Online slides: http://bit.ly/nnstruct
Password: bayes

Neural networks.
from: LSTM
to: Neural Computer

Daniil Polykovskiy
CMC MSU
16 December 2016

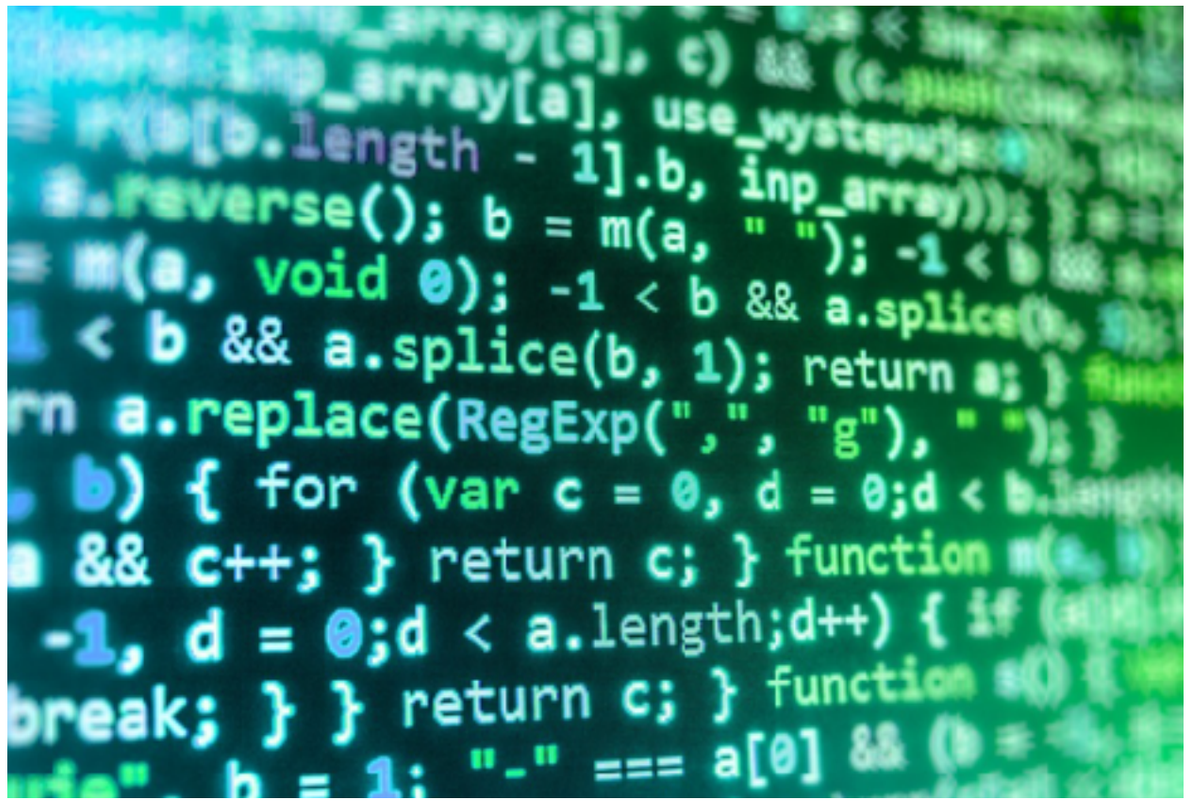
Goal

Narrow the gap between

1) *Neural networks*
2) *Algorithms*

Neural networks

Algorithms



✓ Cat-Dog discrimination ✗

✗ Sequence sorting ✓

Algorithms side: extract features

This is a **dog**

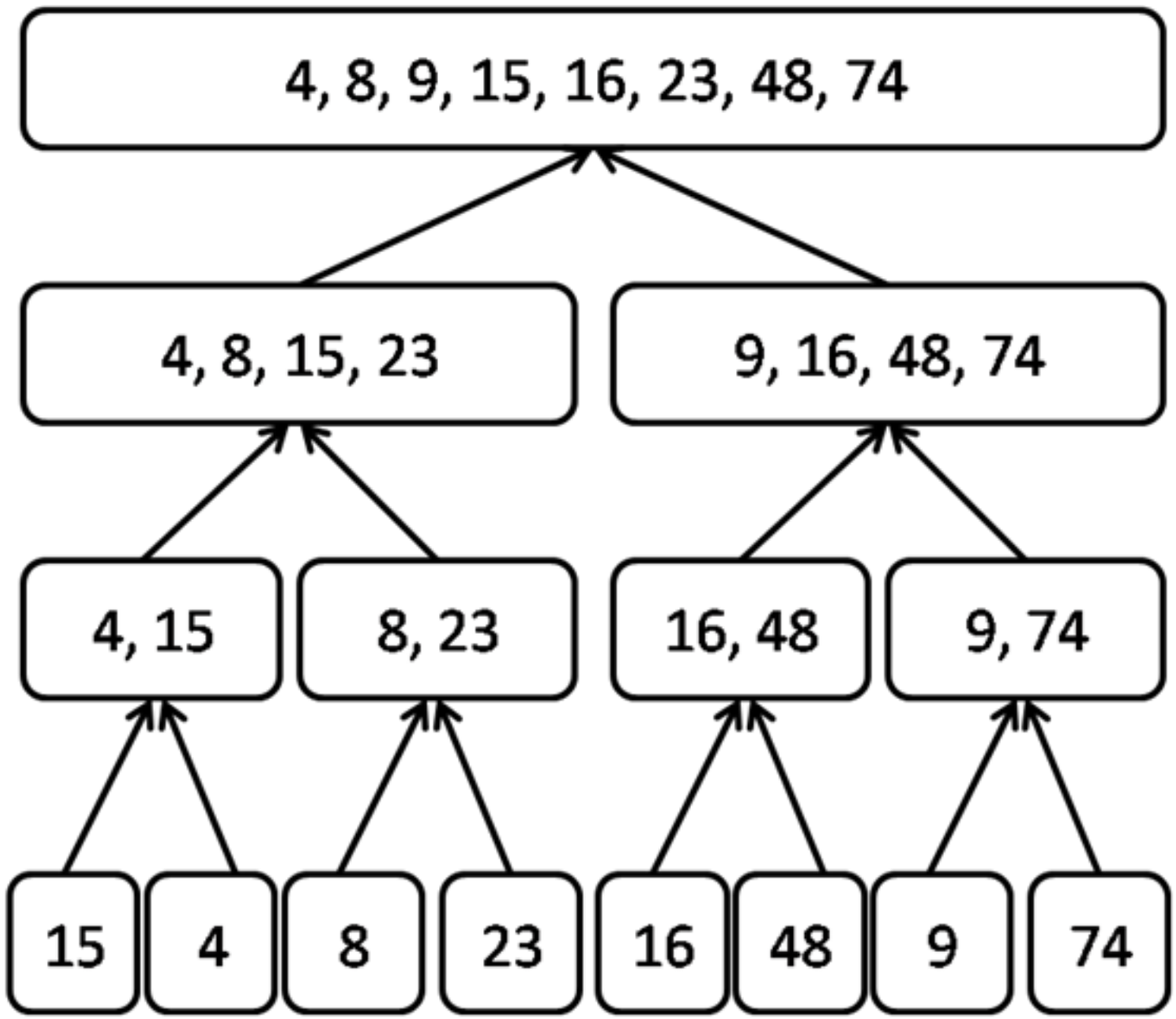
Lop-eared

Pointy ears

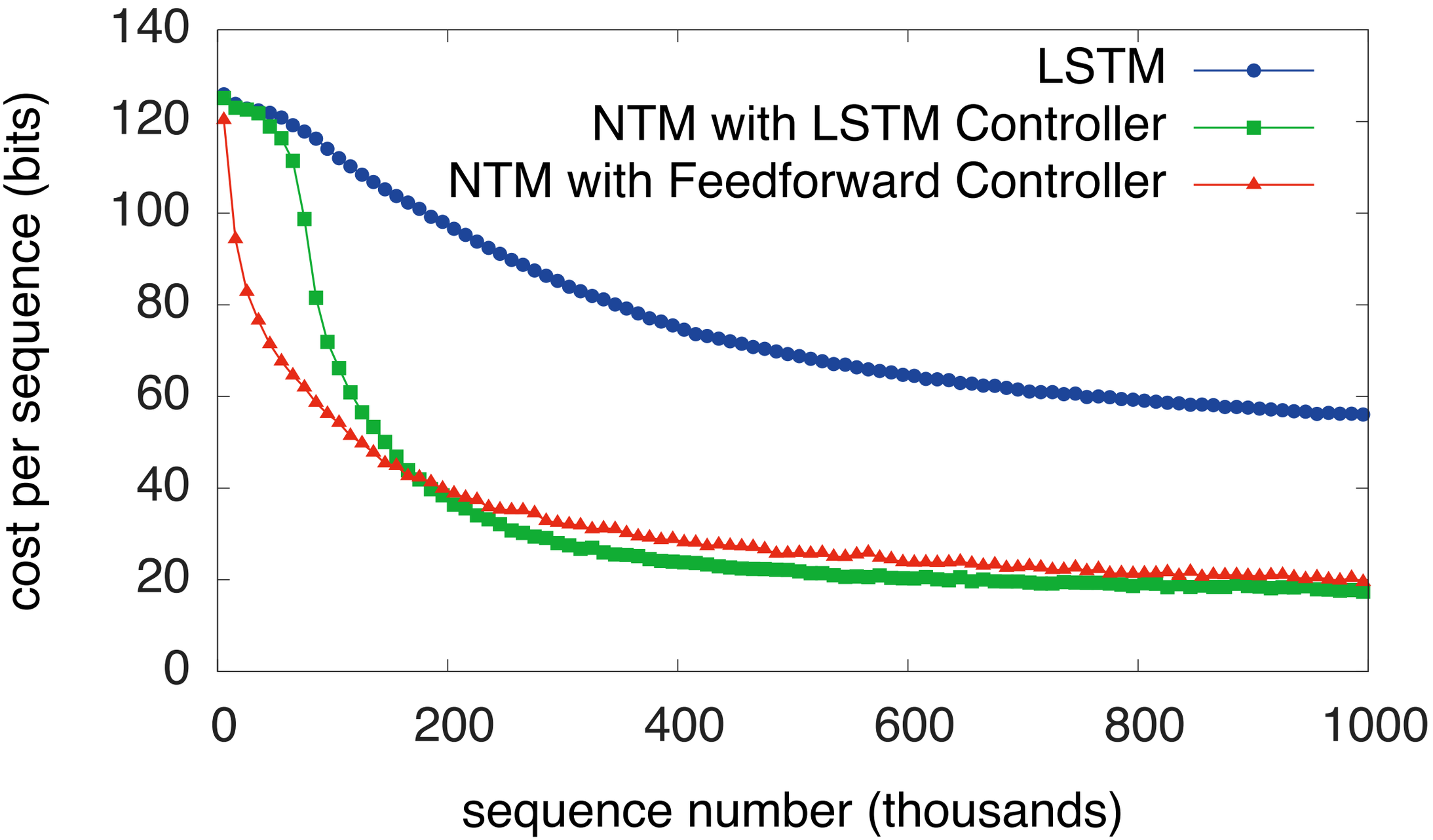
This is a **cat**

Sorting: easy case

Sorting: hard case



Appetizer

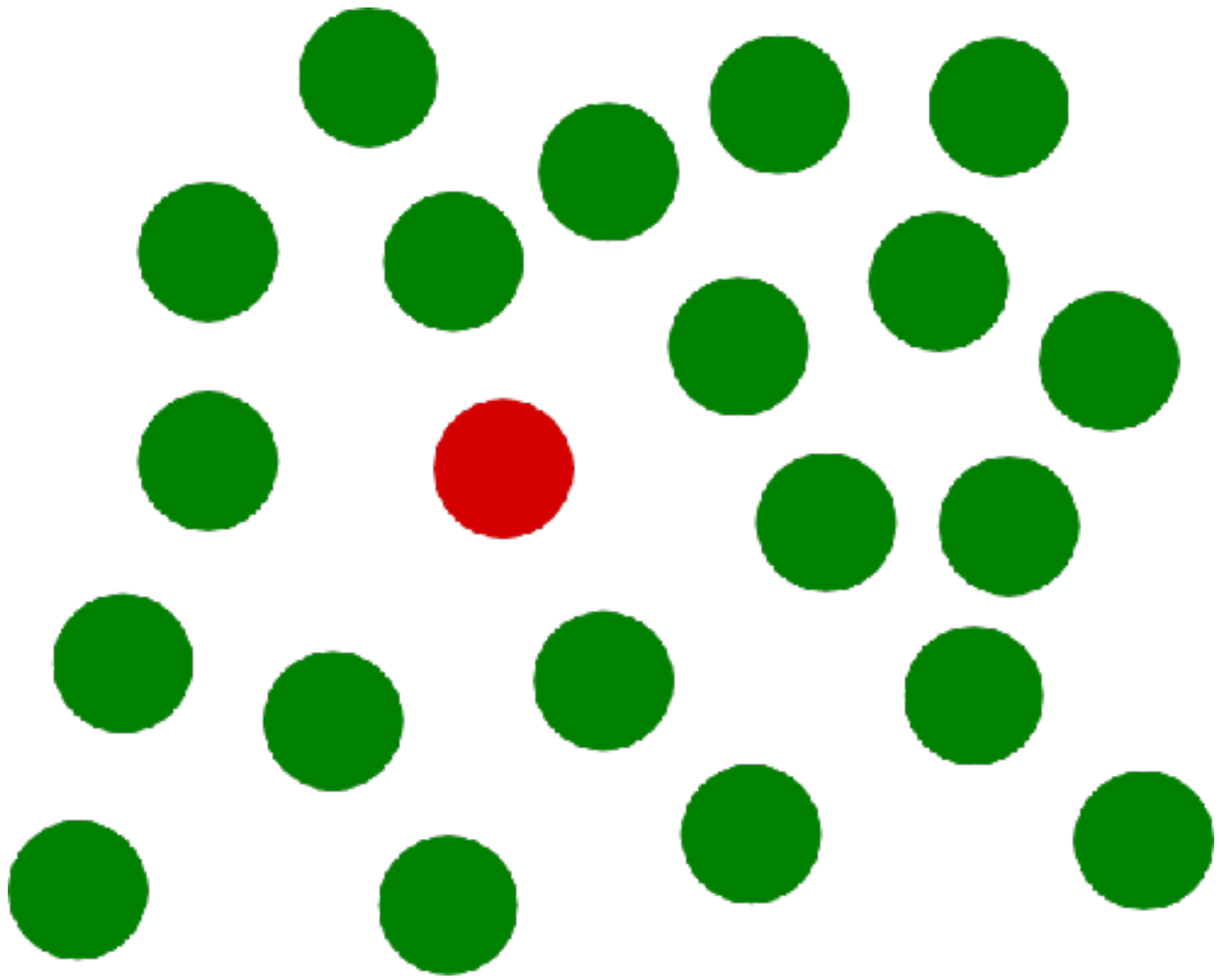

LSTM
NTM with LSTM Controller
NTM with Feedforward Controller
cost per sequence (bits)
0
20
40
60
80
100
120
140
0
200
400
600
800
1000
sequence number (thousands)

Outline

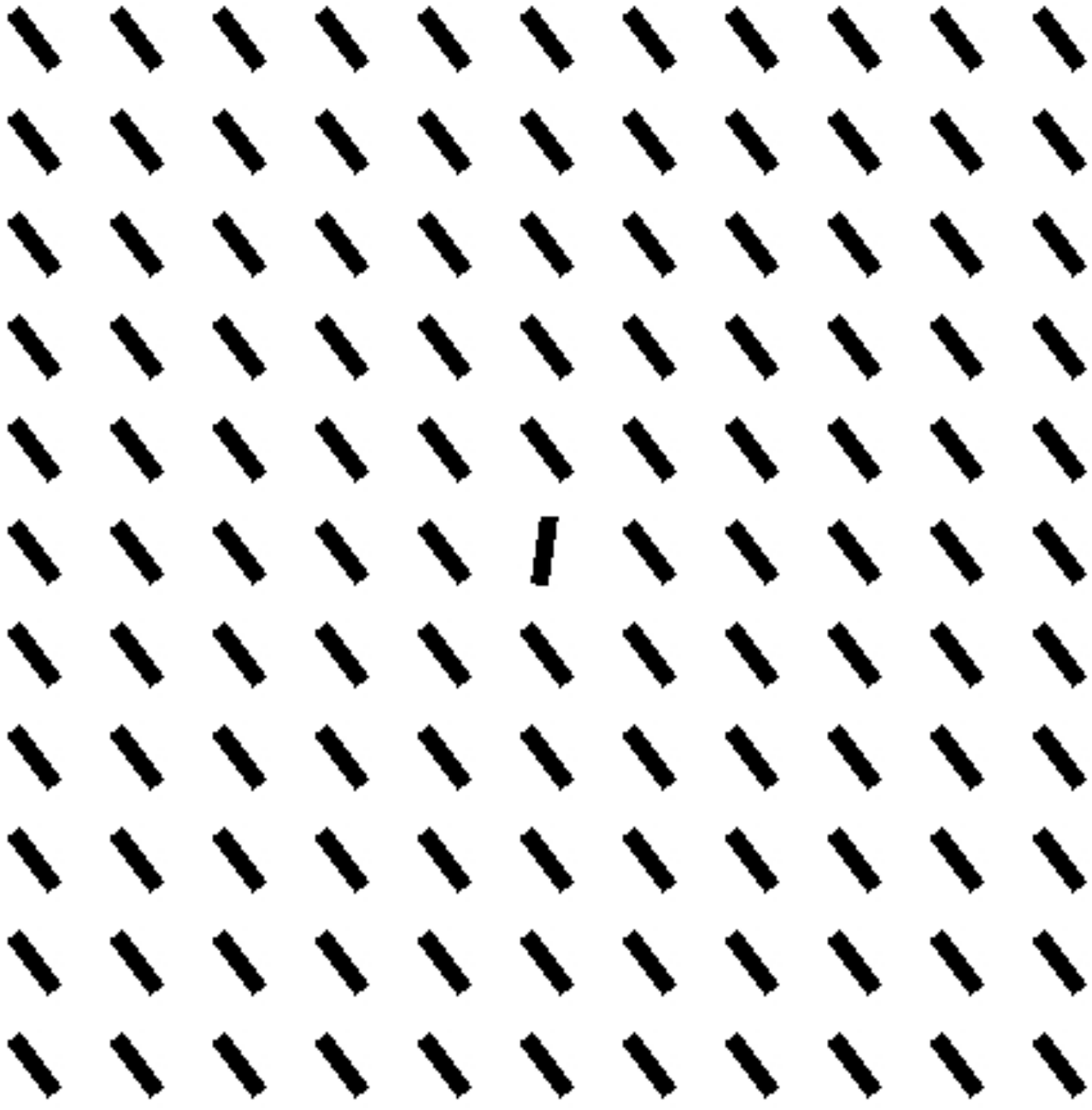
1. Attention
2. Neural turing machines
3. Neural stacks, lists, etc.
4. Differentiable Neural Computer

