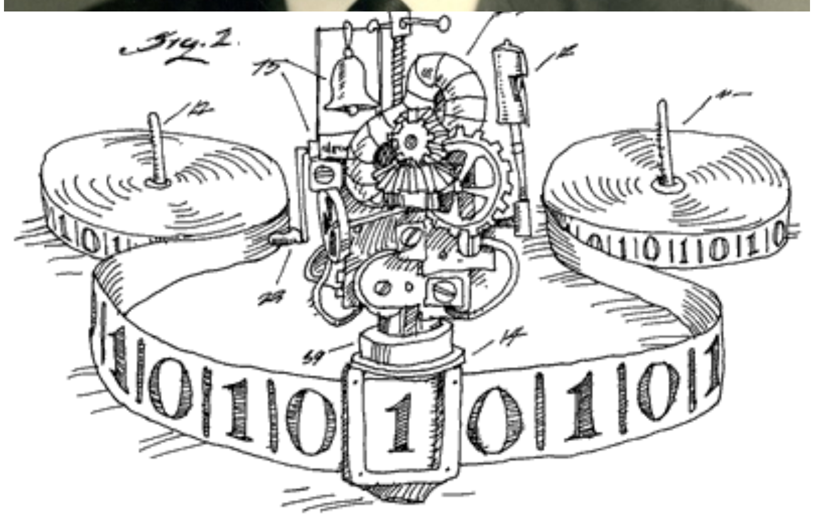
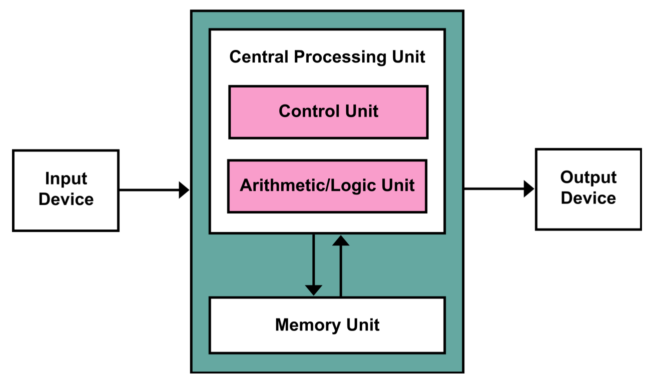
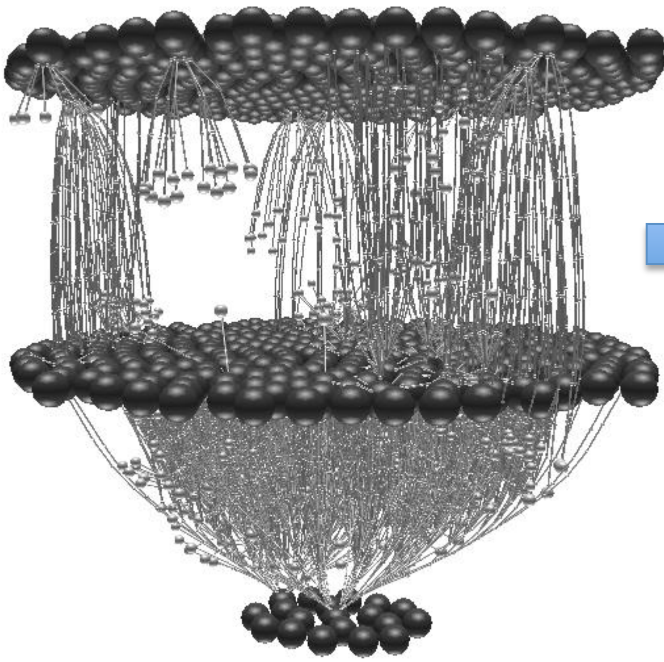
Neural Turing Machines

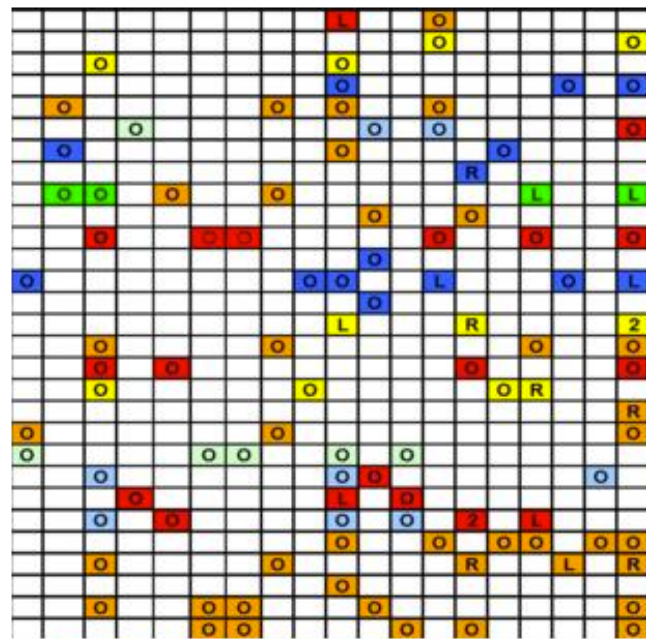
Can neural nets learn programs?

Alex Graves
Greg Wayne
Ivo Danihelka




Central Processing Unit
Control Unit
Arithmetic/Logic Unit
Input Device
Output Device
Memory Unit





Contents

Introduction

- First application of Machine Learning to logical flow and external memory

Introduction

- First application of Machine Learning to logical flow and external memory
- Extend the capabilities of neural networks by coupling them to external memory

Introduction

- First application of Machine Learning to logical flow and external memory
- Extend the capabilities of neural networks by coupling them to external memory
- Analogous to TM coupling a finite state machine to infinite tape

Introduction

- First application of Machine Learning to logical flow and external memory
- Extend the capabilities of neural networks by coupling them to external memory
- Analogous to TM coupling a finite state machine to infinite tape
- RNN's have been shown to be Turing-Complete, Siegelmann et al '95

Introduction

- First application of Machine Learning to logical flow and external memory
- Extend the capabilities of neural networks by coupling them to external memory
- Analogous to TM coupling a finite state machine to infinite tape
- RNN's have been shown to be Turing-Complete, Siegelmann et al '95
- Unlike TM, NTM is completely differentiable

Foundational Research

- Neuroscience and Psychology
  - Concept of “working memory”: short-term memory storage and rule based manipulation
  - Also known as “rapidly created variables”

Foundational Research

- Neuroscience and Psychology
  - Concept of “working memory”: short-term memory storage and rule based manipulation
  - Also known as “rapidly created variables”
  - Observational neuroscience results in the pre-frontal cortex and basal ganglia of monkeys

Foundational Research

- Neuroscience and Psychology
- Cognitive Science and Linguistics
  - AI and Cognitive Science were contemporaneous in 1950's-1970's

Foundational Research

- Neuroscience and Psychology
- Cognitive Science and Linguistics
  - AI and Cognitive Science were contemporaneous in 1950's-1970's
  - Two fields parted ways when neural nets received criticism, Fodor et al. '88
    - Incapable of "variable-binding"
      - eg "Mary spoke to John"
    - Incapable of handling variable sized input

Foundational Research

- Neuroscience and Psychology
- Cognitive Science and Linguistics
  - AI and Cognitive Science were contemporaneous in 1950's-1970's
  - Two fields parted ways when neural nets received criticism, Fodor et al. '88
  - Motivated Recurrent Networks research to handle variable binding and variable length input

Foundational Research

- Neuroscience and Psychology
- Cognitive Science and Linguistics
  - AI and Cognitive Science were contemporaneous in 1950's-1970's
  - Two fields parted ways when neural nets received criticism, Fodor et al. '88
  - Motivated Recurrent Networks research to handle variable binding and variable length input
  - Recursive processing hot debate topic in role inhuman evolution (Pinker vs Chomsky)