Attention

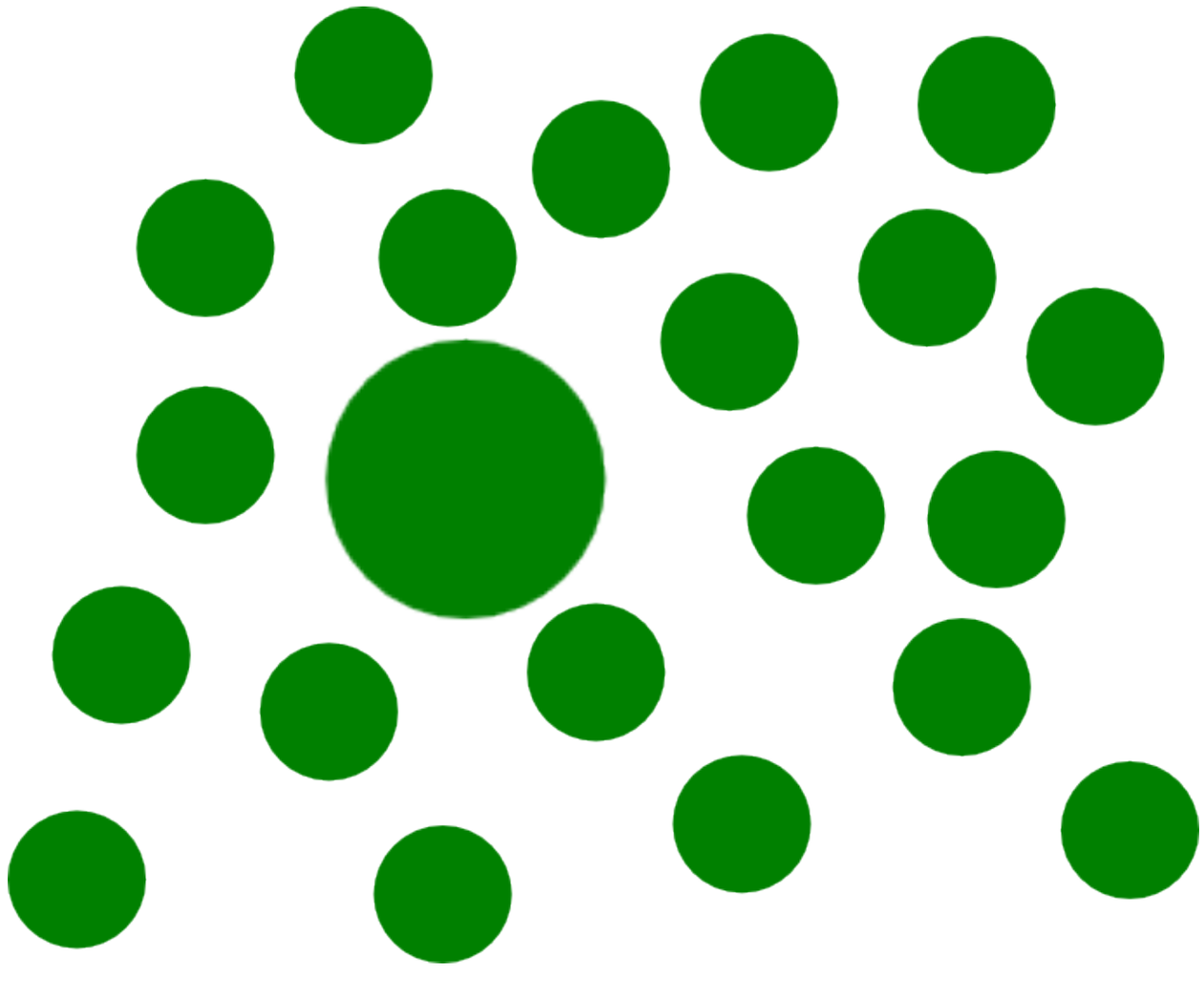
Salient regions



Salient regions



Salient regions



Attention: idea

- Auxiliary network generates distribution over places for attention (regions for images, words for texts)

- Return expectation over attended places.

0.033 0.9 0.033 0.034

The red boat sank.
What color was the boat?

$$r = e_{\text{The}} \cdot 0.033 + e_{\text{red}} \cdot 0.9 + e_{\text{boat}} \cdot 0.033 + e_{\text{sank}} \cdot 0.034$$

Attention: idea

- Auxiliary network generates distribution over places for attention (regions for images, words for texts)

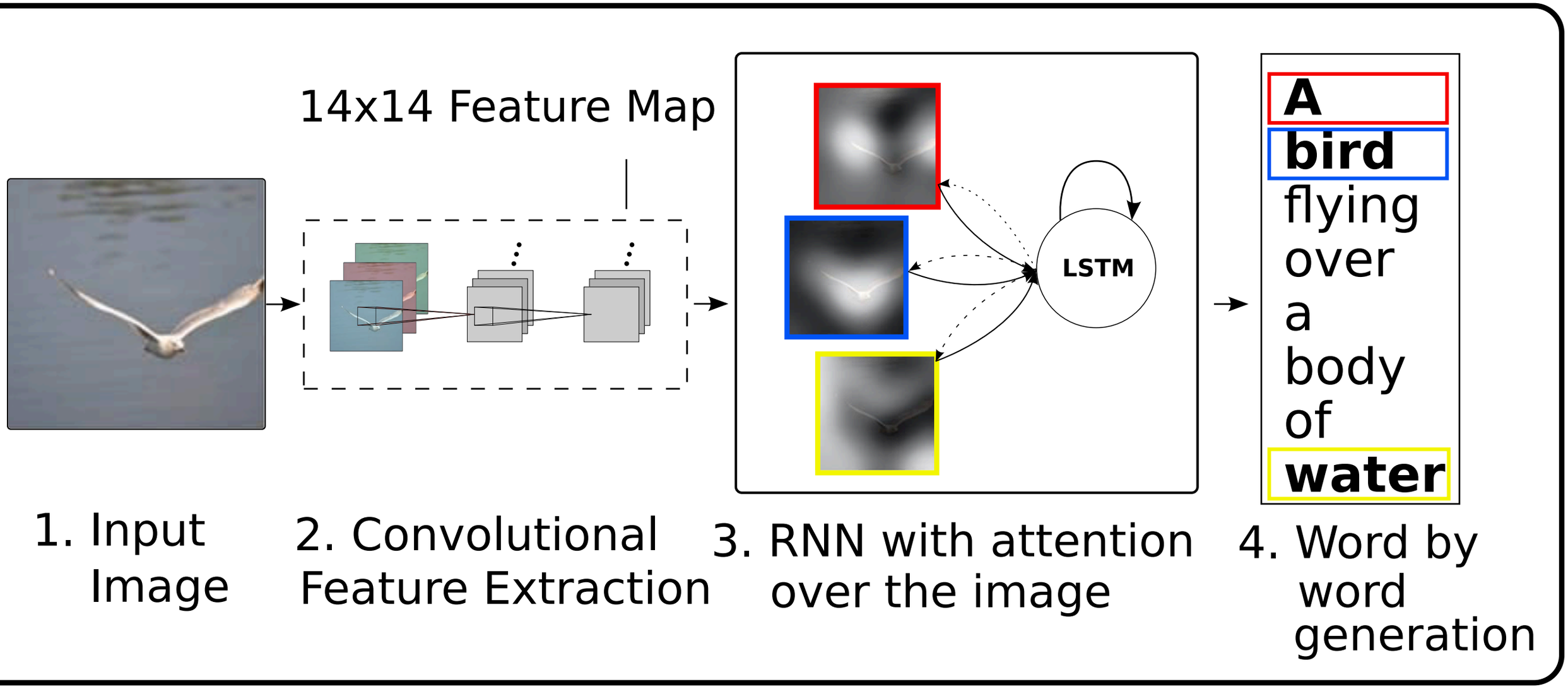
- Return expectation over attended places.

Attention: idea

- Auxiliary network generates distribution over places for attention (regions for images, words for texts)

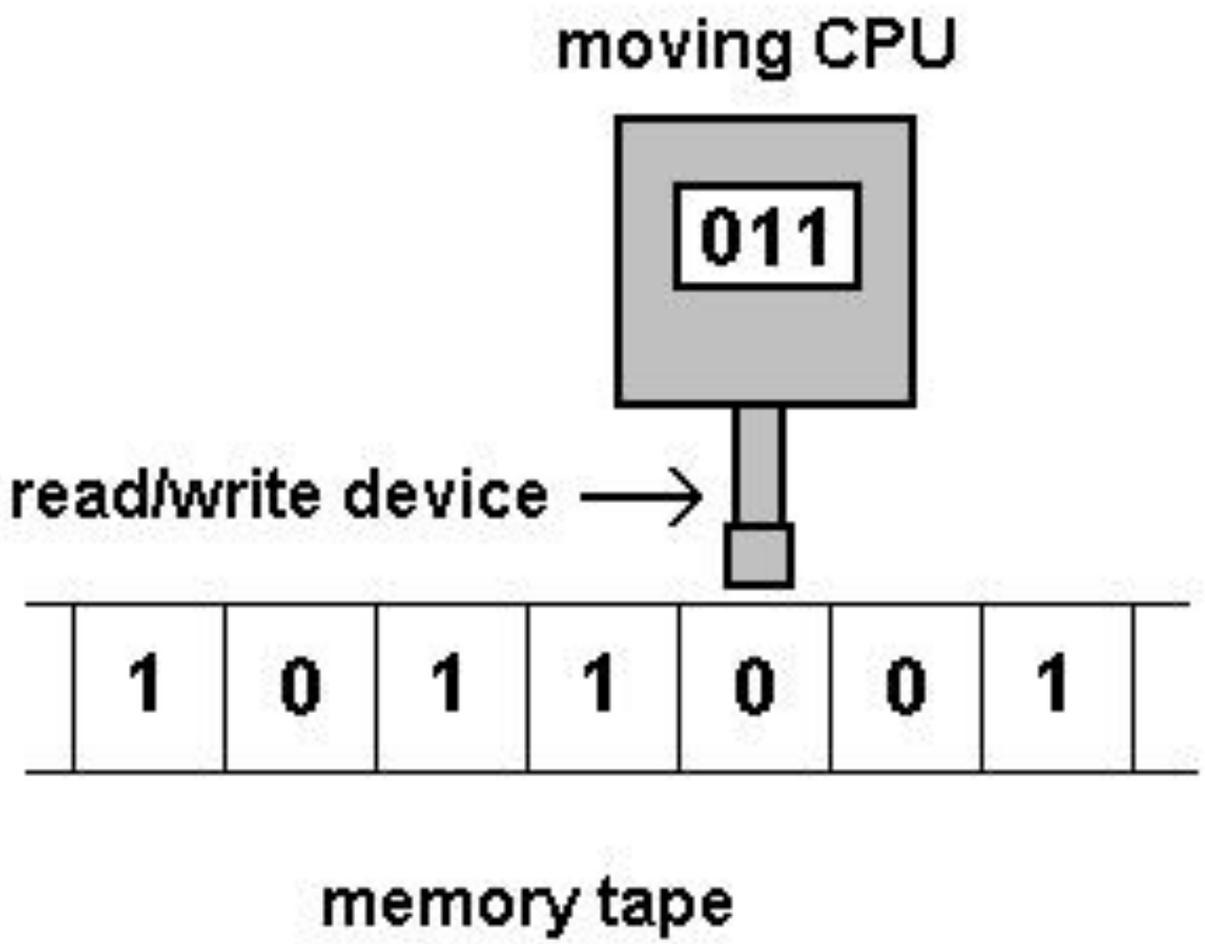
- Return expectation over attended places.

Attention: image captioning

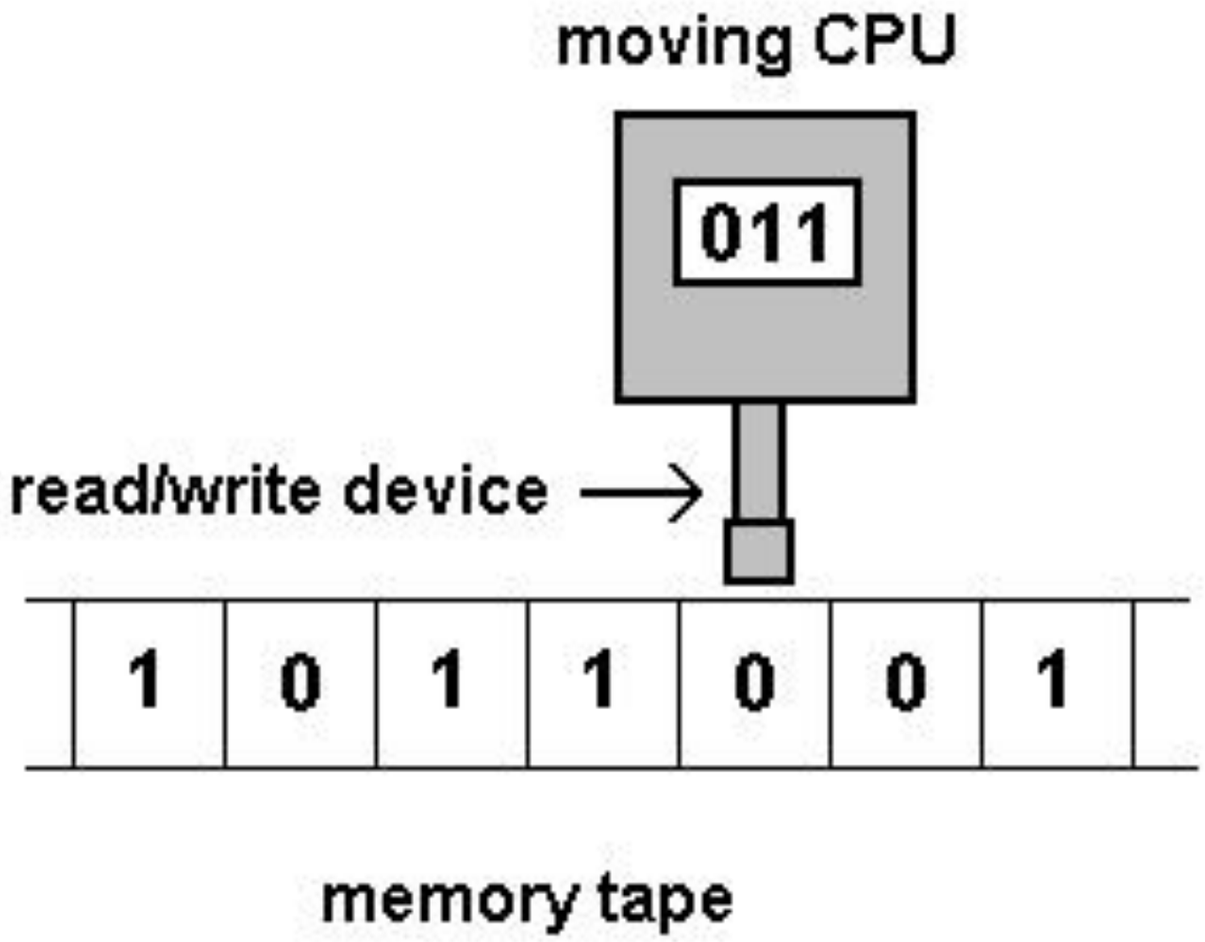


Neural Turing Machines

Turing machines

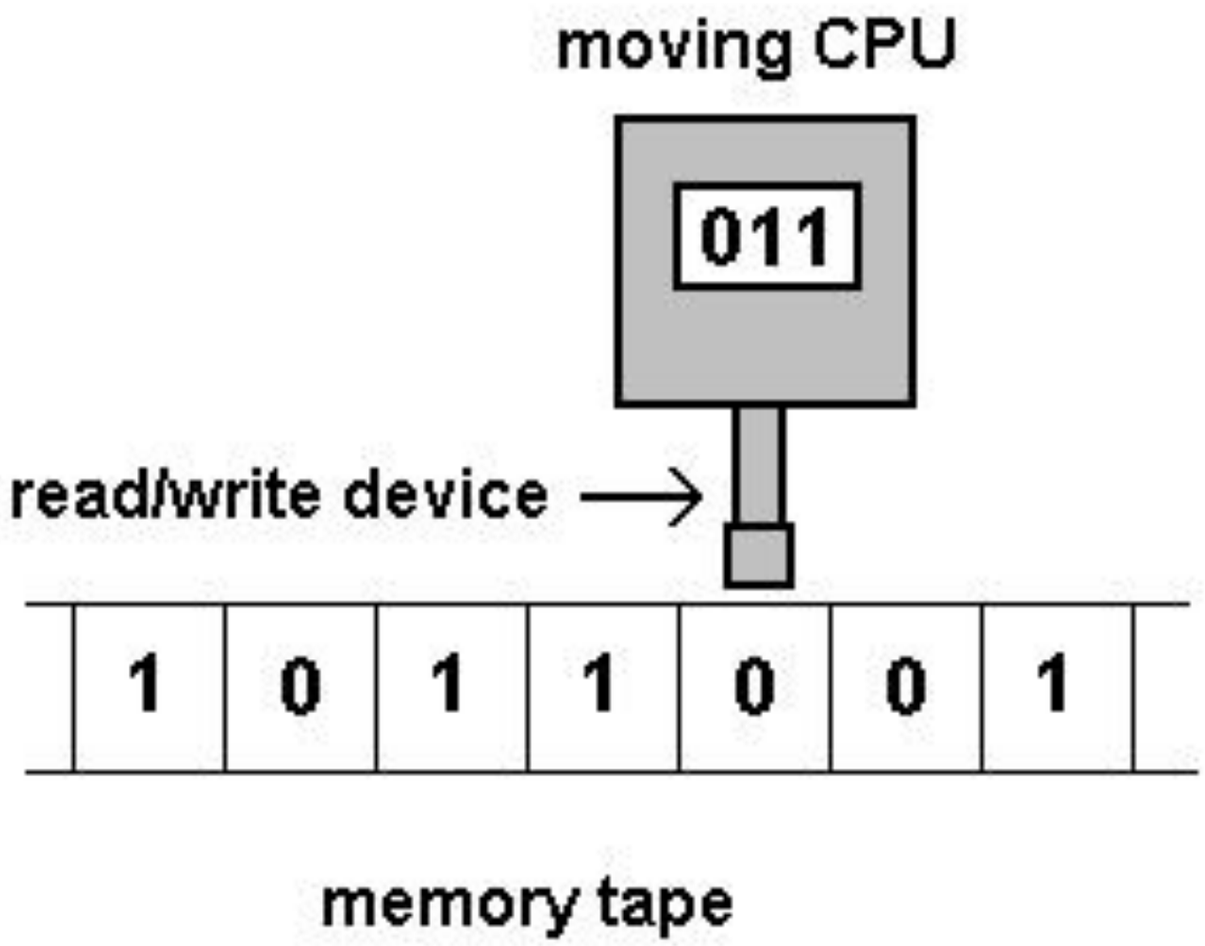


Turing machines



Turing complete system:
Can simulate any single-taped Turing machine.

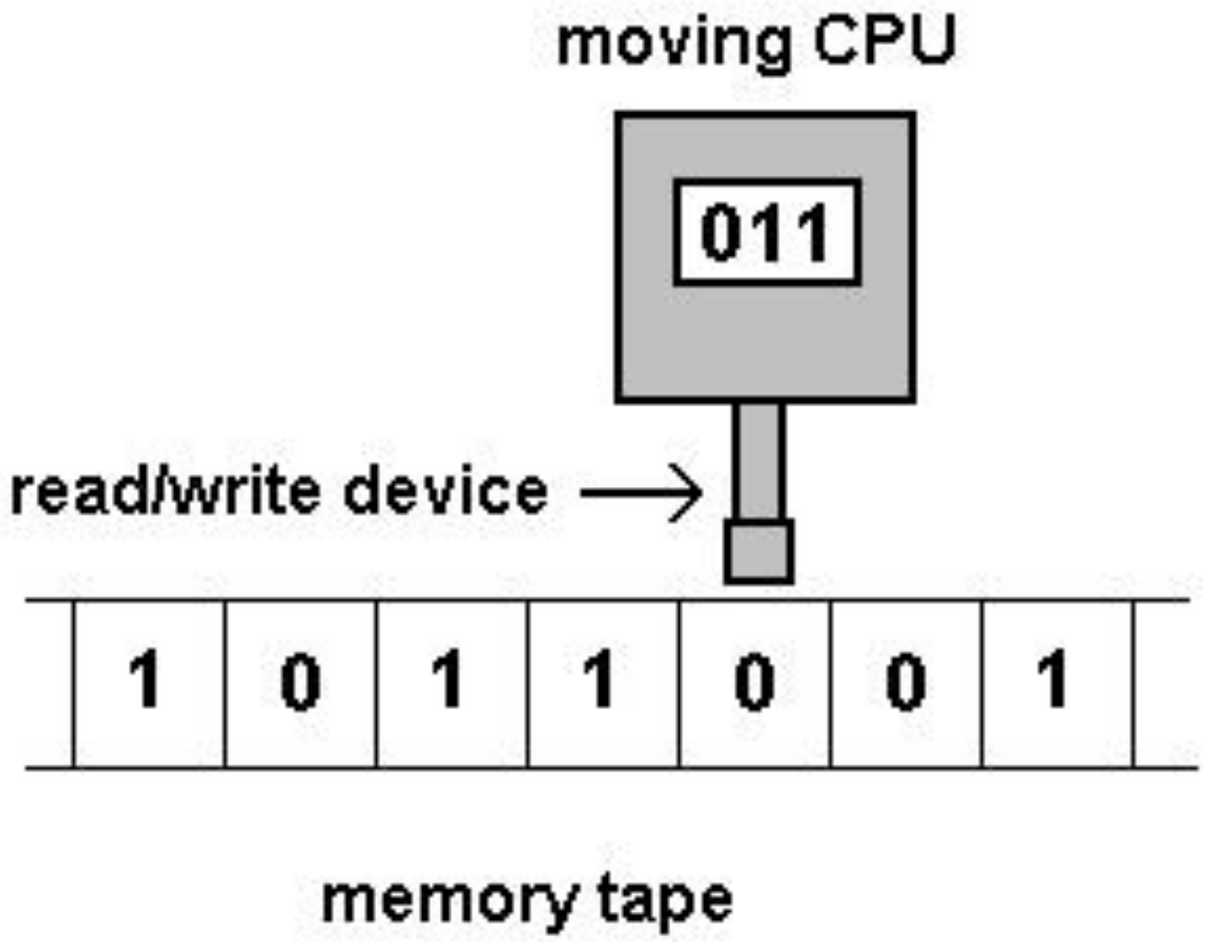
Turing machines



Turing complete system:
Can simulate any single-taped Turing machine.

Are RNNs Turing complete?

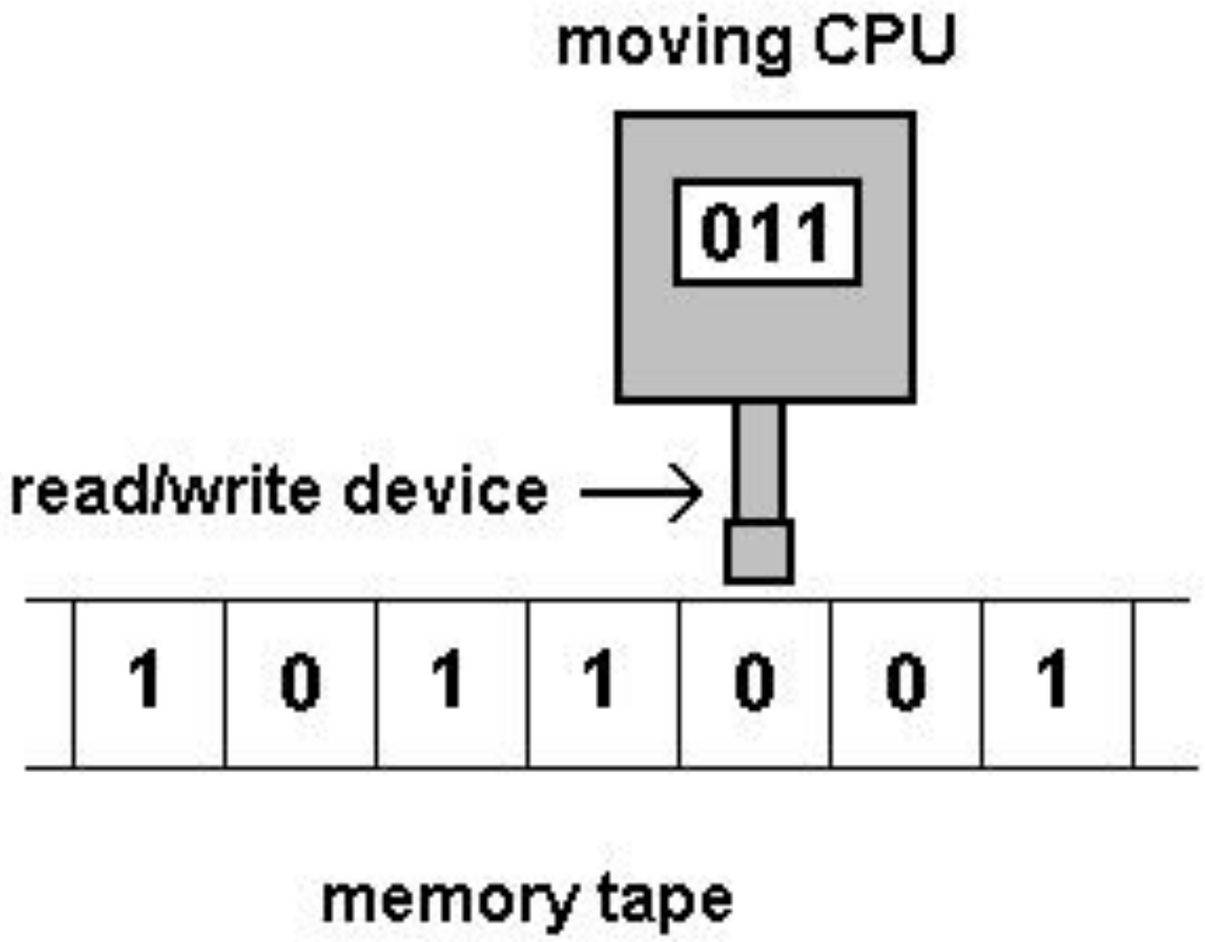
Turing machines



Turing complete system:
Can simulate any single-taped Turing machine.

Are RNNs Turing complete?
YES

Turing machines

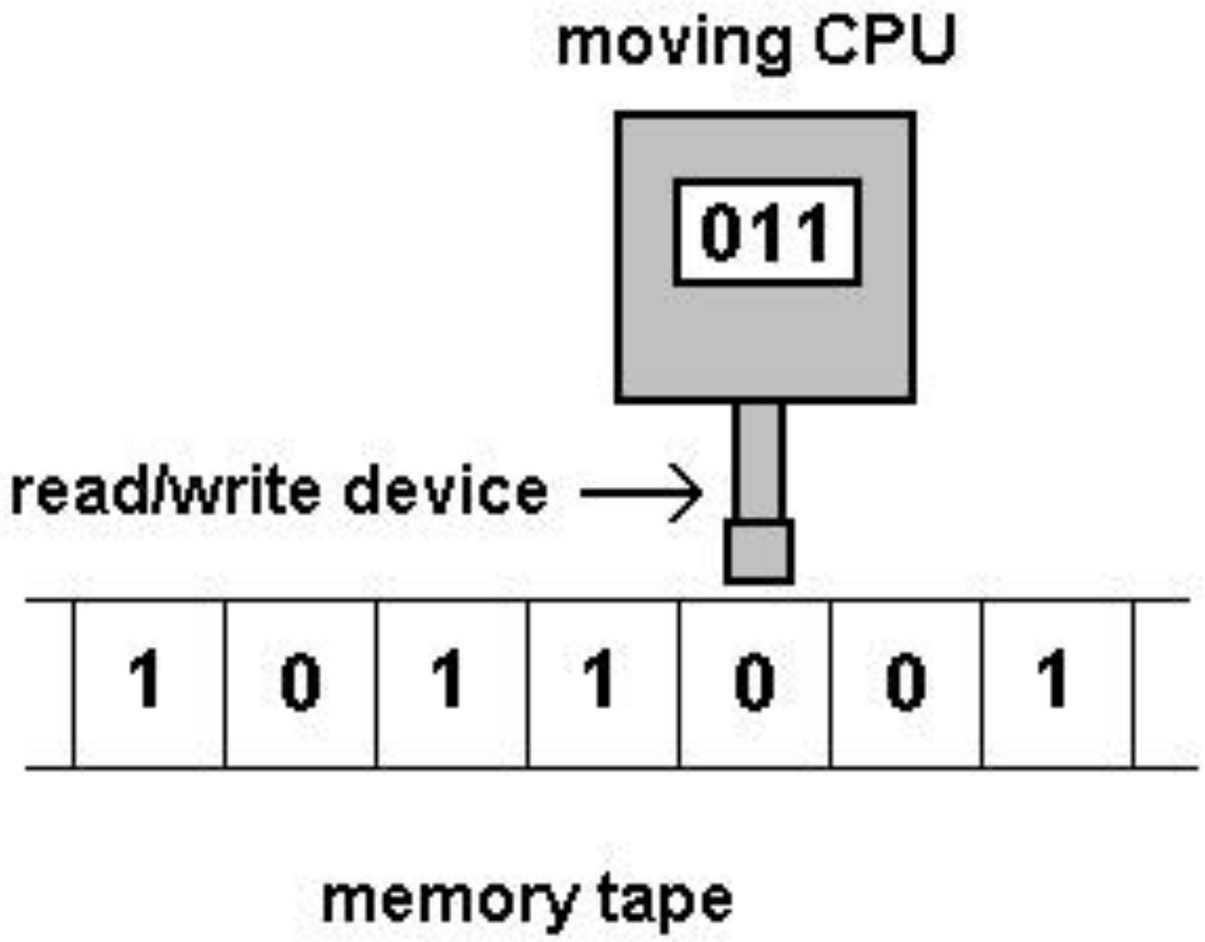


Turing complete system:
Can simulate any single-taped Turing machine.

Are RNNs Turing complete?
YES

Is it possible to train TM with backpropagation?

Turing machines

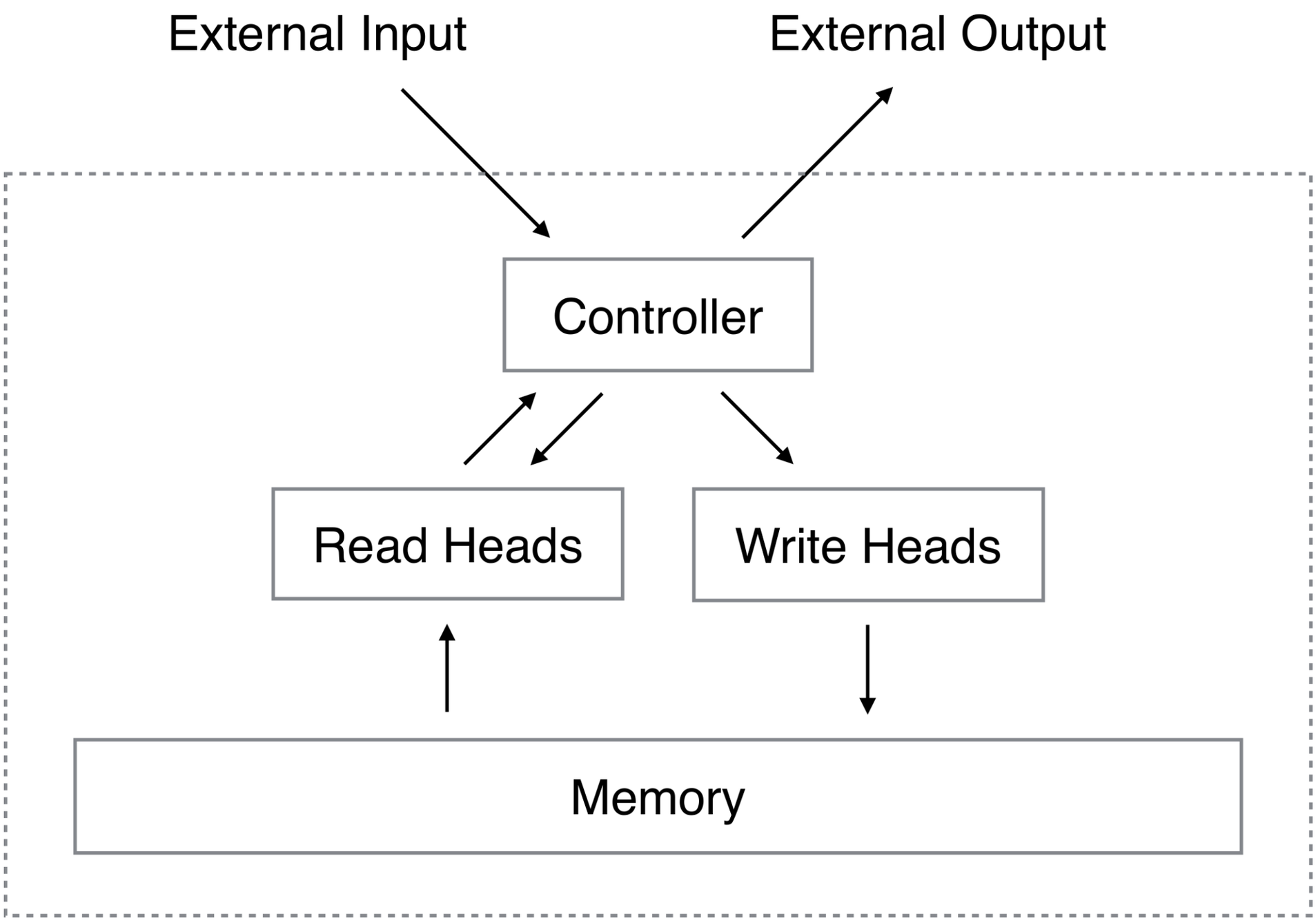


Turing complete system:
Can simulate any single-taped Turing machine.

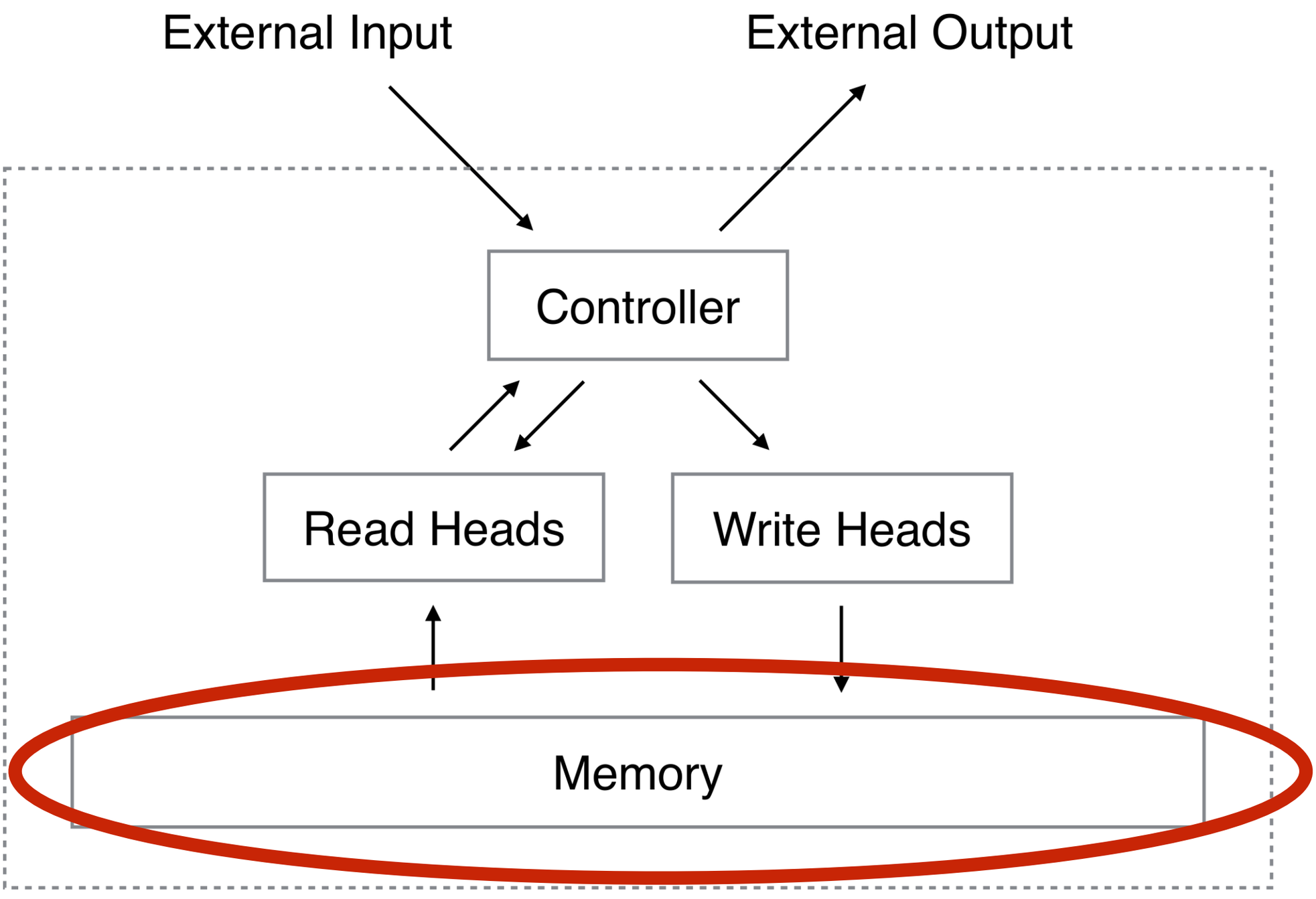
Are RNNs Turing complete?
YES

Is it possible to train TM with backpropagation?
NO

Neural Turing machines



Neural Turing machines



Memory model

- Memory is a matrix

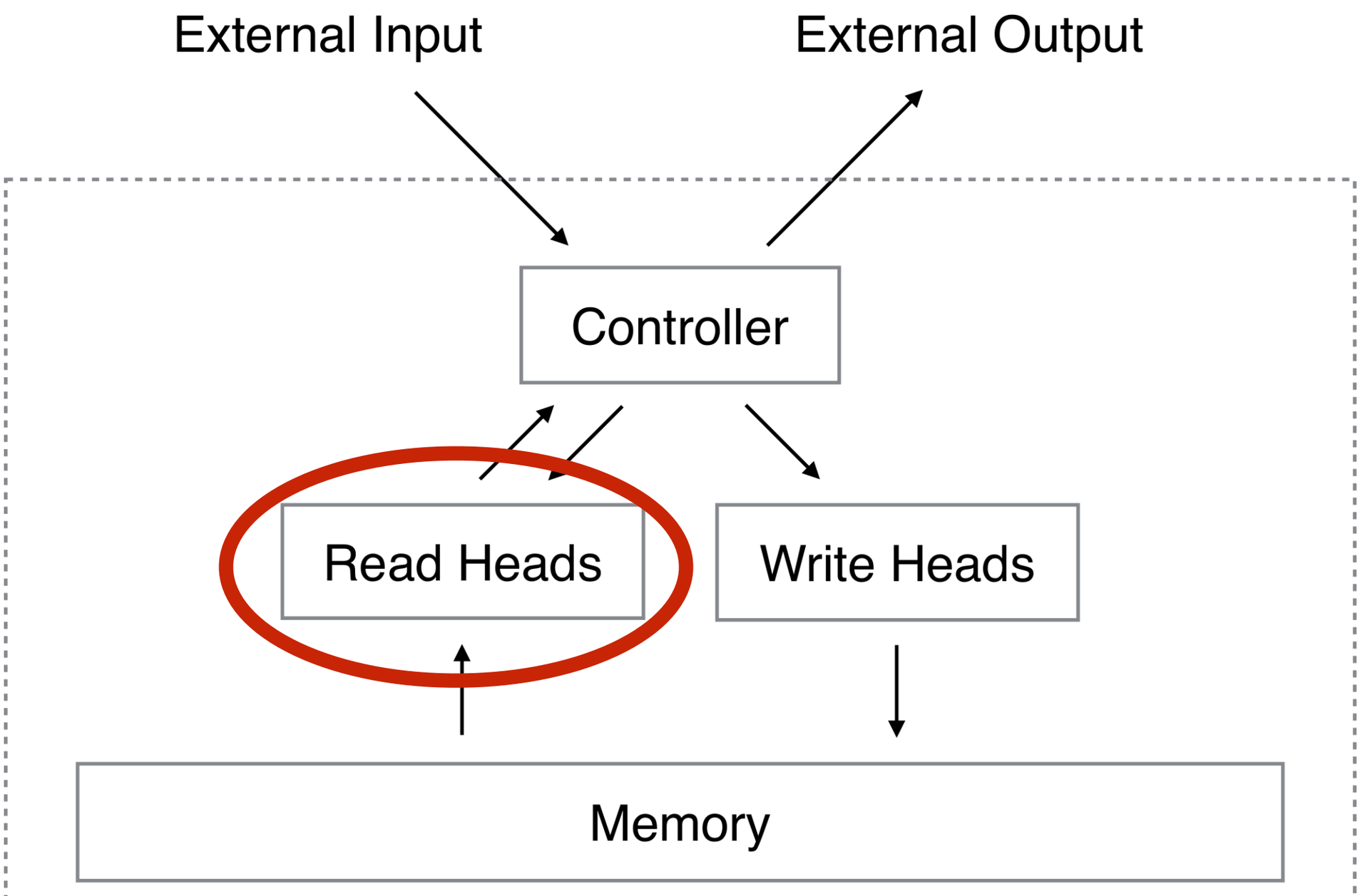
$$M \in R^{NxD}$$

- In read-only scenario its elements are trained

- Memory can learn a knowledge useful for processing

M[1]
M[2]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[N]