Foundational Research

- Neuroscience and Psychology
- Cognitive Science ad Linguistics
- Recurrent Neural networks
  - Broad class of machines with distributed and dynamic state

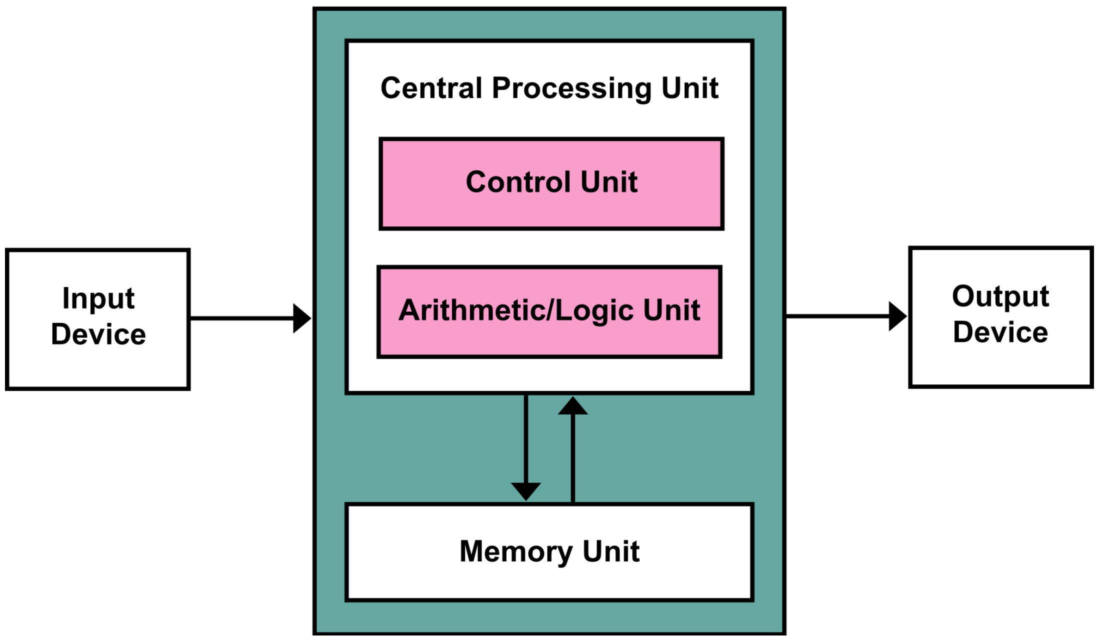
Foundational Research

- Neuroscience and Psychology
- Cognitive Science ad Linguistics
- Recurrent Neural networks
  - Broad class of machines with distributed and dynamic state
  - Long Short Term Memory RNN's designed to handle vanishing and exploding gradient

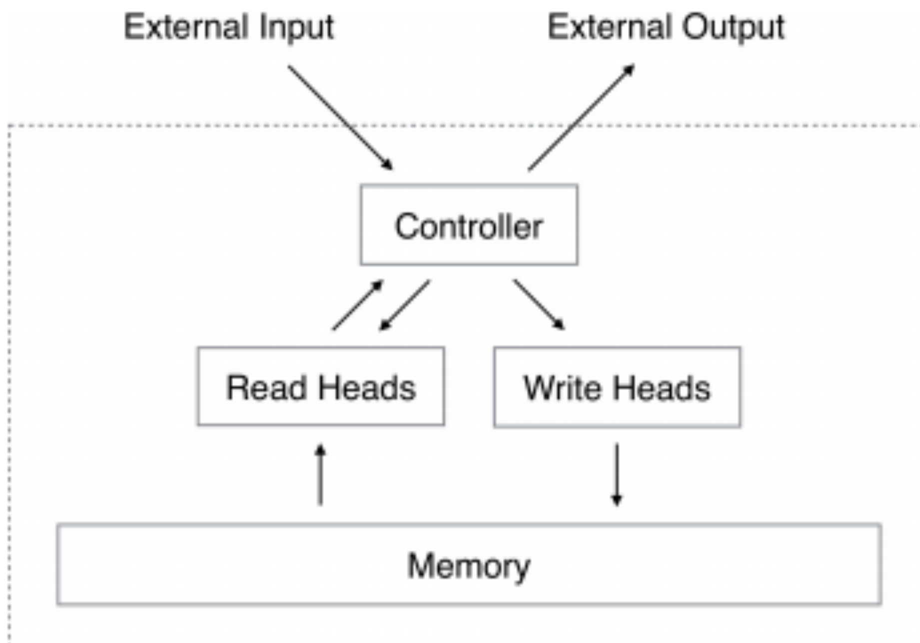
Foundational Research

- Neuroscience and Psychology
- Cognitive Science ad Linguistics
- Recurrent Neural networks
  - Broad class of machines with distributed and dynamic state
  - Long Short Term Memory RNN's designed to handle vanishing and exploding gradient
  - Natively handle variable length structures

Neural Turing Machines



Neural Turing Machines



Neural Turing Machines

1. Reading
   - M_t is NxM matrix of memory at time t

Neural Turing Machines

1. Reading
   - M_t is NxM matrix of memory at time t
   - w_t

$$\sum_i w_t(i) = 1, \qquad 0 \leq w_t(i) \leq 1, \forall i.$$

$$\mathbf{r}_t \longleftarrow \sum_i w_t(i)\mathbf{M}_t(i),$$

Neural Turing Machines

1. Reading
2. Writing involves both erasing and adding

$$\tilde{\mathbf{M}}_t(i) \longleftarrow \mathbf{M}_{t-1}(i)\left[\mathbf{1} - w_t(i)\mathbf{e}_t\right],$$

Neural Turing Machines

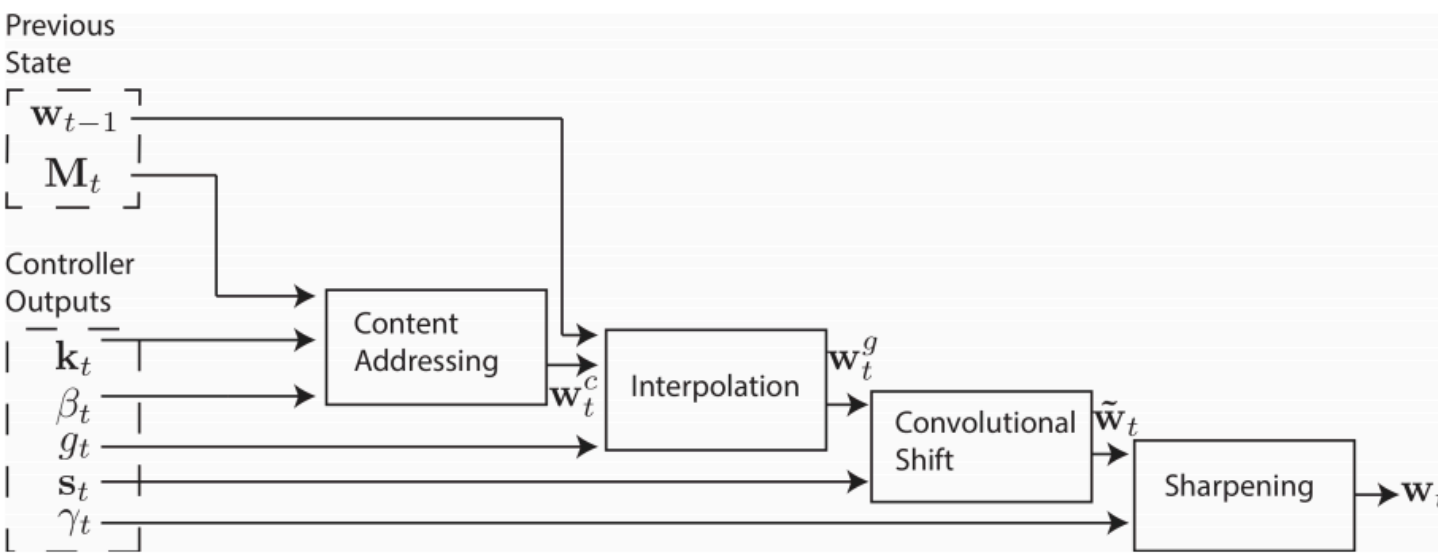
1. Reading
2. Writing involves both erasing and adding

$$\tilde{\mathbf{M}}_t(i) \longleftarrow \mathbf{M}_{t-1}(i)\left[\mathbf{1} - w_t(i)\mathbf{e}_t\right],$$

$$\mathbf{M}_t(i) \longleftarrow \tilde{\mathbf{M}}_t(i) + w_t(i)\,\mathbf{a}_t.$$

Neural Turing Machines

1. Reading
2. Writing involves both erasing and adding
3. Addressing



Neural Turing Machines

- 3. Addressing
  - 1. Focusing by Content
    - Each head produces key vector $\mathbf{k_t}$ of length M
    - Generated a content based weight $\mathbf{w_t}^{\mathbf{c}}$ based on similarity measure, using 'key strength' β_t

$$w_t^c(i) \longleftarrow \frac{\exp\Big(\beta_t K\big[\mathbf{k}_t, \mathbf{M}_t(i)\big]\Big)}{\sum_j \exp\Big(\beta_t K\big[\mathbf{k}_t, \mathbf{M}_t(j)\big]\Big)}.$$

$$K\big[\mathbf{u}, \mathbf{v}\big] = \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||}.$$

Neural Turing Machines

- 3. Addressing
  - 2. Interpolation
    - Each head emits a scalar interpolation gate g_t

$$\mathbf{w}_t^g \longleftarrow g_t \mathbf{w}_t^c + (1 - g_t)\mathbf{w}_{t-1}.$$

Neural Turing Machines

- 3. Addressing
  - 3. Convolutional shift
    - Each head emits a distribution over allowable integer shifts $\mathbf{s_t}$

$$\tilde{w}_t(i) \longleftarrow \sum_{j=0}^{N-1} w_t^g(j)\, s_t(i-j)$$

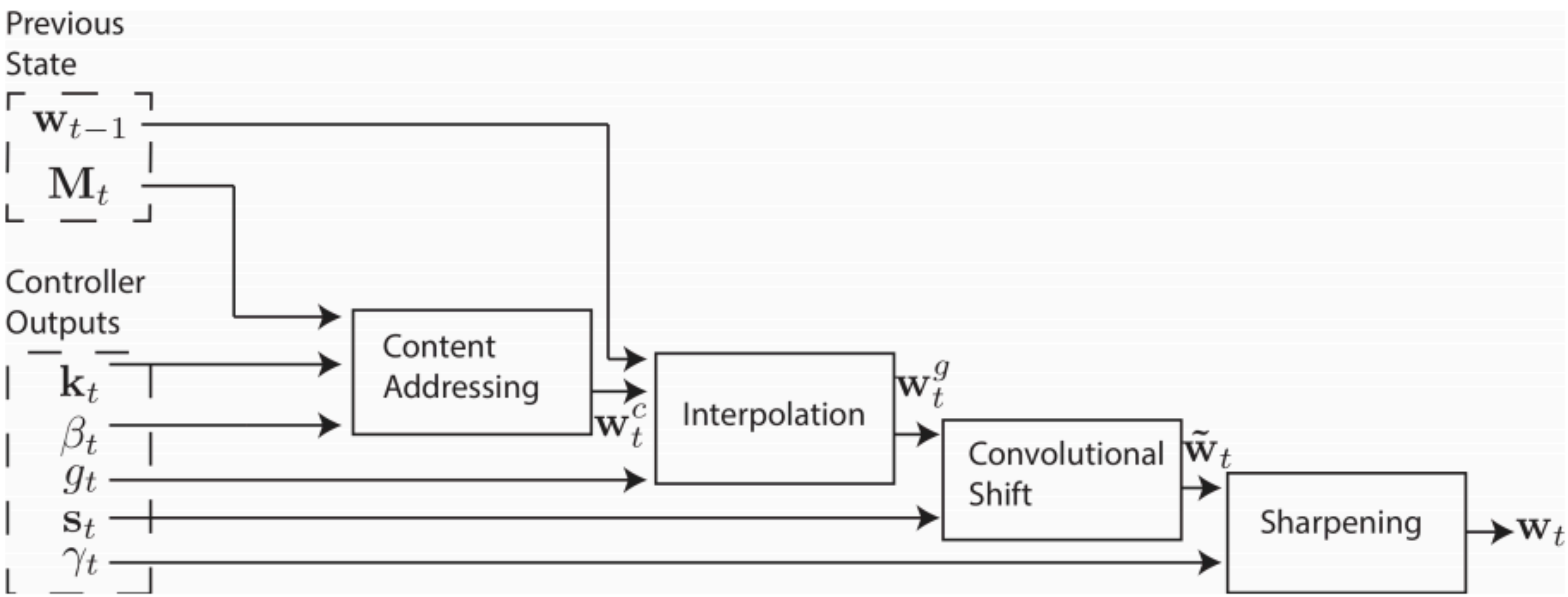
Neural Turing Machines

- 3. Addressing
  - 4. Sharpening
    - Each head emits a scalar sharpening parameter γ_t

$$w_t(i) \longleftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_j \tilde{w}_t(j)^{\gamma_t}}$$

Neural Turing Machines

- 3. Addressing (putting it all together)



Neural Turing Machines

- 3. Addressing (putting it all together)
  - This can operate in three complementary modes
    - A weighting can be chosen by the content system without any modification by the location system

Neural Turing Machines

- 3. Addressing (putting it all together)
  - This can operate in three complementary modes
    - A weighting can be chosen by the content system without any modification by the location system
    - A weighting produced by the content addressing system can be chosen and then shifted

Neural Turing Machines

- 3. Addressing (putting it all together)
  - This can operate in three complementary modes
    - A weighting can be chosen by the content system without any modification by the location system
    - A weighting produced by the content addressing system can be chosen and then shifted
    - A weighting from the previous time step can be rotated without any input from the content-based addressing system

Neural Turing Machines

- Controller Network Architecture
  - Feed Forward vs Recurrent

Neural Turing Machines

- Controller Network Architecture
  - Feed Forward vs Recurrent
  - The LSTM version of RNN has own internal memory complementary to M

Neural Turing Machines

- Controller Network Architecture
  - Feed Forward vs Recurrent
  - The LSTM version of RNN has own internal memory complementary to M
  - Hidden LSTM layers are ‘like’ registers in processor

Neural Turing Machines

- Controller Network Architecture
  - Feed Forward vs Recurrent
  - The LSTM version of RNN has own internal memory complementary to M
  - Hidden LSTM layers are ‘like’ registers in processor
  - Allows for mix of information across multiple time-steps

Neural Turing Machines

- Controller Network Architecture
  - Feed Forward vs Recurrent
  - The LSTM version of RNN has own internal memory complementary to M
  - Hidden LSTM layers are ‘like’ registers in processor
  - Allows for mix of information across multiple time-steps
  - Feed Forward has better transparency

Experiments

- Test NTM's ability to learn simple algorithms like copying and sorting

Experiments

- Test NTM's ability to learn simple algorithms like copying and sorting
- Demonstrate that solutions generalize well beyond the range of training