Neural Turing machines



Reading from ROM

- Generate weights v_i for each element and temperature β
- Normalize distribution with SoftMax $w_i = \frac{e^{\beta v i}}{\sum e^{\beta v_j}}$
- Return a weighted sum over memory elements

M[1]	0
M[2]	**0.3**
M[…]	0
M[…]	0
M[…]	0
M[…]	0
M[…]	0
M[…]	0
M[…]	0
M[15]	**0.5**
M[…]	0
M[…]	0
M[N]	**0.2**

$$r = 0.3 \cdot M[2] + 0.5 \cdot M[15] + 0.2 \cdot M[N]$$

Focusing by content

- Weights are generated with $v_i = K[k, M[i]]$ for some similarity measure K

- $K[u, v] = \frac{u \cdot v}{\|u\| \cdot \|v\|}$

- This recalls associative arrays

M[1]
M[2]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[N]

Bad version

Focusing by index

- Set first number in array to the index
- Use $K[u, v] = \|u_0\text{-}v_0\|$

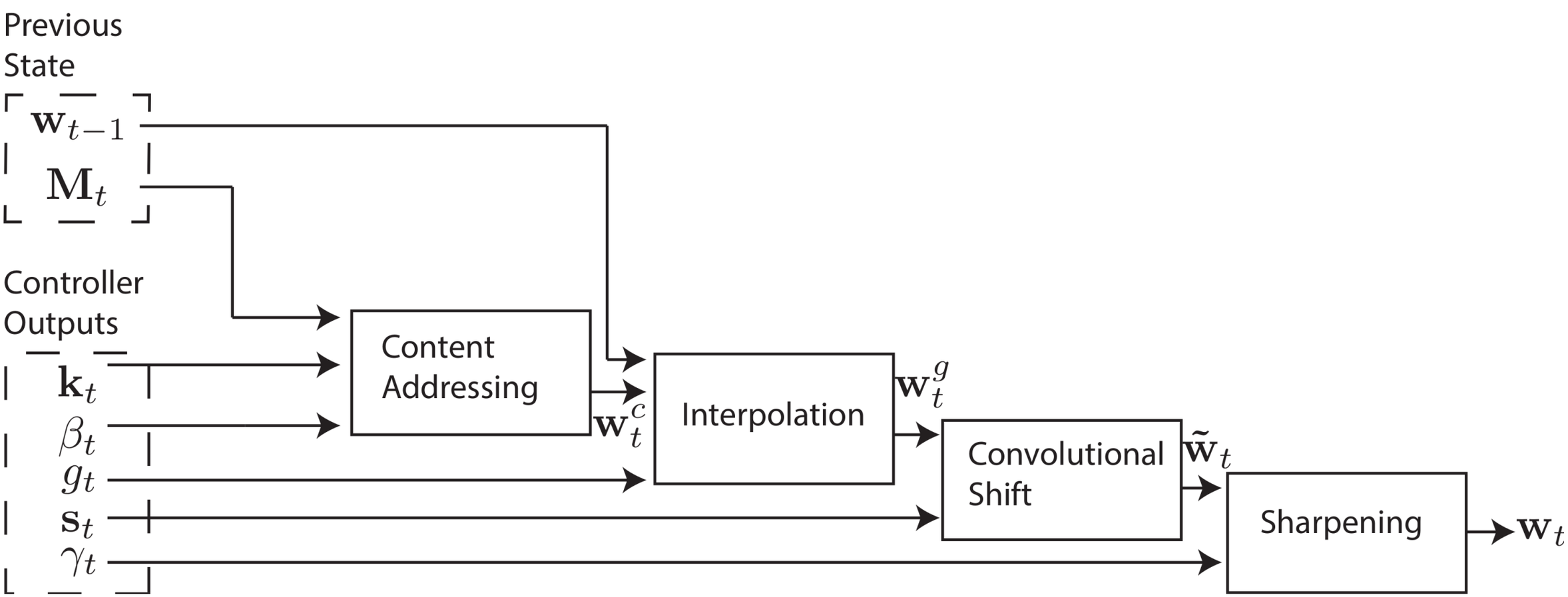
1	M[1]
2	M[2]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
…	M[…]
N	M[N]

Focusing by index

- Use previous w
- Emit another distribution s
- $w_{new} = w * s$
- $s = [0, 0, 1]$ will shift attention right (like i++)

M[1]
M[2]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[N]

Indexing: combining

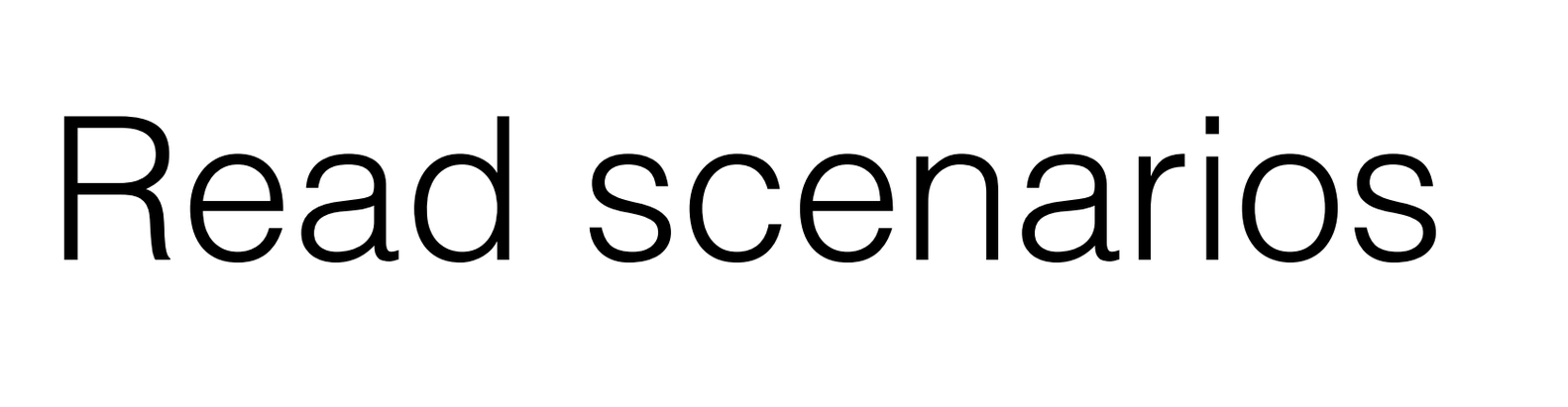


Read scenarios

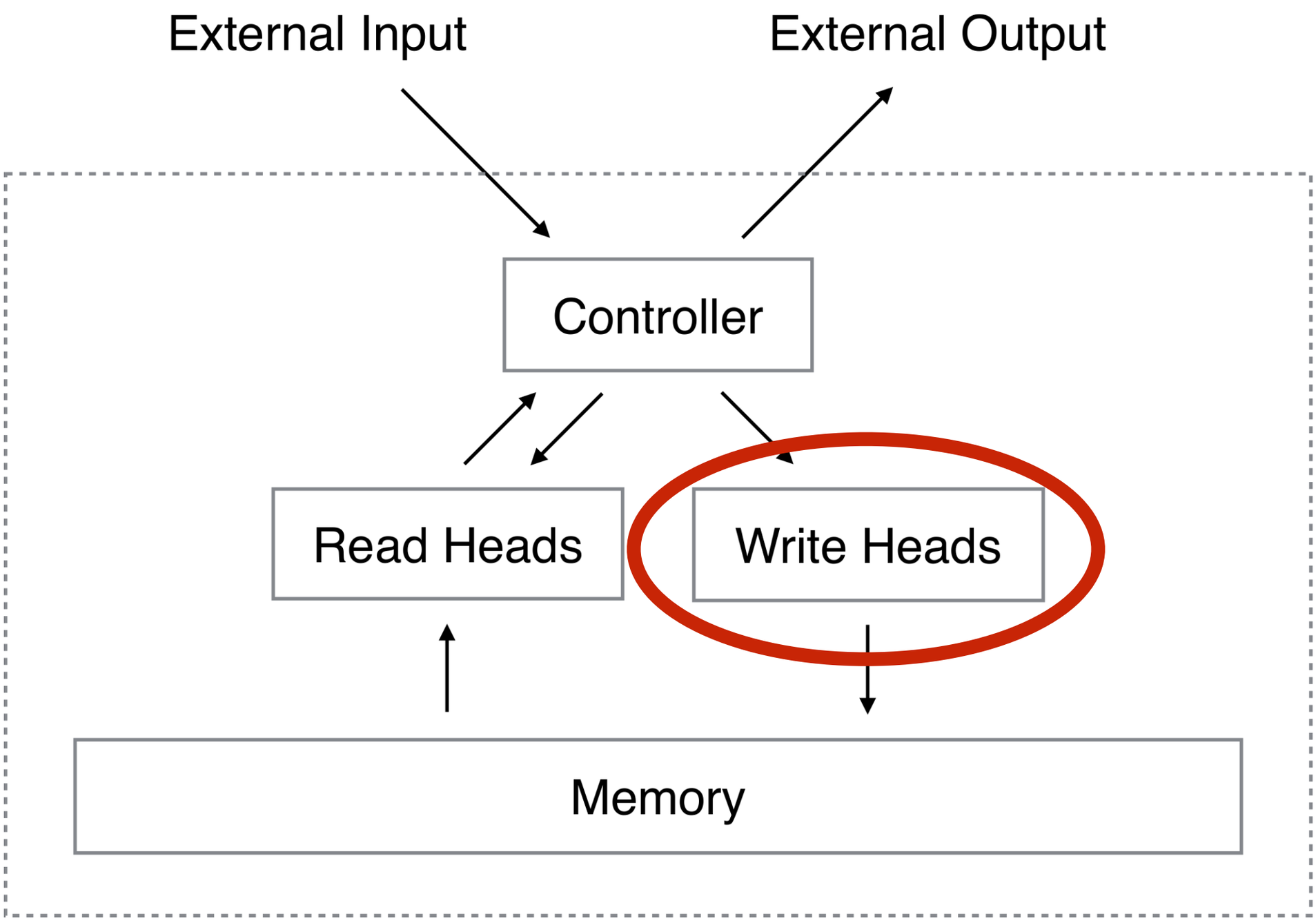
- $g_t = 0, s = [0, 1, 0]$ —> content retrieval
- $g_t = 1, s = [0, 0, 1]$ —> iteration

Read scenarios

- $g_t = 0, s = [0, 1, 0]$ —> content retrieval
- $g_t = 1, s = [0, 0, 1]$ —> iteration
- Combination:



Neural Turing machines



Writing

$$M_t[i] = M_{t\text{-}1}[i] \cdot (1 - w_t[i]e_t) + w_t[i]a_t$$

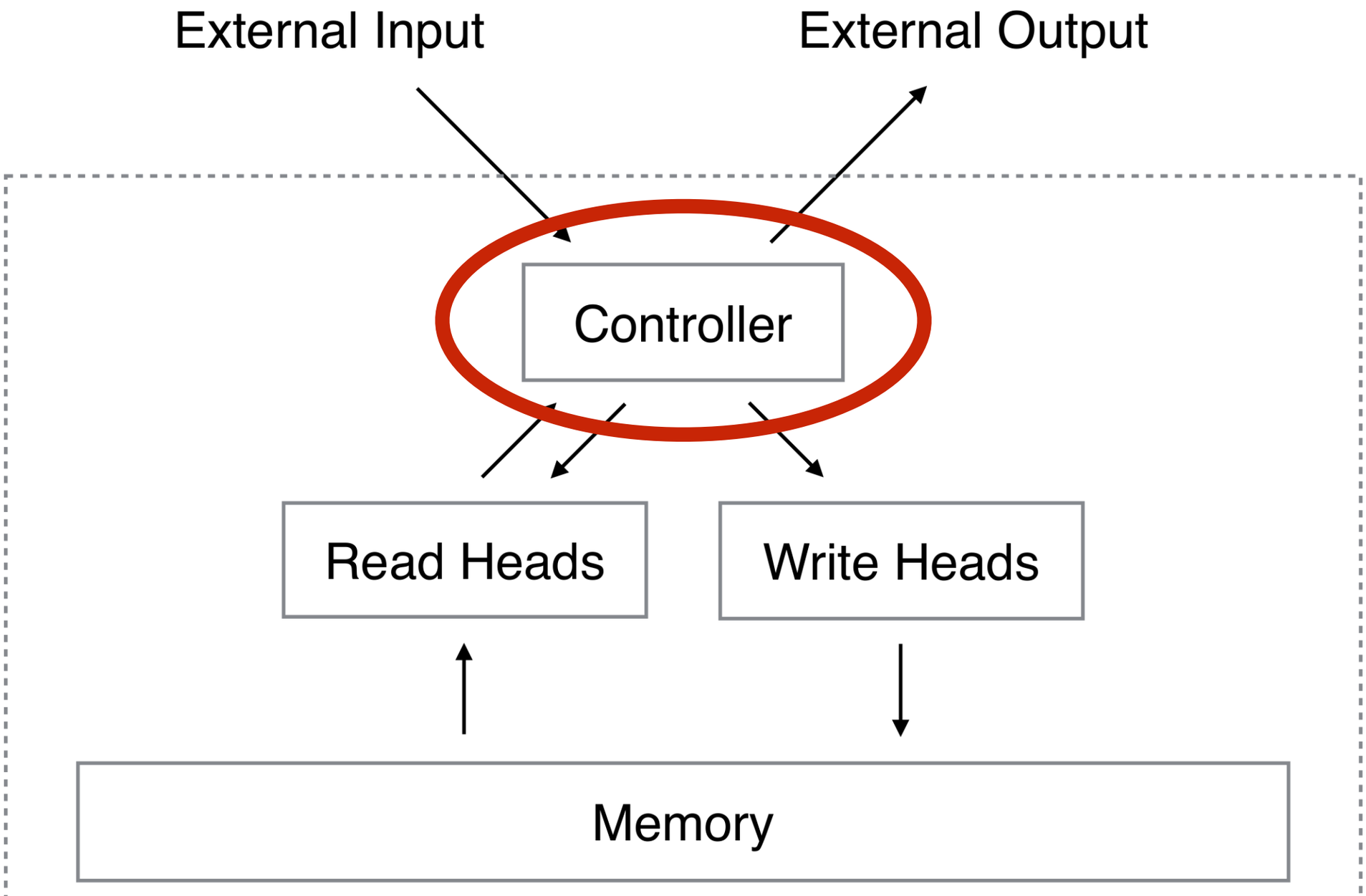
Erase

Add

- There may be multiple read/ write heads
- In this case erasure must be performed over all heads first, followed by addition

M[1]
M[2]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[…]
M[N]

Neural Turing machines



Controller

Recurrent (LSTM)

Controller: CPU

LSTM cell: Registers

Memory bank: RAM

Feed-forward

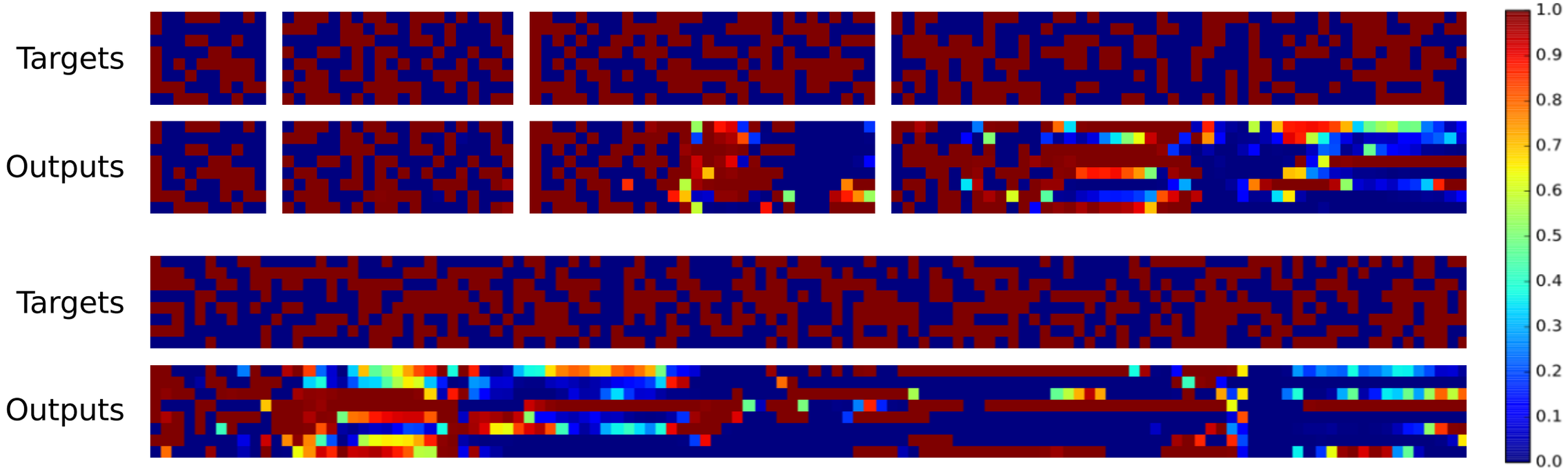
- Can analyze read/write operations
- Can mimic RNN with a lot of layers (but network will be very deep)

Note: train model can operate with arbitrary memory size

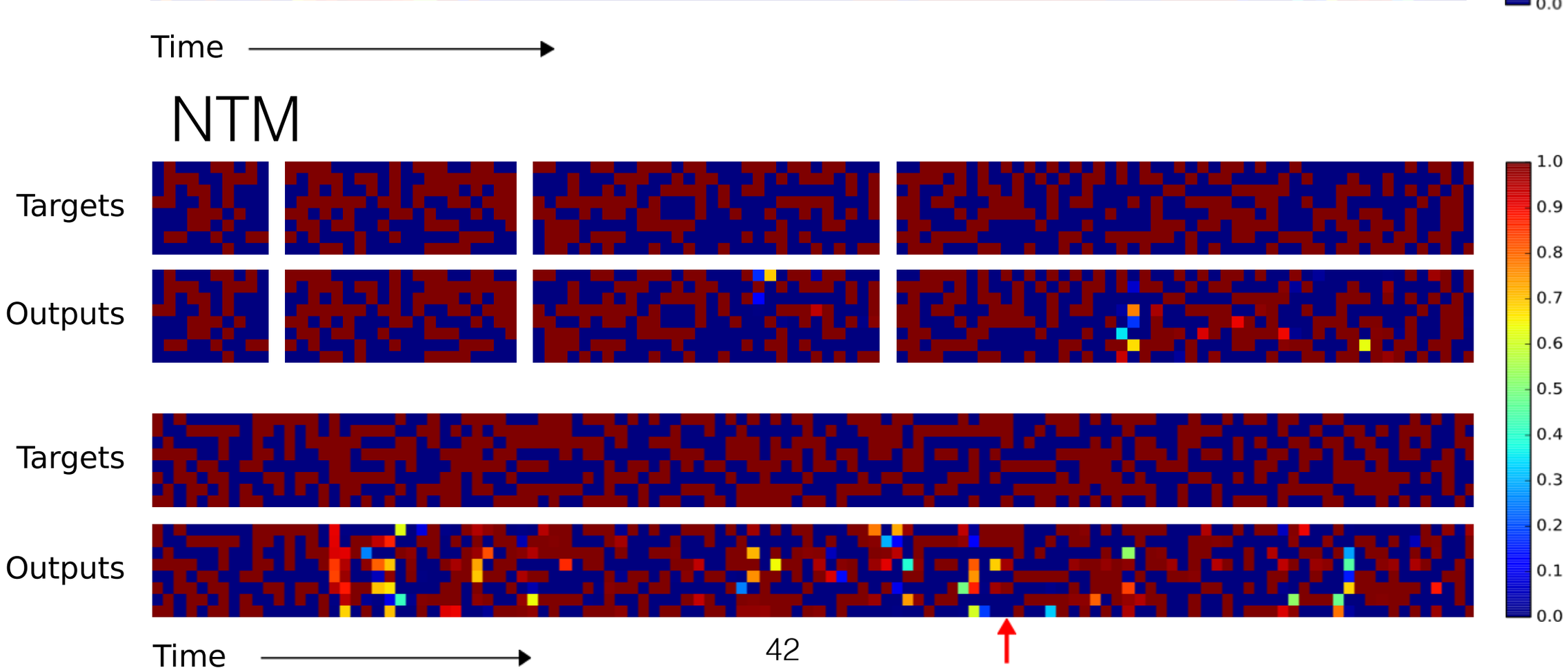
Experiments on NTM

Copy task

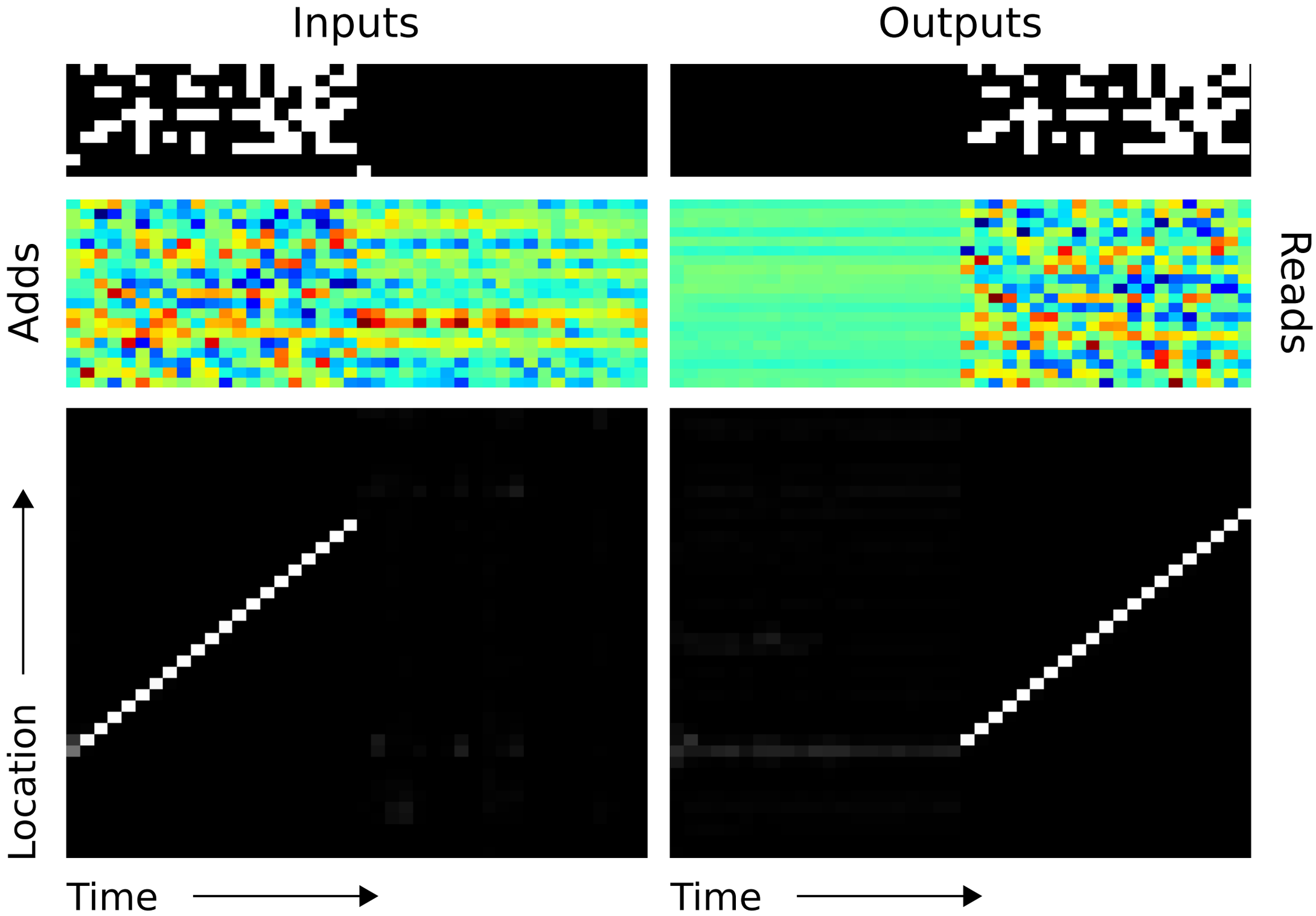
LSTM



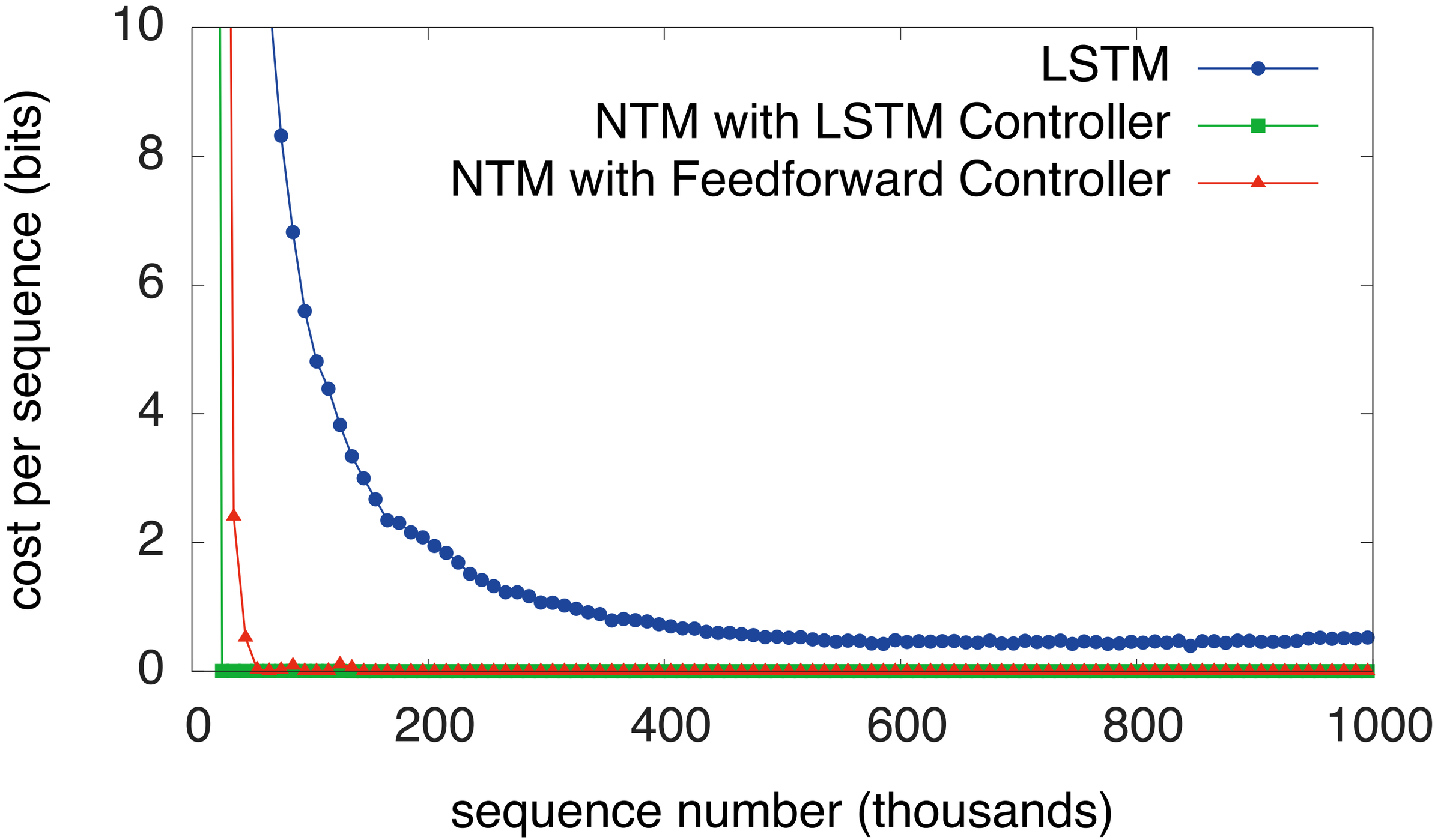
NTM



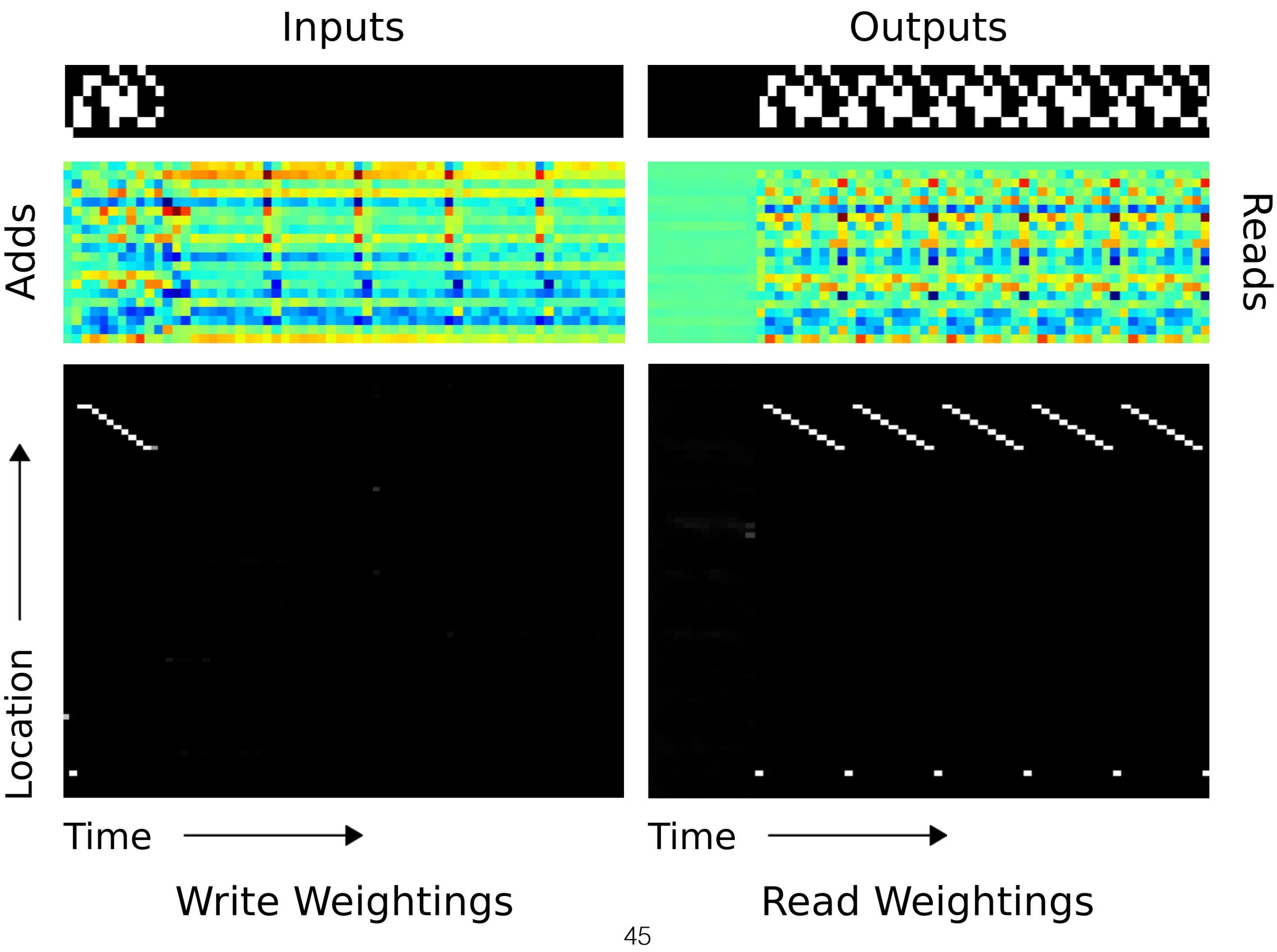
Copy task


Inputs
Outputs
Adds
Reads
Location
Time
Time
Write Weightings
Read Weightings

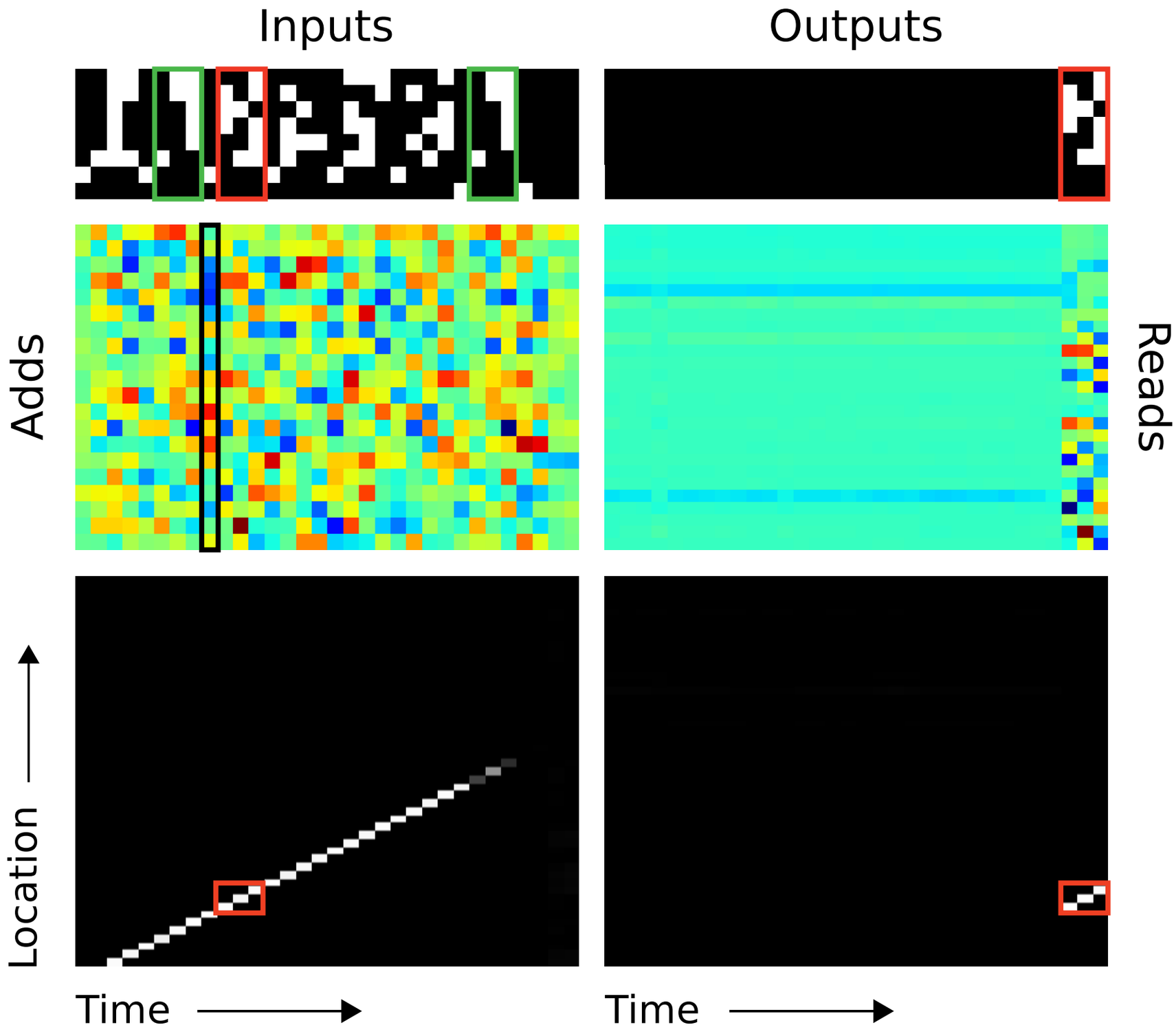
Copy task


cost per sequence (bits)
10
8
6
4
2
0
0
200
400
600
800
1000
sequence number (thousands)
LSTM
NTM with LSTM Controller
NTM with Feedforward Controller