Experiments

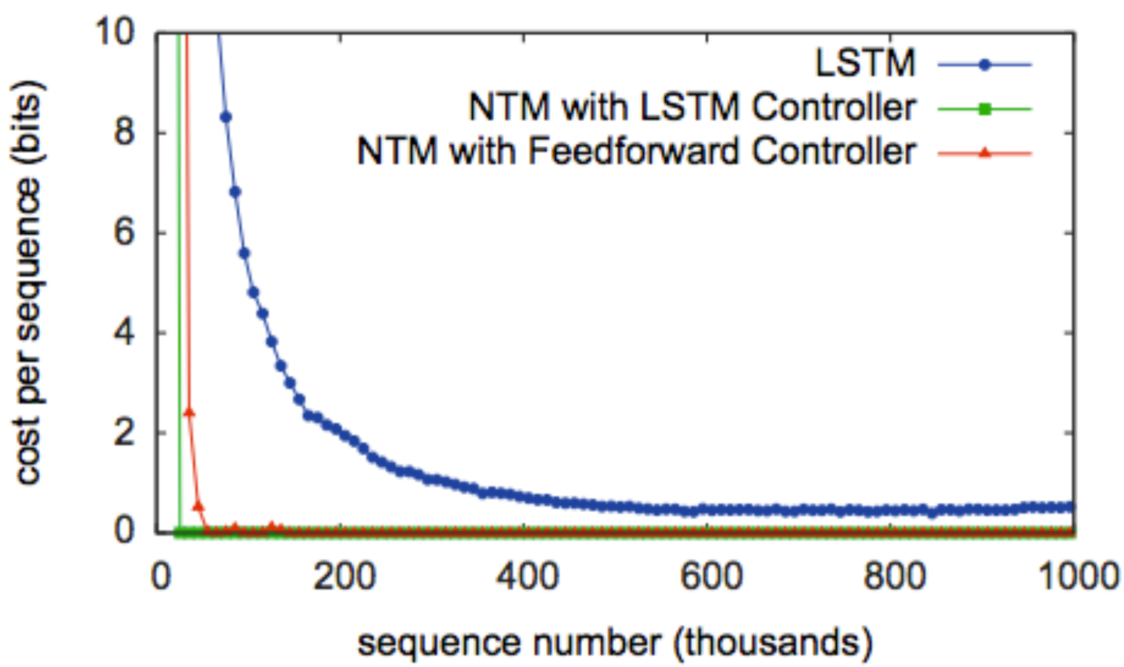
- Test NTM's ability to learn simple algorithms like copying and sorting
- Demonstrate that solutions generalize well beyond the range of training
- Tests three architectures
  - NTM with feed forward controller
  - NTM with LSTM controller
  - Standard LSTM network

Experiments

- 1. Copy
  - Tests whether NTM can store and retrieve data
  - Trained to copy sequences of 8 bit vectors
  - Sequences vary between 1-20 vectors

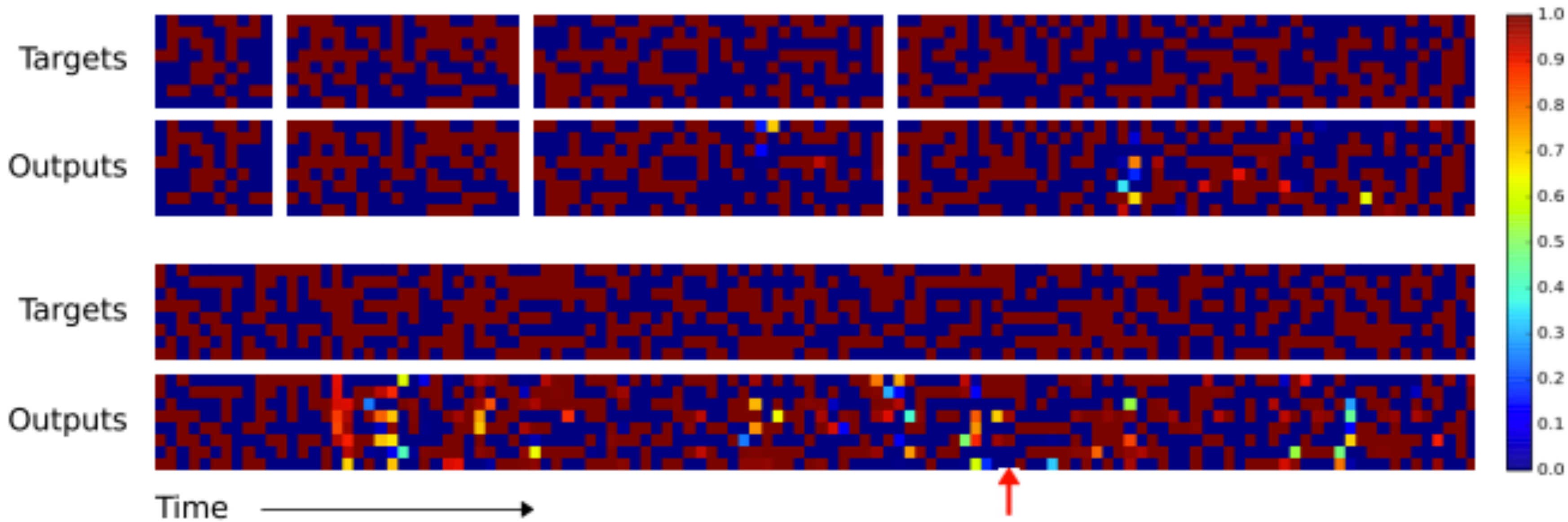
Experiments

- 1. Copy



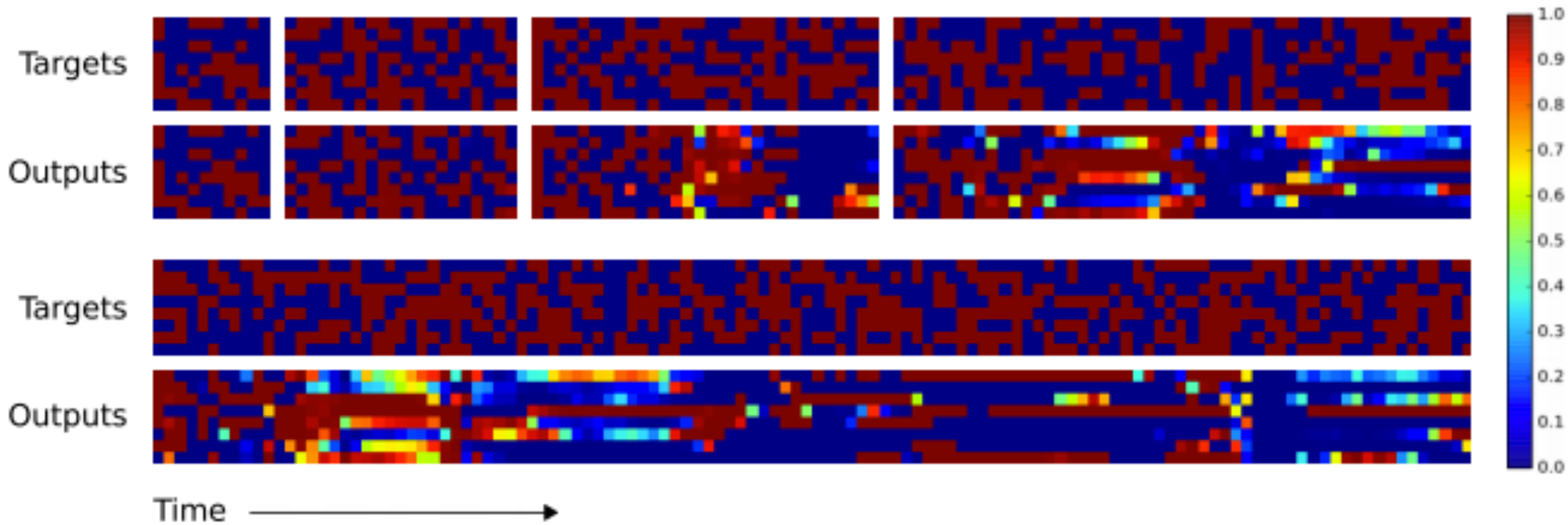
Experiments

- 1. Copy
  - NTM



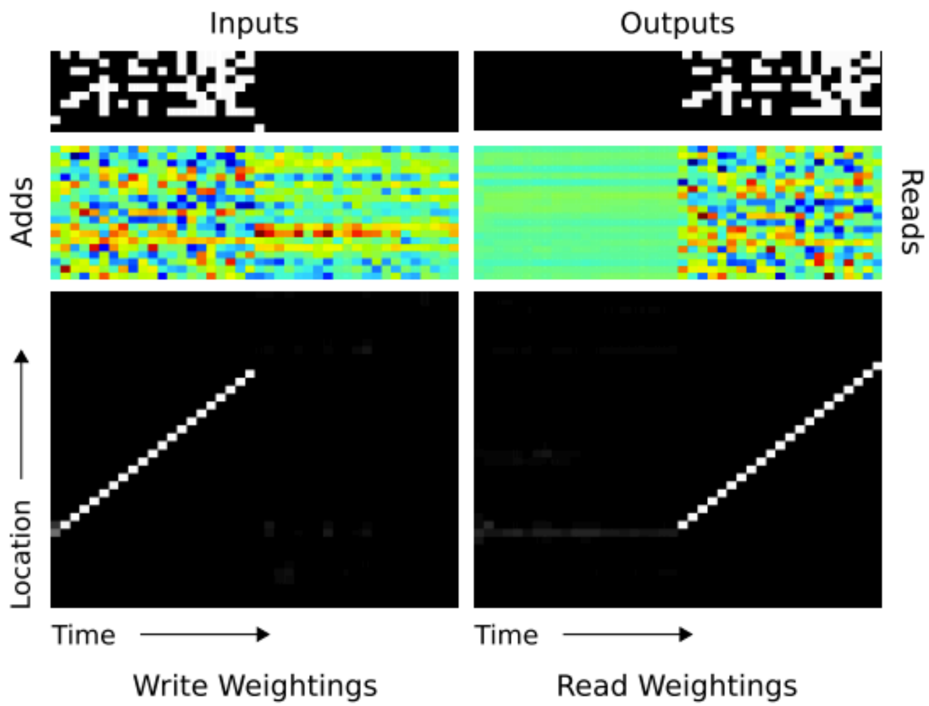
Experiments

- 1. Copy
  - LSTM



Experiments

- 1. Copy



Experiments

- 2. Repeat Copy
  - Tests whether NTM can learn simple nested function
  - Extend copy by repeatedly copying input specified number of times
  - Training is a random-length sequence of 8 bit binary inputs plus a scalar value for # of copies
  - Scalar value is random between 1-10

Experiments

- 2. Repeat Copy

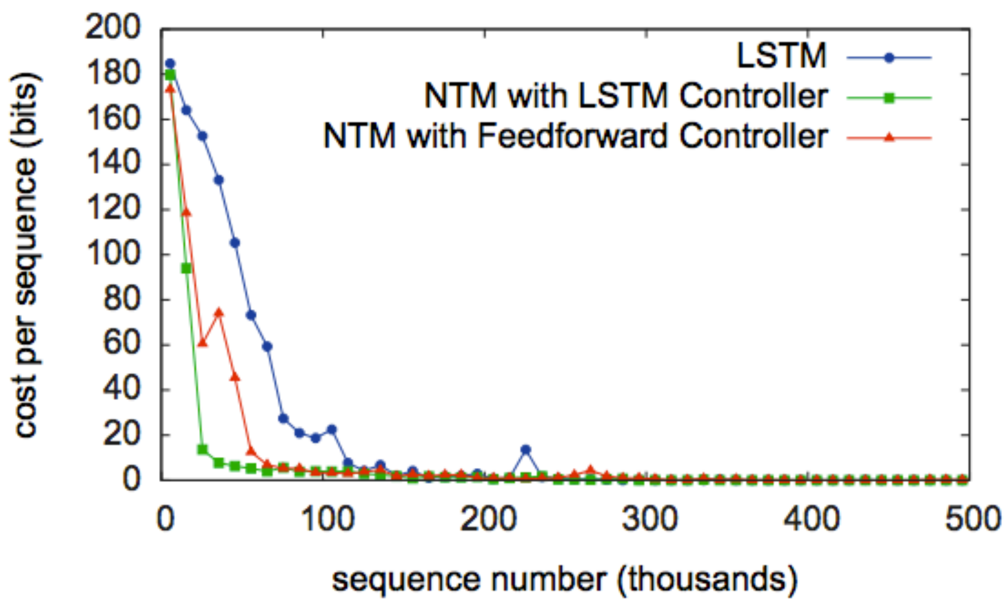
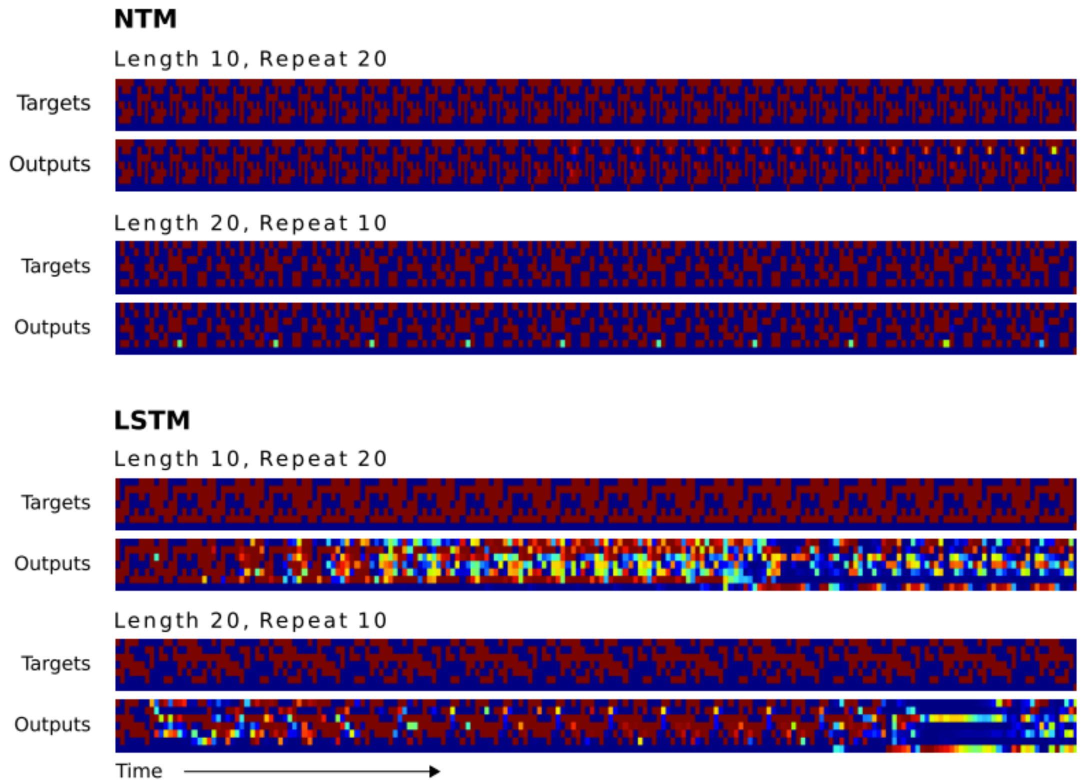


Figure 7: Repeat Copy Learning Curves.