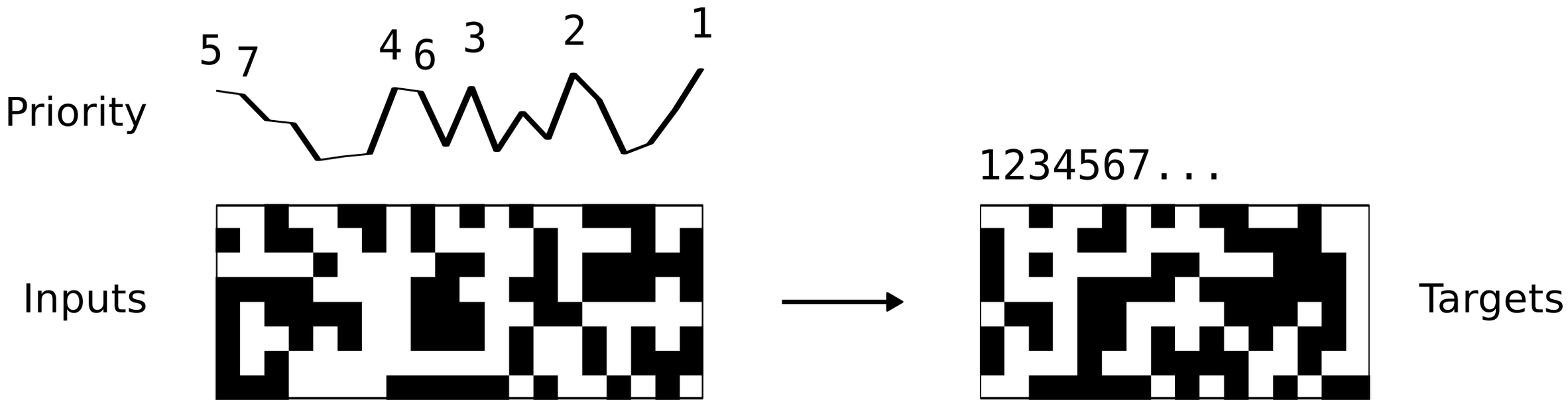
Repeat copy

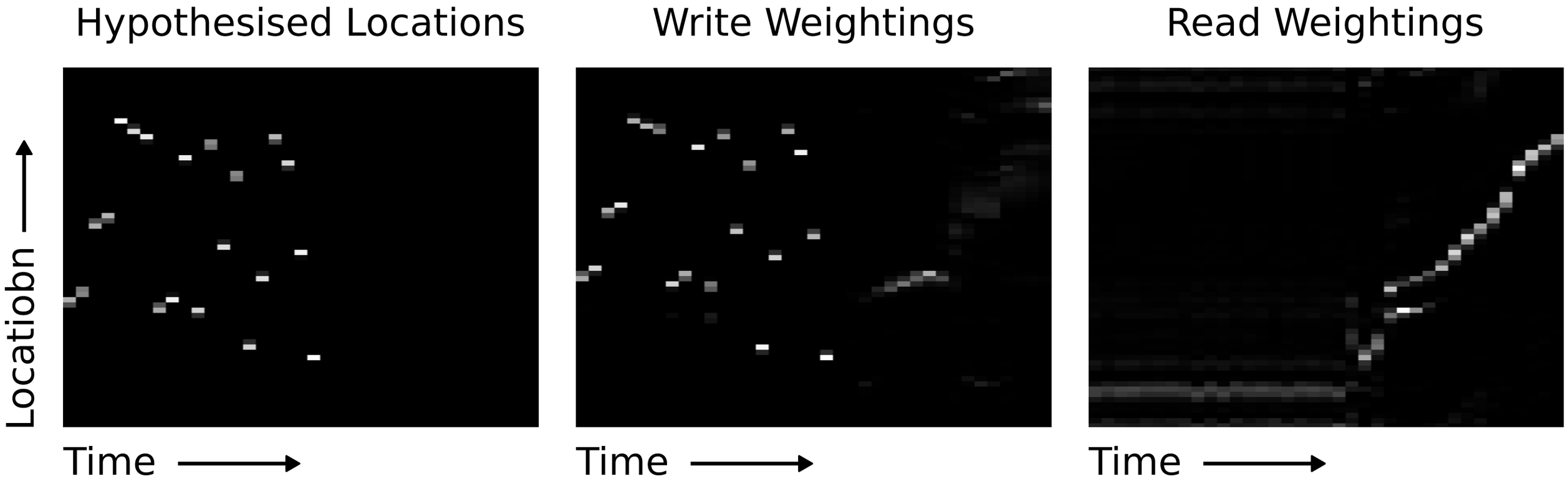

Inputs
Outputs
Adds
Reads
Location
Time
Time
Write Weightings
Read Weightings

Associative recall

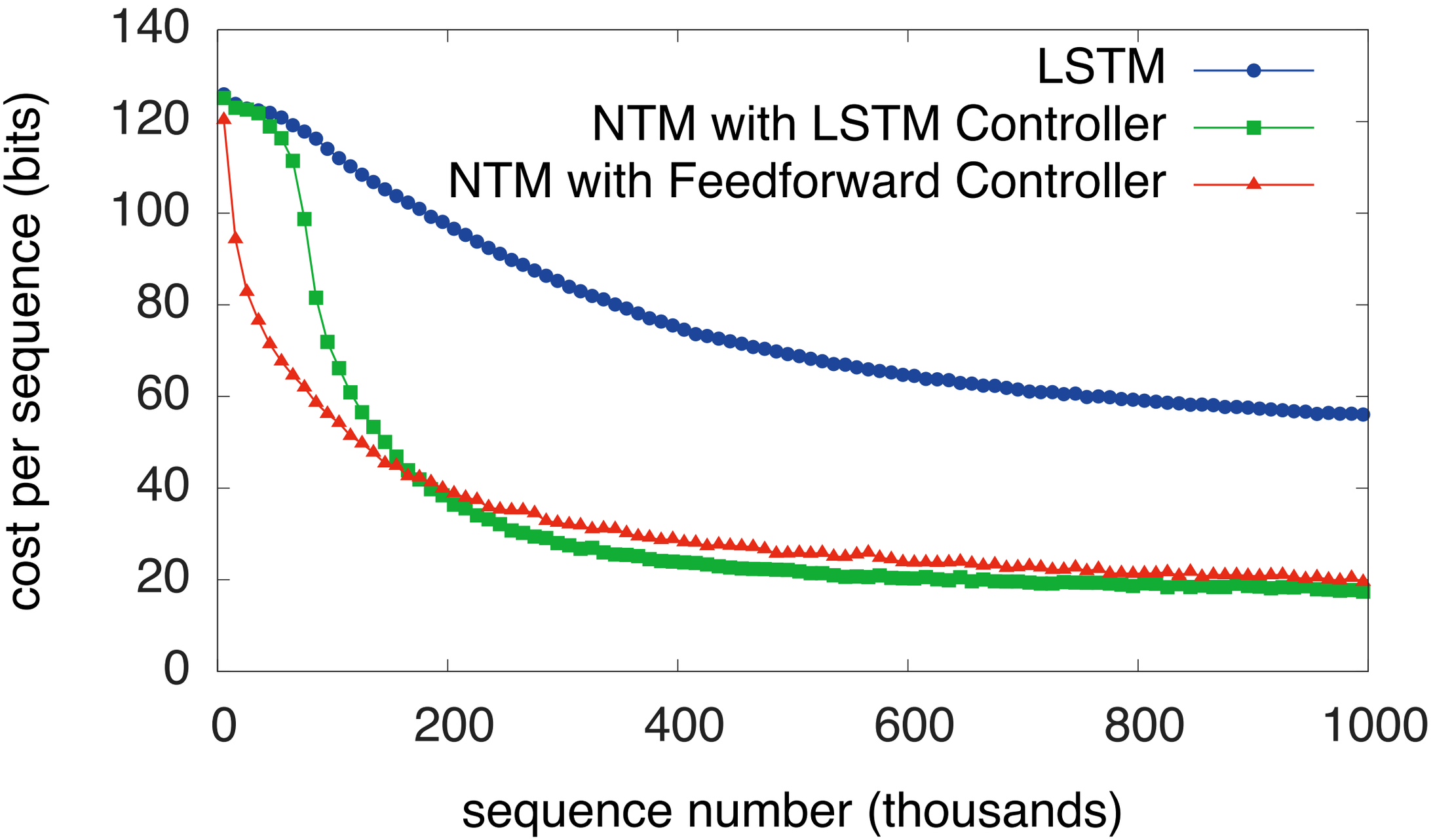


Sorting

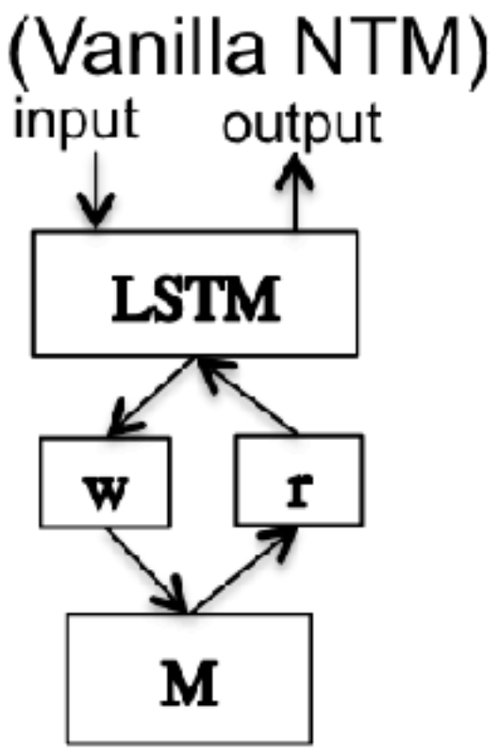

5 7
4 6
3
2
1
Priority
1234567...
Inputs
Targets


Hypothesised Locations
Write Weightings
Read Weightings
Locatiobn
Time
Time
Time

Sorting

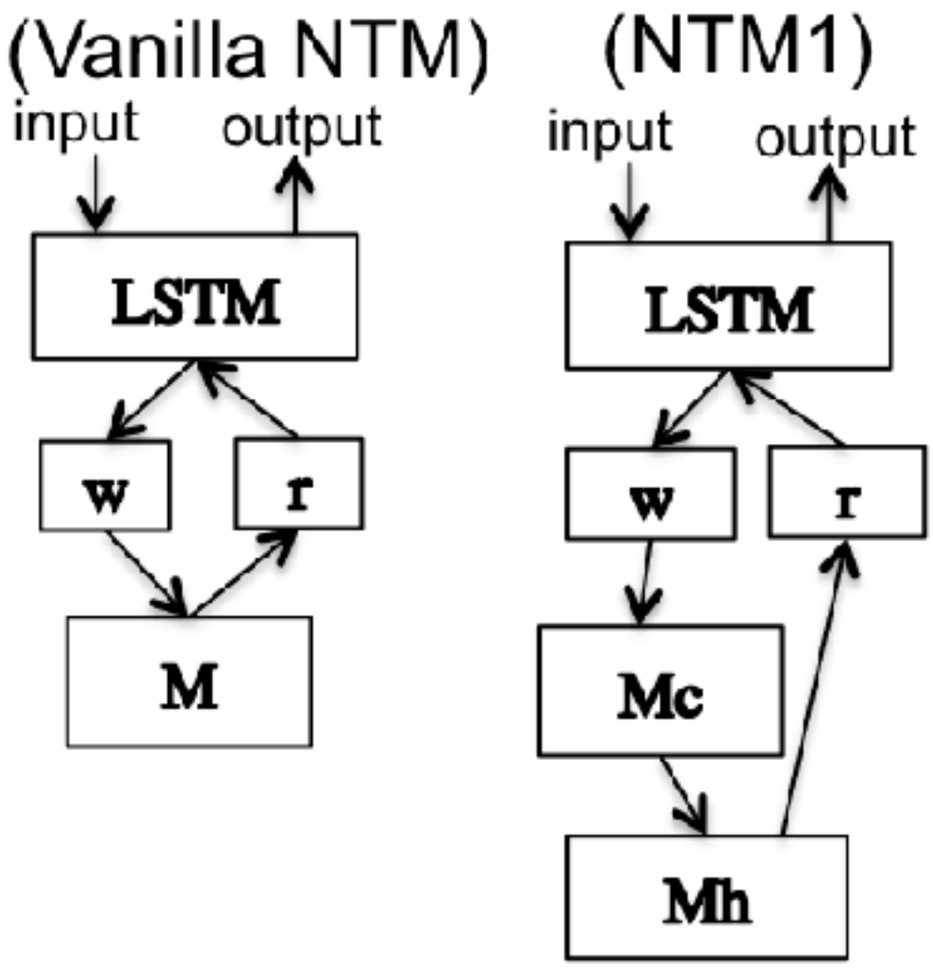

LSTM
NTM with LSTM Controller
NTM with Feedforward Controller
cost per sequence (bits)
0
20
40
60
80
100
120
140
0
200
400
600
800
1000
sequence number (thousands)

Structured memory NTM



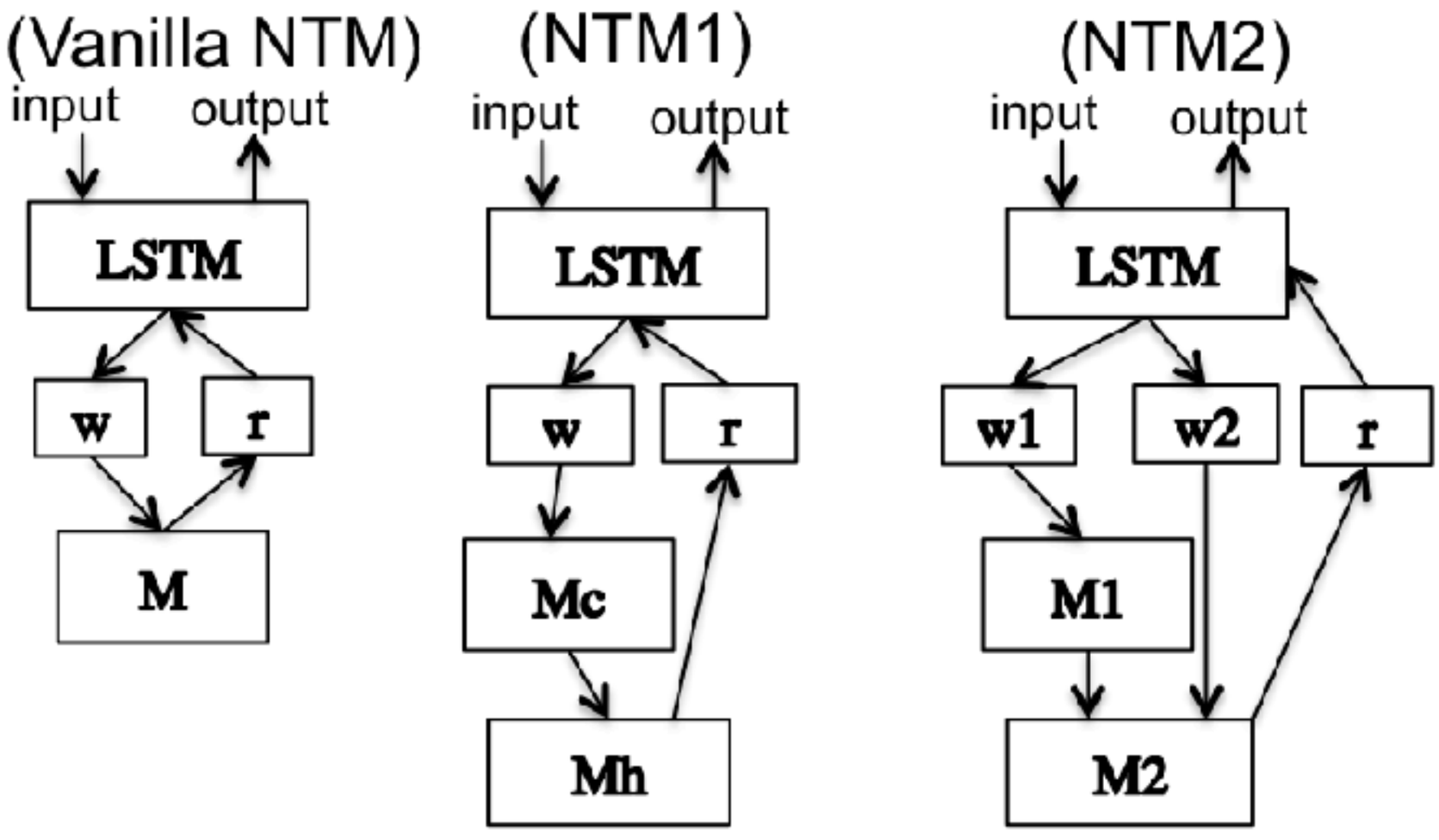
Zhang et.al, Structured Memory for Neural Turing Machines

Structured memory NTM



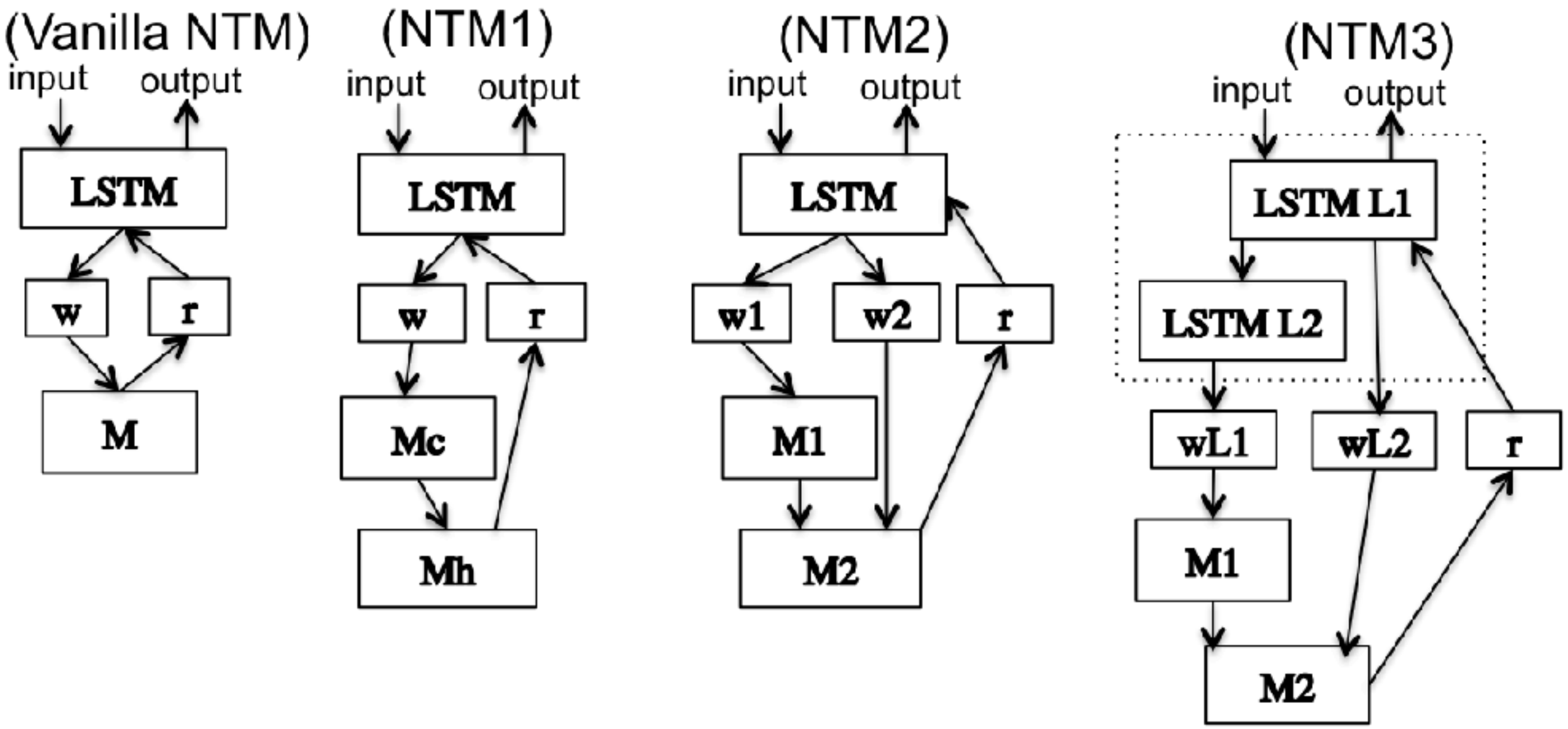
Zhang et.al, Structured Memory for Neural Turing Machines