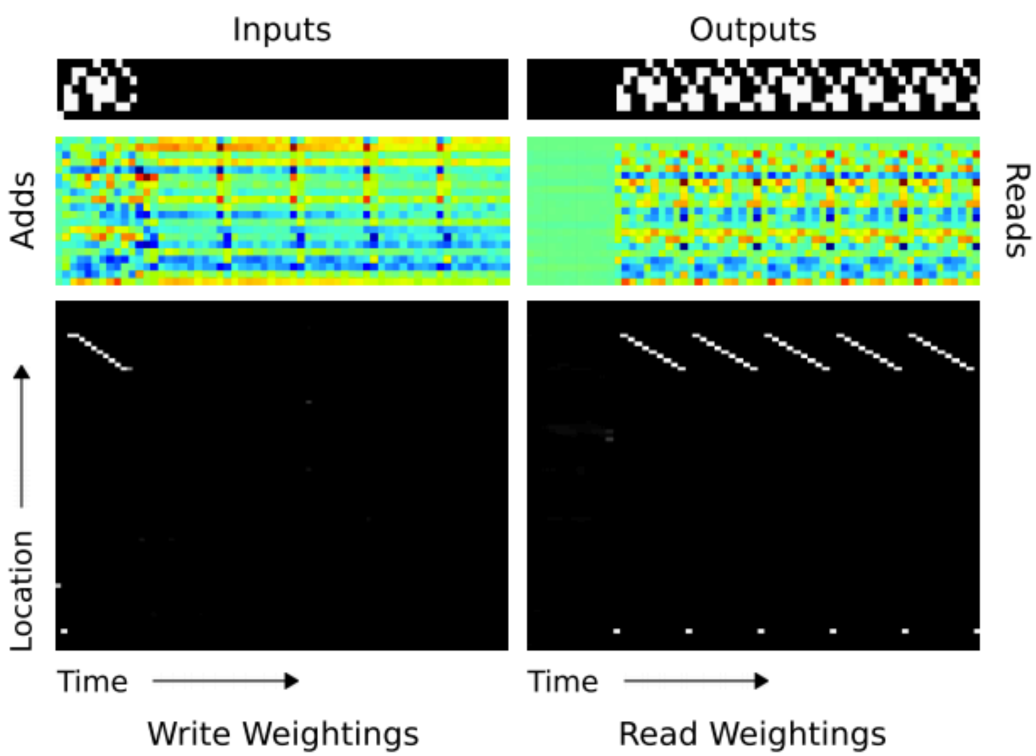
Experiments

- 2. Repeat Copy



Experiments

- 2. Repeat Copy



Experiments

- 3. Associative Recall
  - Tests NTM's ability to associate data references
  - Training input is list of items, followed by a query item
  - Output is subsequent item in list
  - Each item is a three sequence 6-bit binary vector
  - Each 'episode' has between two and six items

Experiments

- 3. Associative Recall

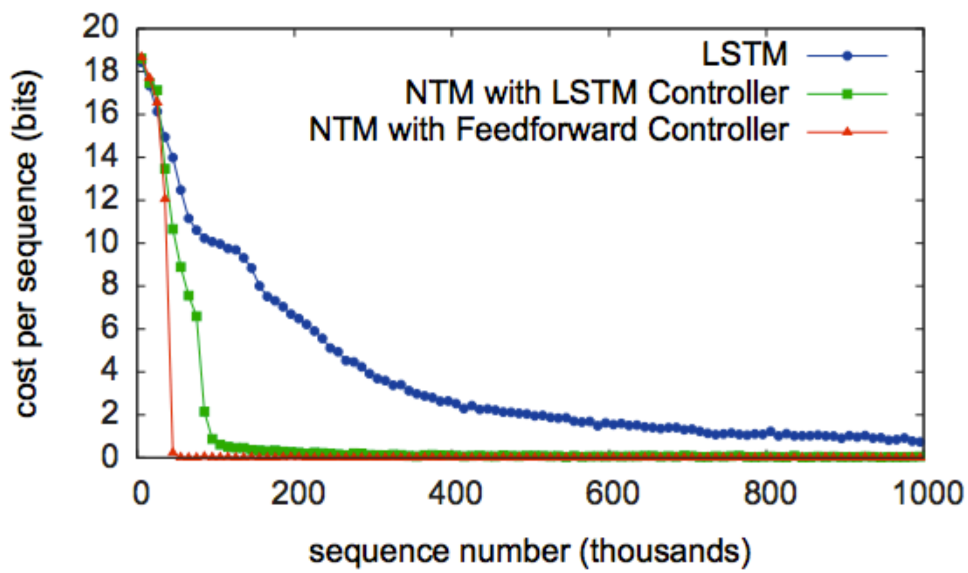


Figure 10: Associative Recall Learning Curves for NTM and LSTM.

Experiments

- 3. Associative Recall

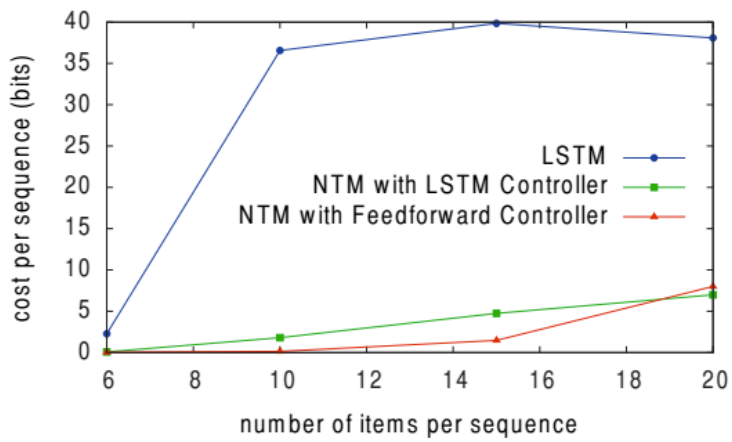
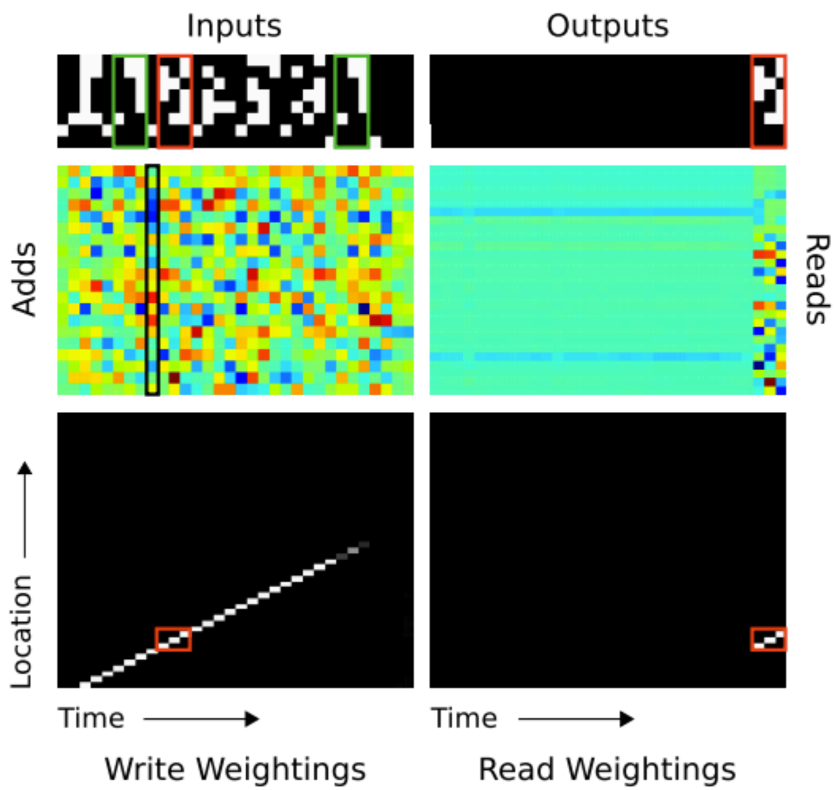


Figure 11: Generalisation Performance on Associative Recall for Longer Item Sequences. The NTM with either a feedforward or LSTM controller generalises to much longer sequences of items than the LSTM alone. In particular, the NTM with a feedforward controller is nearly perfect for item sequences of twice the length of sequences in its training set.

Experiments

- 3. Associative Recall



Experiments

- 4. Dynamic N-Grams
  - Test whether NTM could rapidly adapt to new predictive distributions
  - Trained on 6-gram binary pattern on 200 bit sequences
  - Can NTM learn optimal estimator

$$P(B = 1|N_1, N_0, \mathbf{c}) = \frac{N_1 + \frac{1}{2}}{N_1 + N_0 + 1}$$

Experiments

- 4. Dynamic N-Grams

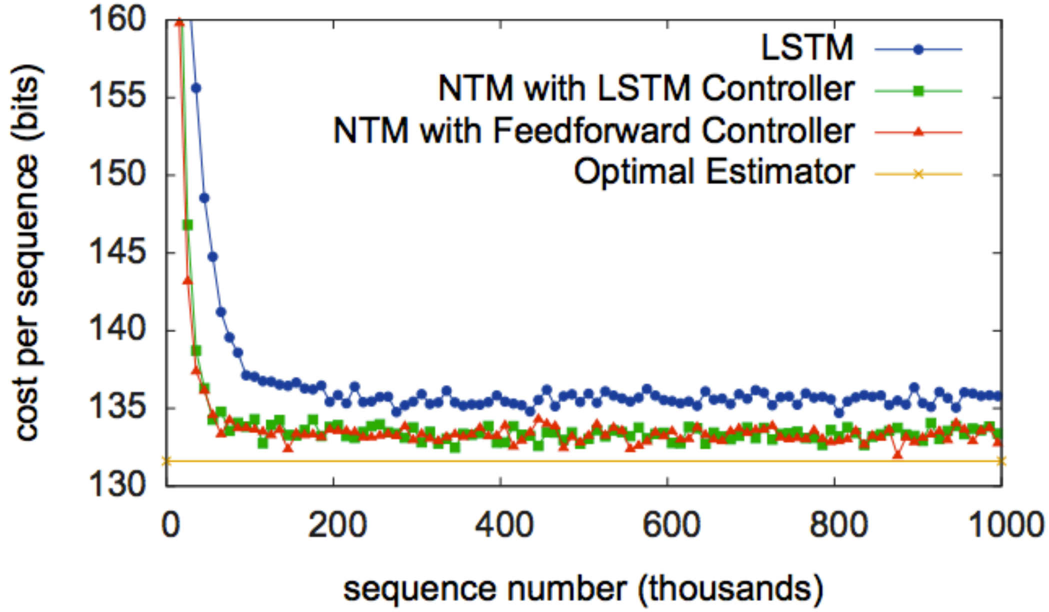
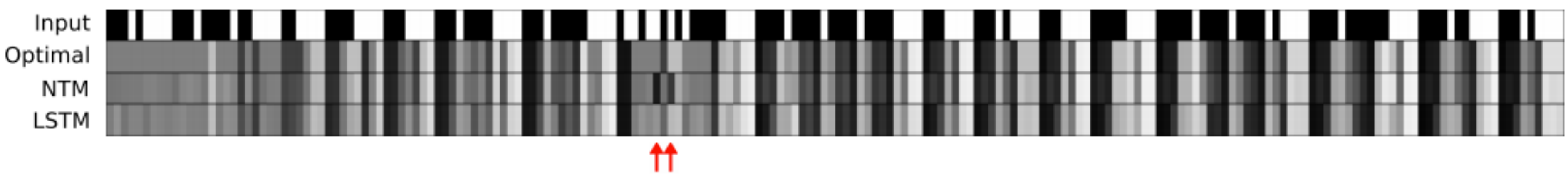


Figure 13: Dynamic N-Gram Learning Curves.

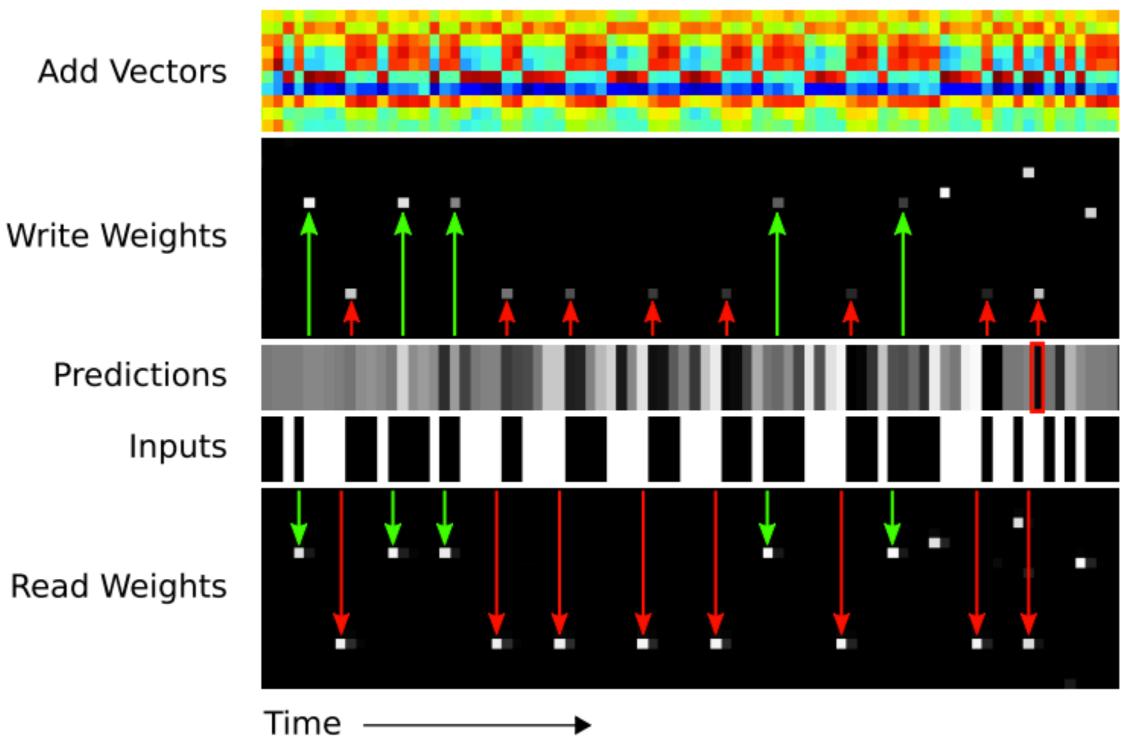
Experiments

- 4. Dynamic N-Grams



Experiments

- 4. Dynamic N-Grams

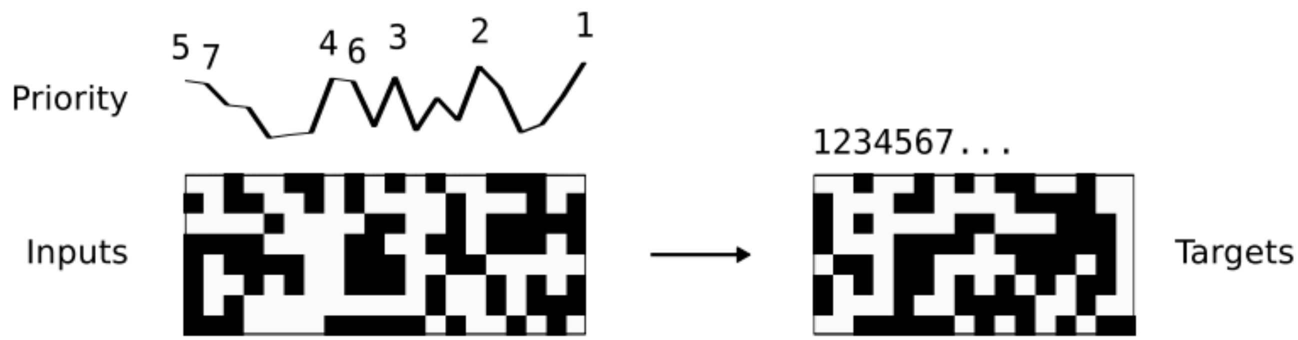


Experiments

- 5. Priority Sort
  - Tests whether NTM can sort data
  - Input is sequence of 20 random binary vectors, each with a scalar rating drawn from [-1, 1]
  - Target sequence is 16-highest priority vectors