Structured memory NTM



Zhang et.al, Structured Memory for Neural Turing Machines

Structured memory NTM



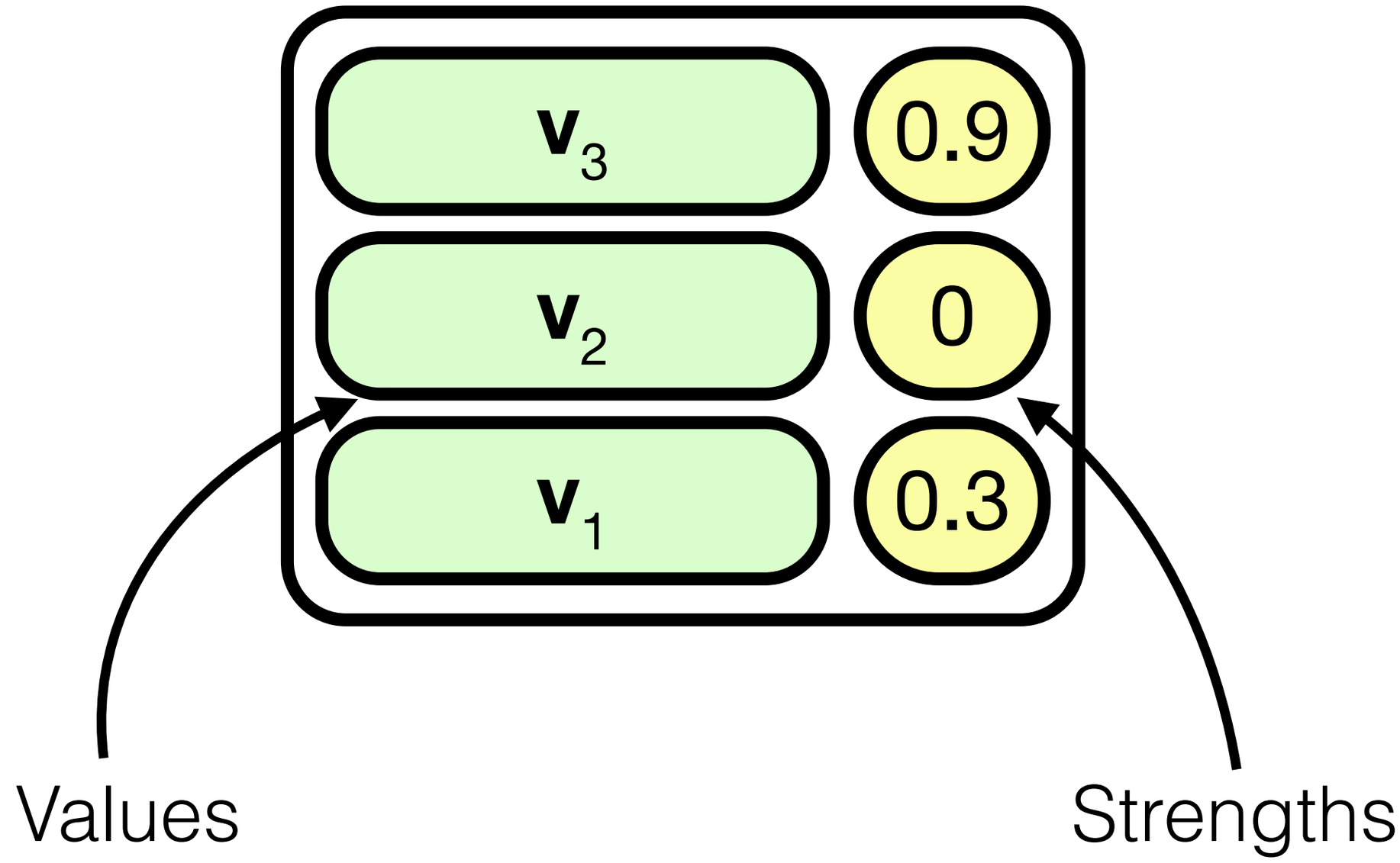
Zhang et.al, Structured Memory for Neural Turing Machines

Neural Data Structures

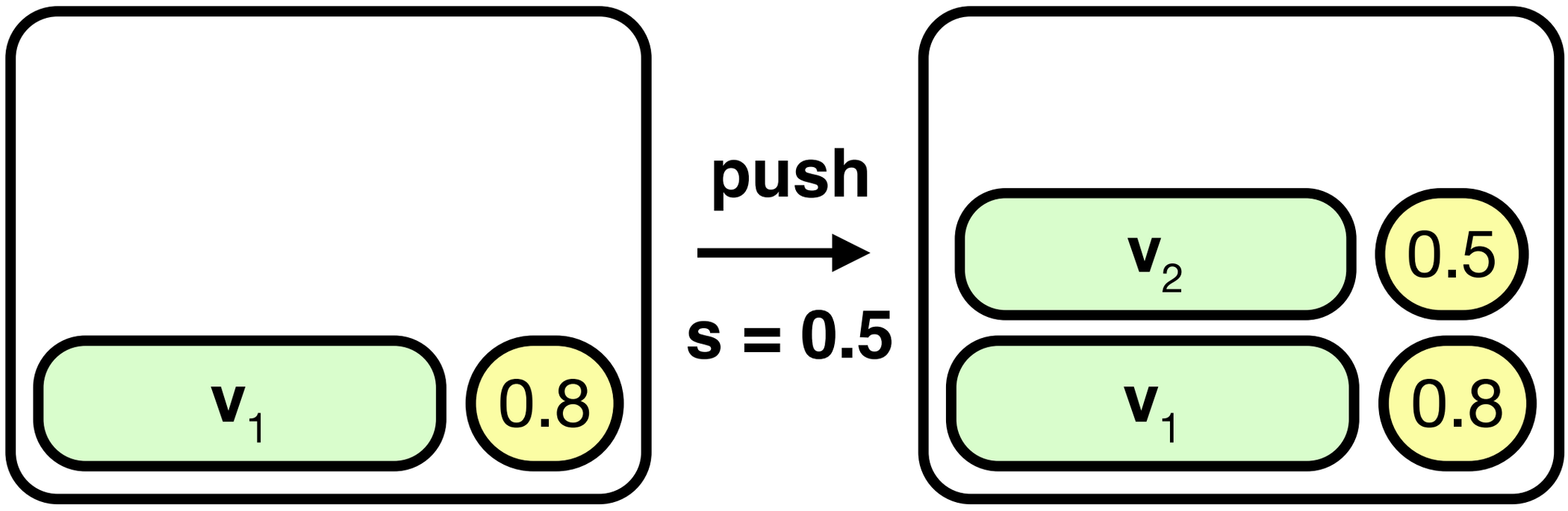
Turing vs Data structures

- Turing machines are hard to program
- Neural Turing machine does not allow dynamic memory growth
- Data structures are often useful: stacks, lists, queues, etc.

Stack

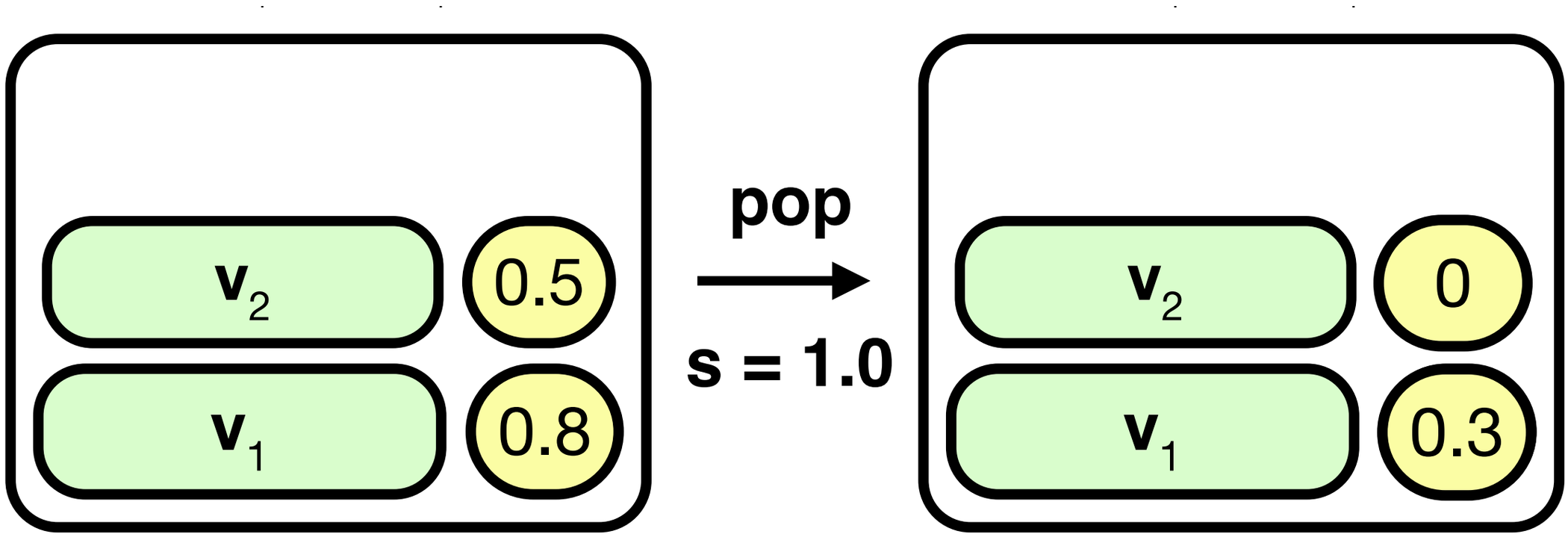


Push



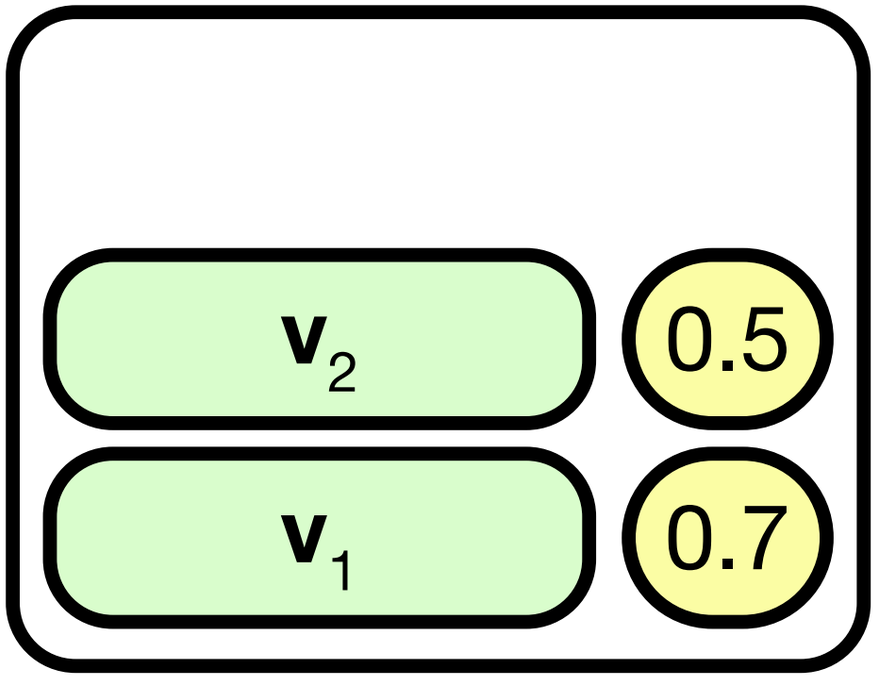
- Push **v** with strength **s**: put a vector on top of the stack, assign value **s** to it.

Pop



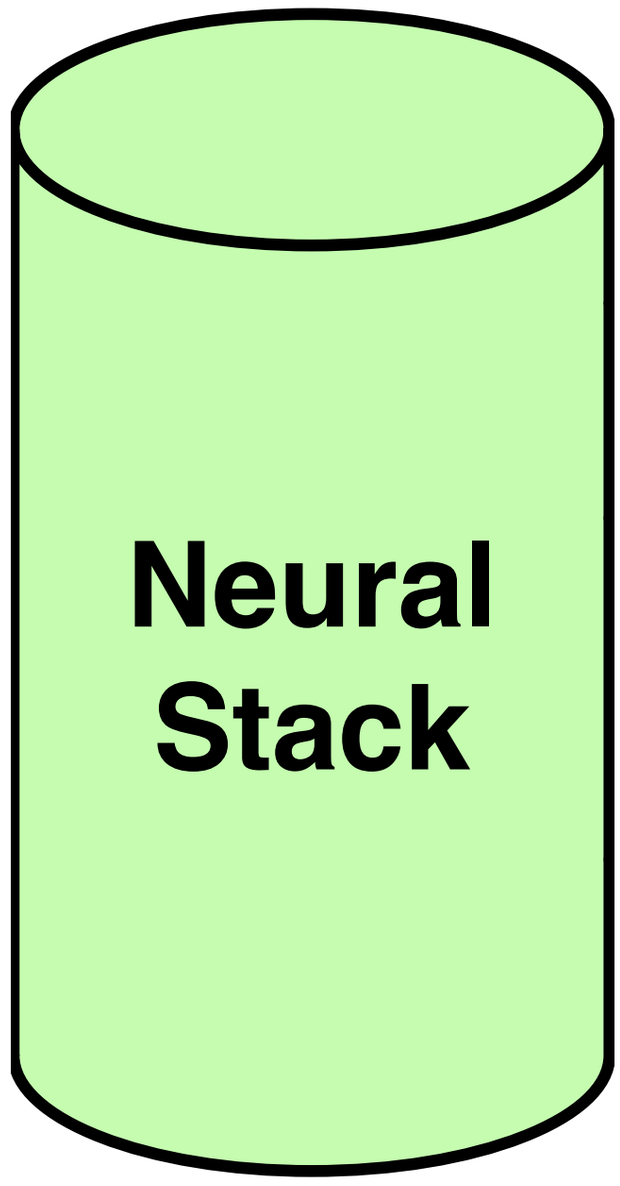
- Pop with strength **s**: subtract total of **s** from top vectors preserving non-negative presence.

Head

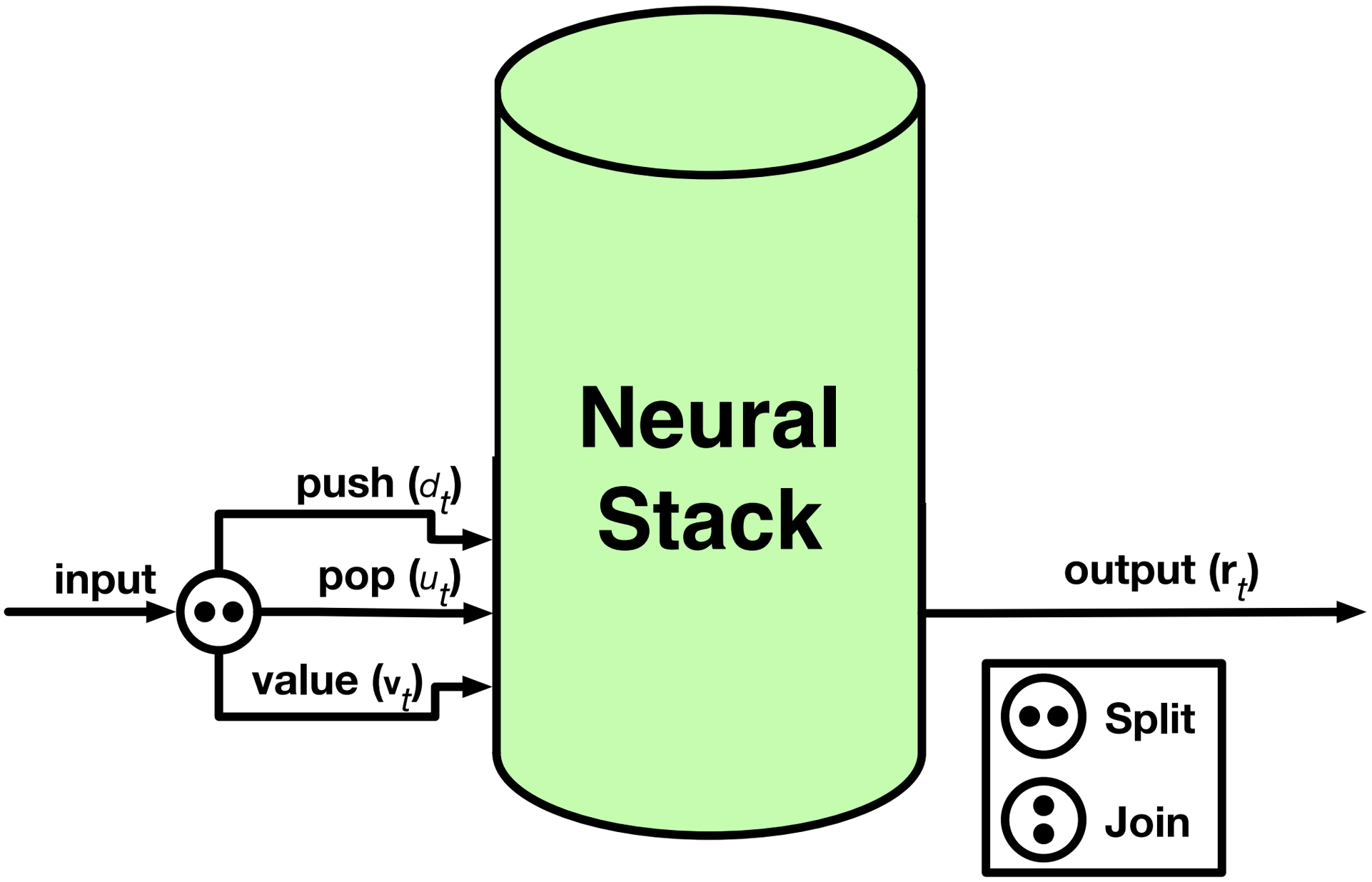


- Head: compute convex sum of top vectors until sum of 1 is reached.