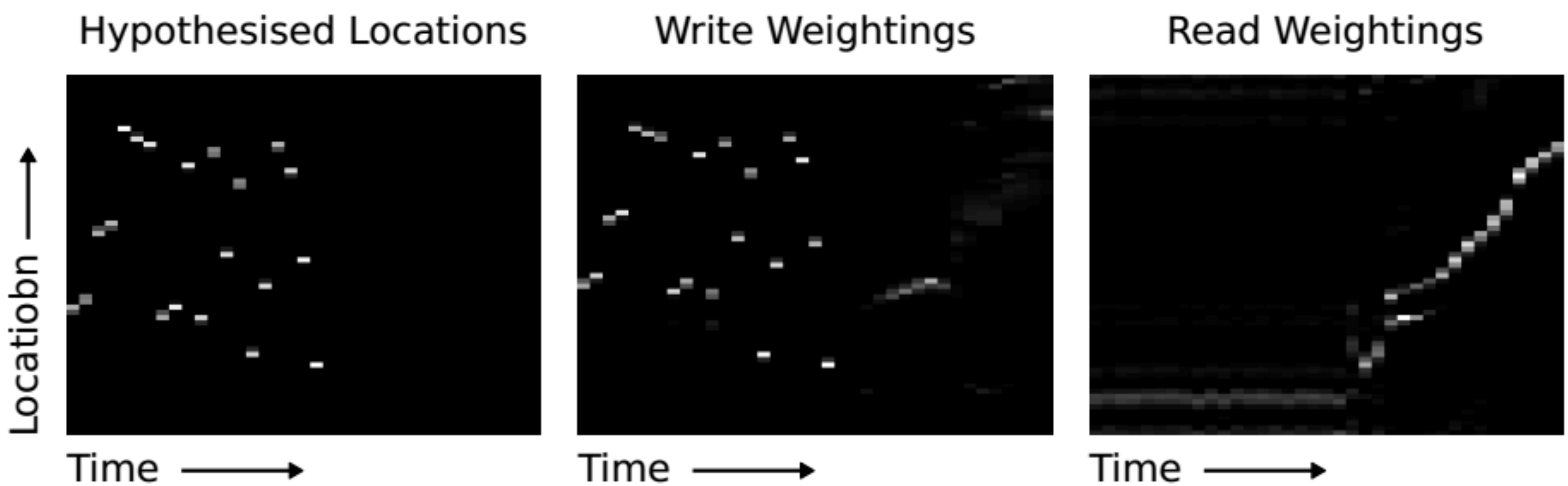
Experiments

- 5. Priority Sort



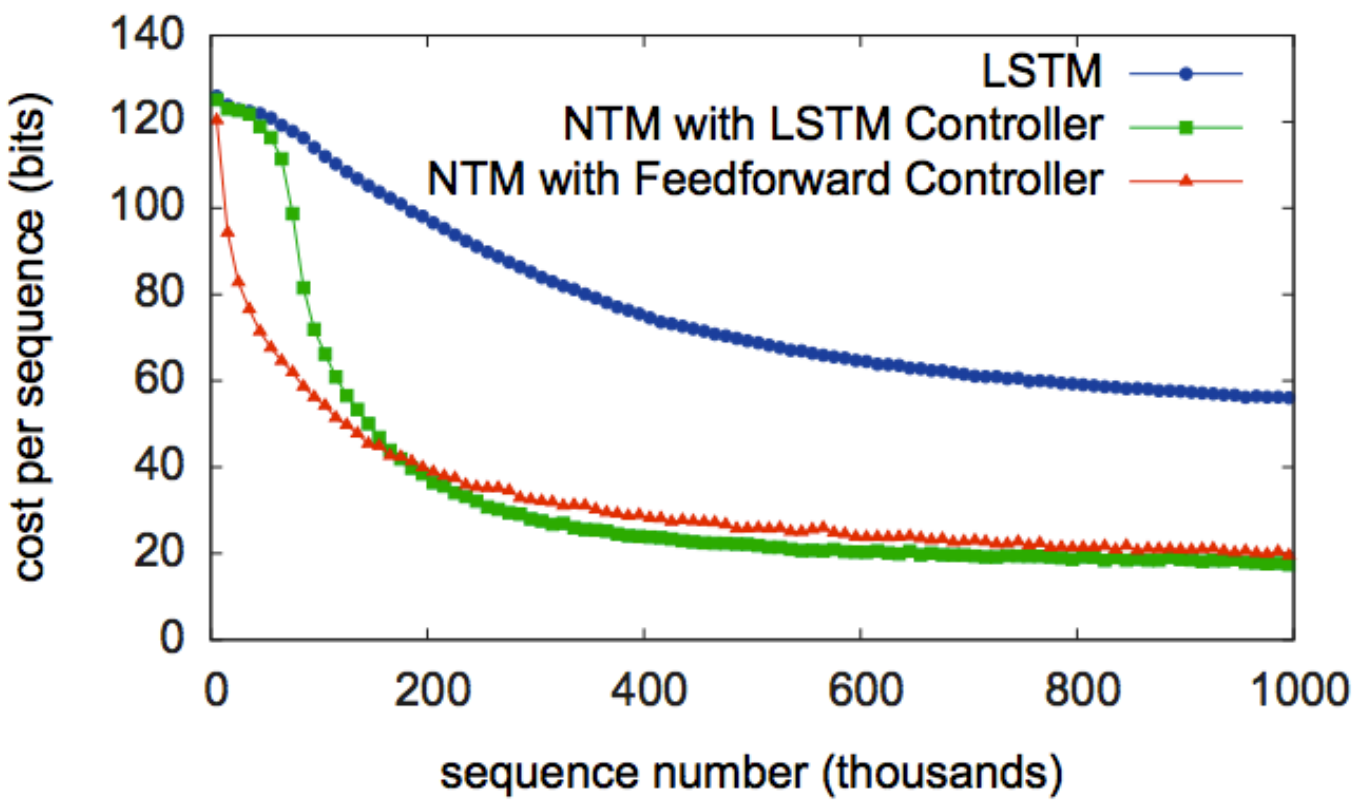
Experiments

- 5. Priority Sort



Experiments

- 5. Priority Sort



Experiments

- 6. Details
  - RMSProp algorithm
  - Momentum 0.9
  - All LSTM’s had three stacked hidden layers

Experiments

- 6. Details

Task	#Heads	Controller Size	Memory Size	Learning Rate	#Parameters
Copy	1	100	128×20	10^{-4}	$17,162$
Repeat Copy	1	100	128×20	10^{-4}	$16,712$
Associative	4	256	128×20	10^{-4}	$146,845$
N-Grams	1	100	128×20	3×10^{-5}	$14,656$
Priority Sort	8	512	128×20	3×10^{-5}	$508,305$

Table 1: NTM with Feedforward Controller Experimental Settings

Experiments

- 6. Details

Task	#Heads	Controller Size	Memory Size	Learning Rate	#Parameters
Copy	1	100	128×20	10^{-4}	$67,561$
Repeat Copy	1	100	128×20	10^{-4}	$66,111$
Associative	1	100	128×20	10^{-4}	$70,330$
N-Grams	1	100	128×20	3×10^{-5}	$61,749$
Priority Sort	5	2×100	128×20	3×10^{-5}	$269,038$

Table 2: NTM with LSTM Controller Experimental Settings

Experiments

- 6. Details

Task	Network Size	Learning Rate	#Parameters
Copy	3×256	3×10^{-5}	$1,352,969$
Repeat Copy	3×512	3×10^{-5}	$5,312,007$
Associative	3×256	10^{-4}	$1,344,518$
N-Grams	3×128	10^{-4}	$331,905$
Priority Sort	3×128	3×10^{-5}	$384,424$

Table 3: LSTM Network Experimental Settings

Conclusion

- Introduced an neural net architecture with external memory that is differentiable end-to-end
- Experiments demonstrate that NTM are capable of leaning simple algorithms and are capable of generalizing beyond training regime

“ *Again, it [the Analytical Engine] might act upon other things besides numbers… the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.* ” — Ada Lovelace