Stack



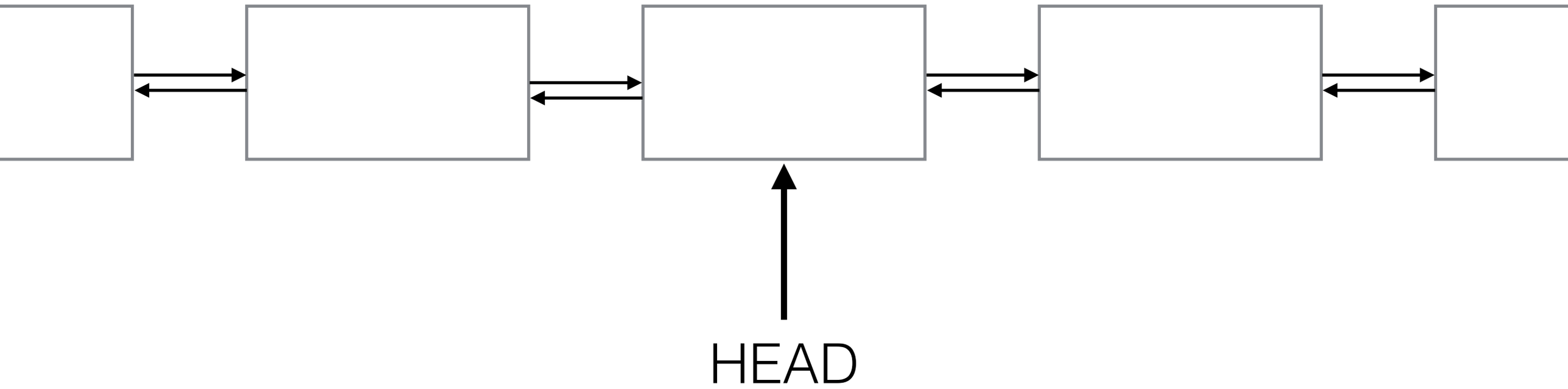
Stack


Neural Stack
push (d_t)
pop (u_t)
value (v_t)
input
output (r_t)
Split
Join

Neural data structures

- Queues and DeQues are constructed similarly
- During training network tends to ignore stack
- This is fixed with bias towards pushing

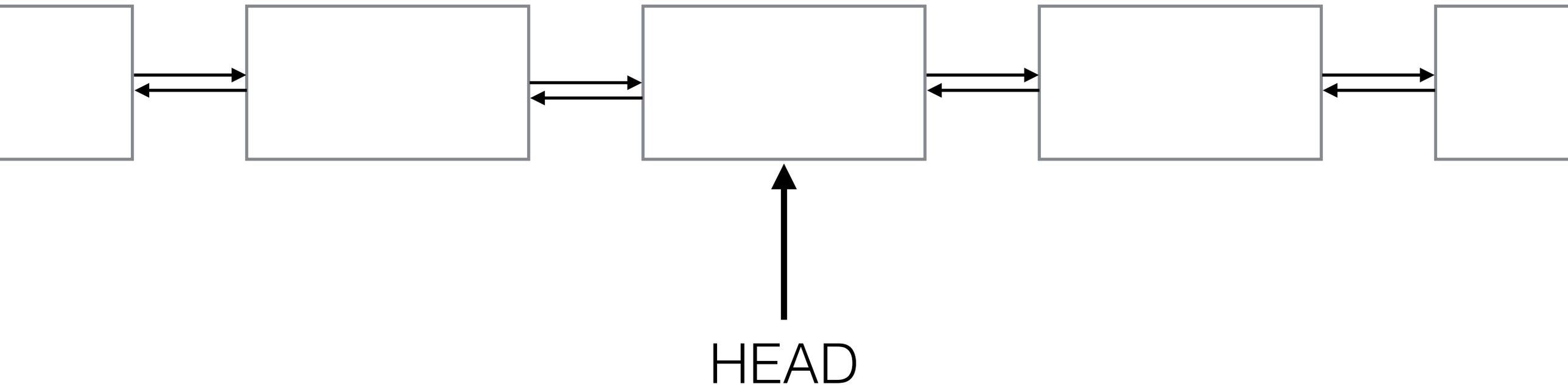
Double-linked lists



Operations:

- Move HEAD left:
- Move HEAD right:
- Insert new element at position HEAD:

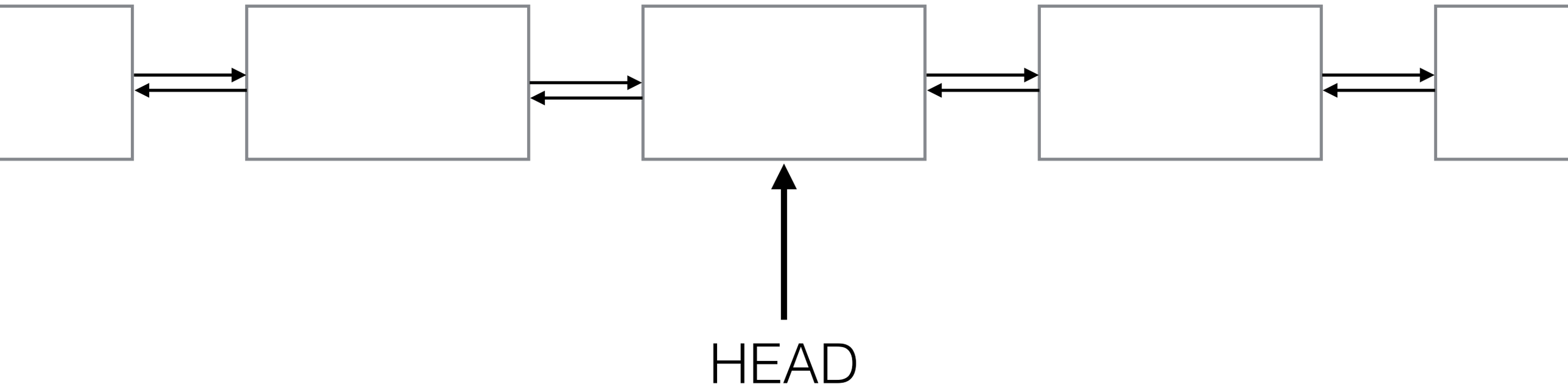
Double-linked lists



Operations:

- Move HEAD left: $L_{t+1}[i] = L_t[i\text{-}1]$
- Move HEAD right:
- Insert new element at position HEAD:

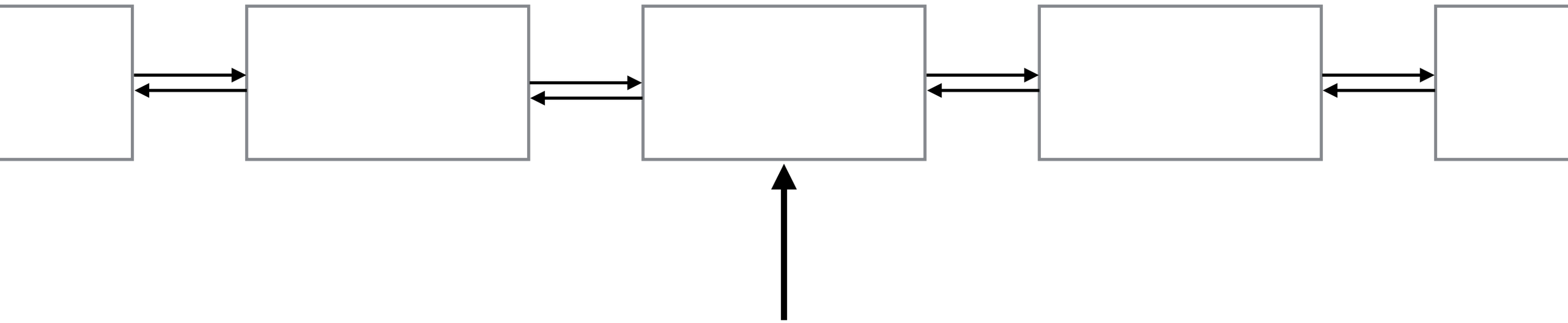
Double-linked lists



Operations:

- Move HEAD left: $L_{t+1}[i] = L_t[i\text{-}1]$
- Move HEAD right: $L_{t+1}[i] = L_t[i\text{+}1]$
- Insert new element at position HEAD:

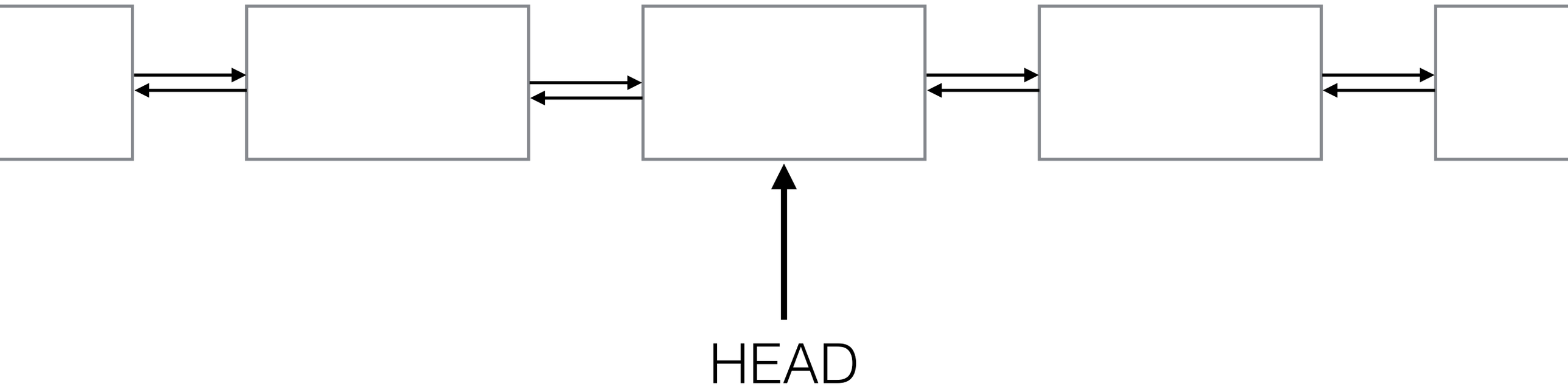
Double-linked lists



Operations:

- Move HEAD left: $L_{t+1}[i] = L_t[i\text{-}1]$
- Move HEAD right: $L_{t+1}[i] = L_t[i{+}1]$
- Insert new element at position HEAD: $L_{t+1}[i] = \begin{cases} L_t[i{+}1] & i < HEAD \\ L_t[i] & i > HEAD \\ v & i = HEAD \end{cases}$

Double-linked lists



Differentiable update:

$$L_{t+1}[i] = a_t \cdot L_t[i\text{-}1] + b_t \cdot L_t[i\text{+}1] + c_t \cdot \begin{cases} L_t[i\text{+}1] & i < HEAD \\ L_t[i] & i > HEAD \\ v & i = HEAD \end{cases}$$

Differentiable Neural Computer

Differentiable Neural Computer

- John Locke: *memories are connected if they were formed nearby in time and space.*

Differentiable Neural Computer

- John Locke: *memories are connected if they were formed nearby in time and space.*

- So memory can naturally be organized as a graph

Gives et.al, Hybrid computing using a neural network with dynamic external memory

DNC: Memory graph

- Add an adjacency matrix $L \in R^{NxN}$
- $L[i, j]$ is close to 1 if location i was written right after location j
- Move focus to the next memory moment: Lw
- Move focus to the previous memory moment: $L^T w$

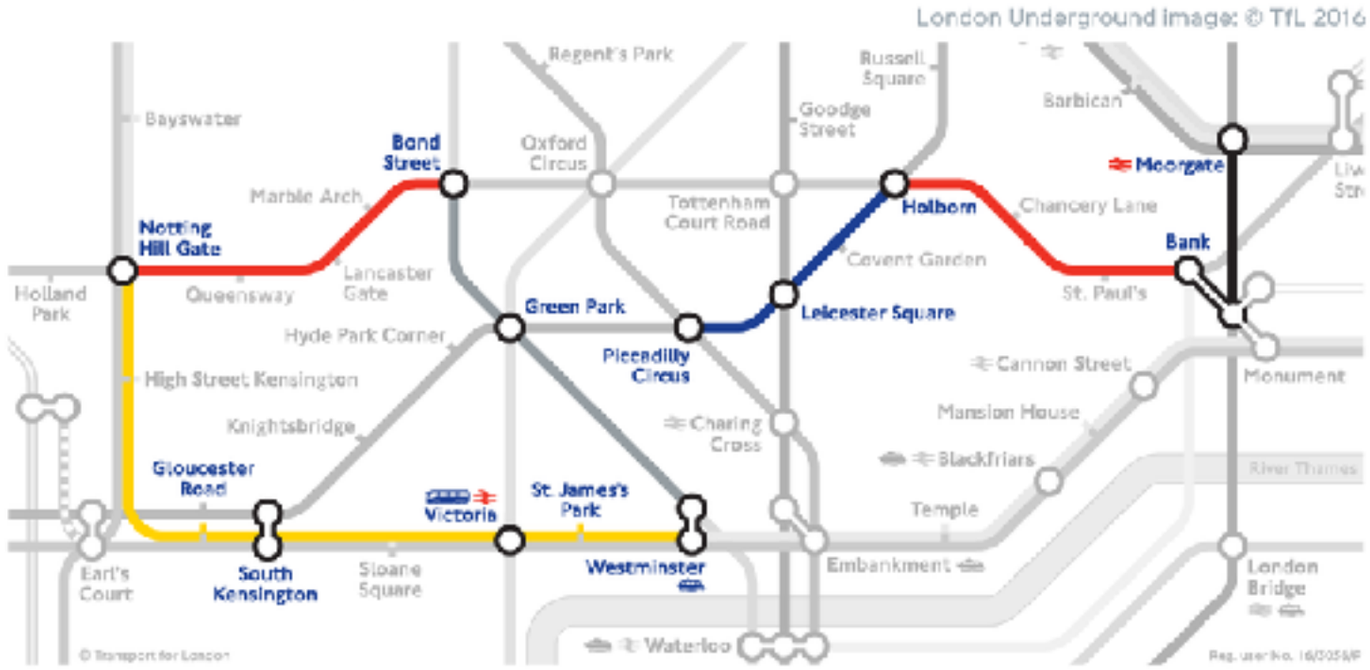
DNC: Allocator

- Memory usage vector **u**: values [0, 1] indicating presence of data
- Increases after writes
- Decreases after reads
- Writing to used positions is not allowed

DNC Experiments: Underground

Random Training Graph

London Underground



Traversal

Shortest

Underground Input:

(OxfordCircus, TottenhamCtRd, Central)
(TottenhamCtRd, OxfordCircus, Central)
(BakerSt, Marylebone, Circle)
(BakerSt, Marylebone, Bakerloo)
(BakerSt, OxfordCircus, Bakerloo)
...
(LeicesterSq, CharingCross, Northern)
(TottenhamCtRd, LeicesterSq, Northern)
(OxfordCircus, PiccadillyCircus, Bakerloo)
(OxfordCircus, NottingHillGate, Central)
(OxfordCircus, Euston, Victoria)

Traversal Question:

(BondSt, _, Central),
(_, _, Circle), (_, _, Circle),
(_, _, Circle), (_, _, Circle),
(_, _, Jubilee), (_, _, Jubilee),

Answer:

(BondSt, NottingHillGate, Central)
(NottingHillGate, GloucesterRd, Circle)
...
(Westminster, GreenPark, Jubilee)
(GreenPark, BondSt, Jubilee)

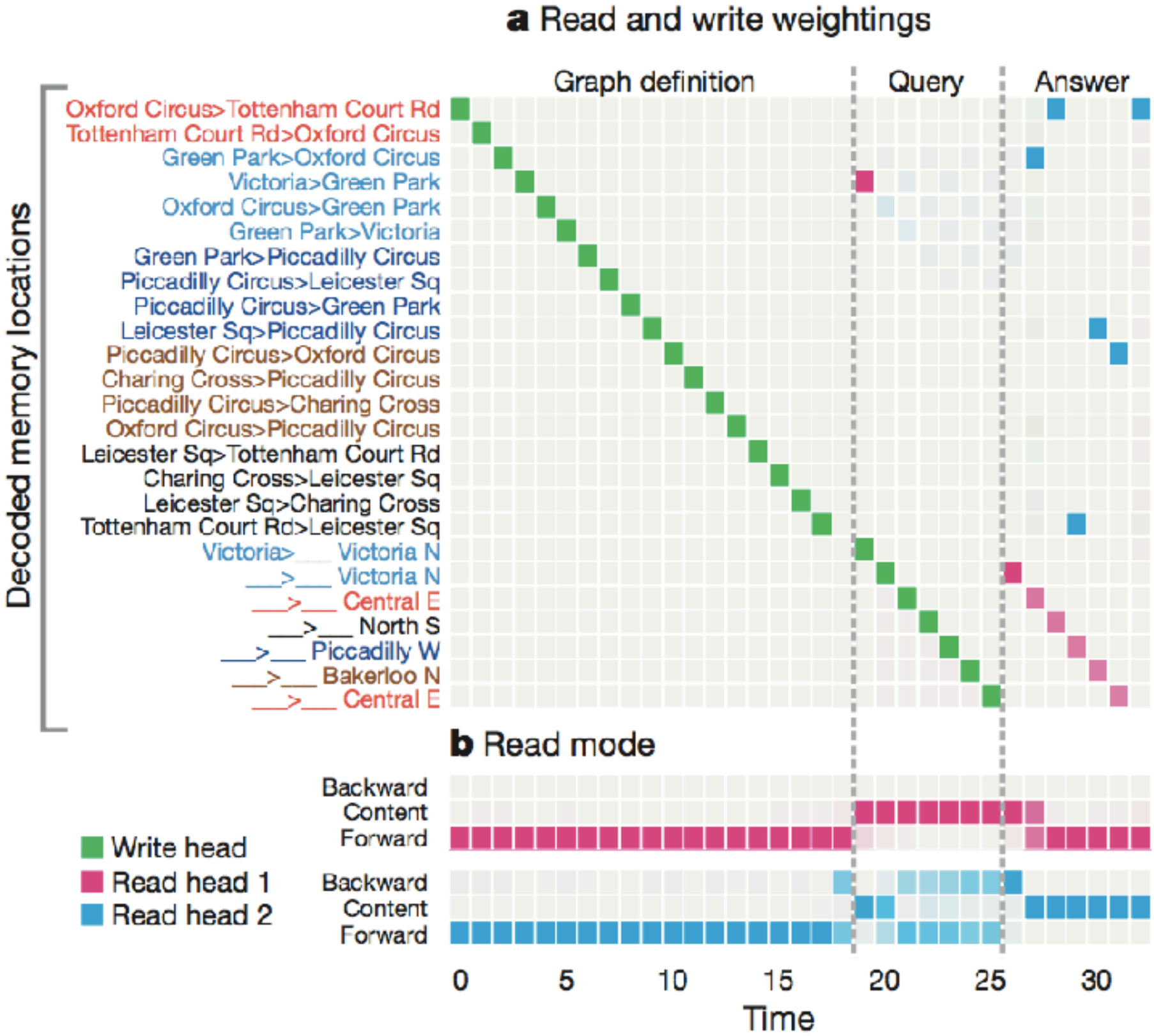
Shortest Path Question:

(Moorgate, PiccadillyCircus, _)

Answer:

(Moorgate, Bank, Northern)
(Bank, Holborn, Central)
(Holborn, LeicesterSq, Piccadilly)
(LeicesterSq, PiccadillyCircus, Piccadilly)

DNC Experiments: Underground



DNC Experiments

Family tree inference task

VIDEO

Live Slides web content

To view

Download the add-in.

liveslides.com/download

Start the presentation.

Structures summary

Neural Turing Machines Graves, et.al 2014	**Learning to Transduce with Unbounded Memory** Grefenstette, et.al 2015	**Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets** Mikolov, et.al 2015
Arrays	Stack	Stack
Associative arrays	Queue	Double-linked list
	DeQueue	

Is it possible to have Universal Neural Turing Machine?

Thank you for attention