Semester project presentation

Extending Dynamic Structure in Memory Network for Response Generation

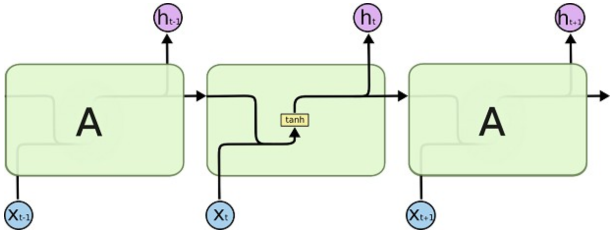
- Supervisor: Mi Fei
- Director: Prof .Boi Faltings
- Myself: André Cibils

Plan:

- 1. Introduction
- 2. Encoder-Decoder modification
- 3. Pointer Network modification
- 4. Conclusion
- 5. Questions

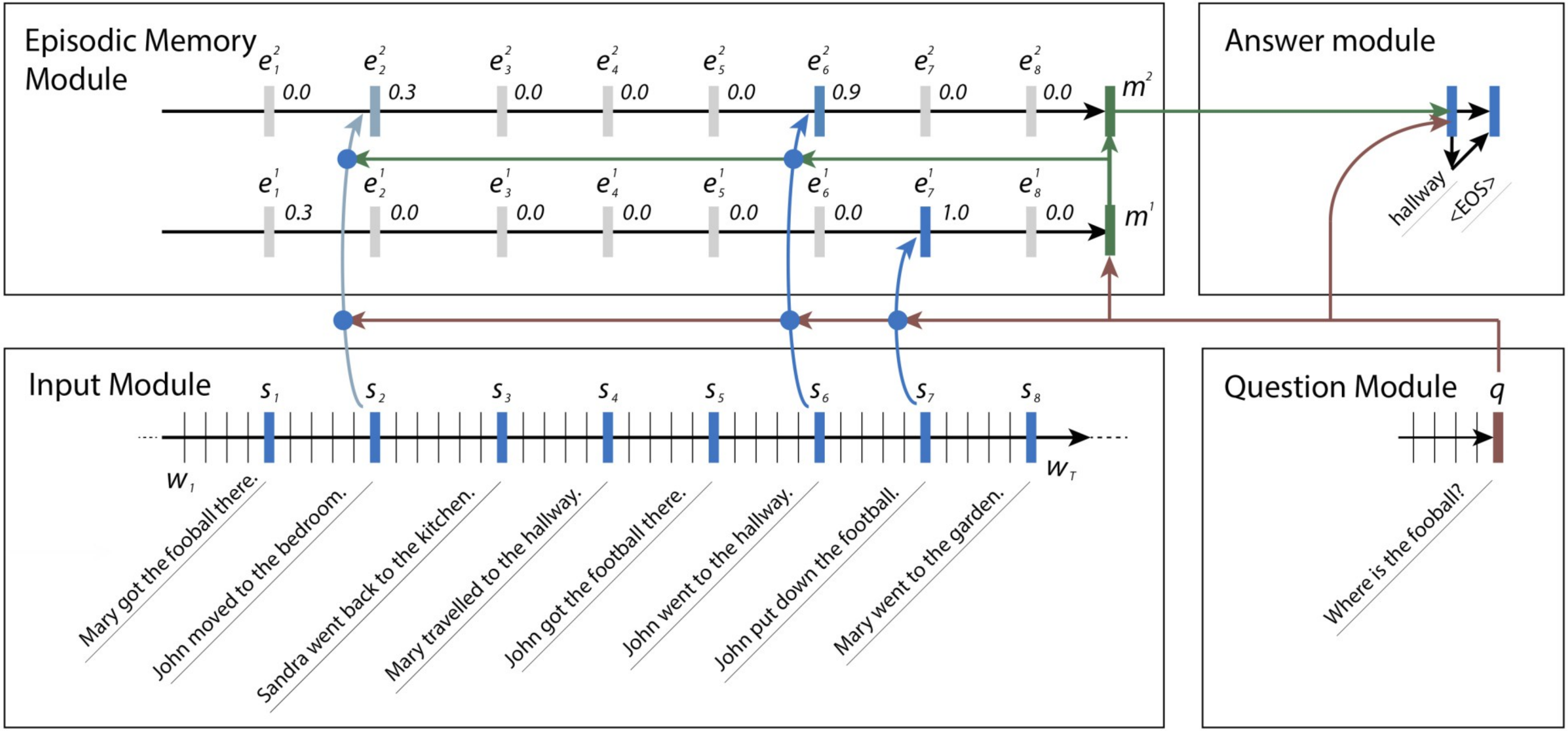
Dynamic Memory Network [1]

- 4 parts:
  - Input module
  - Question module
  - Episodic memory module
  - Answer module

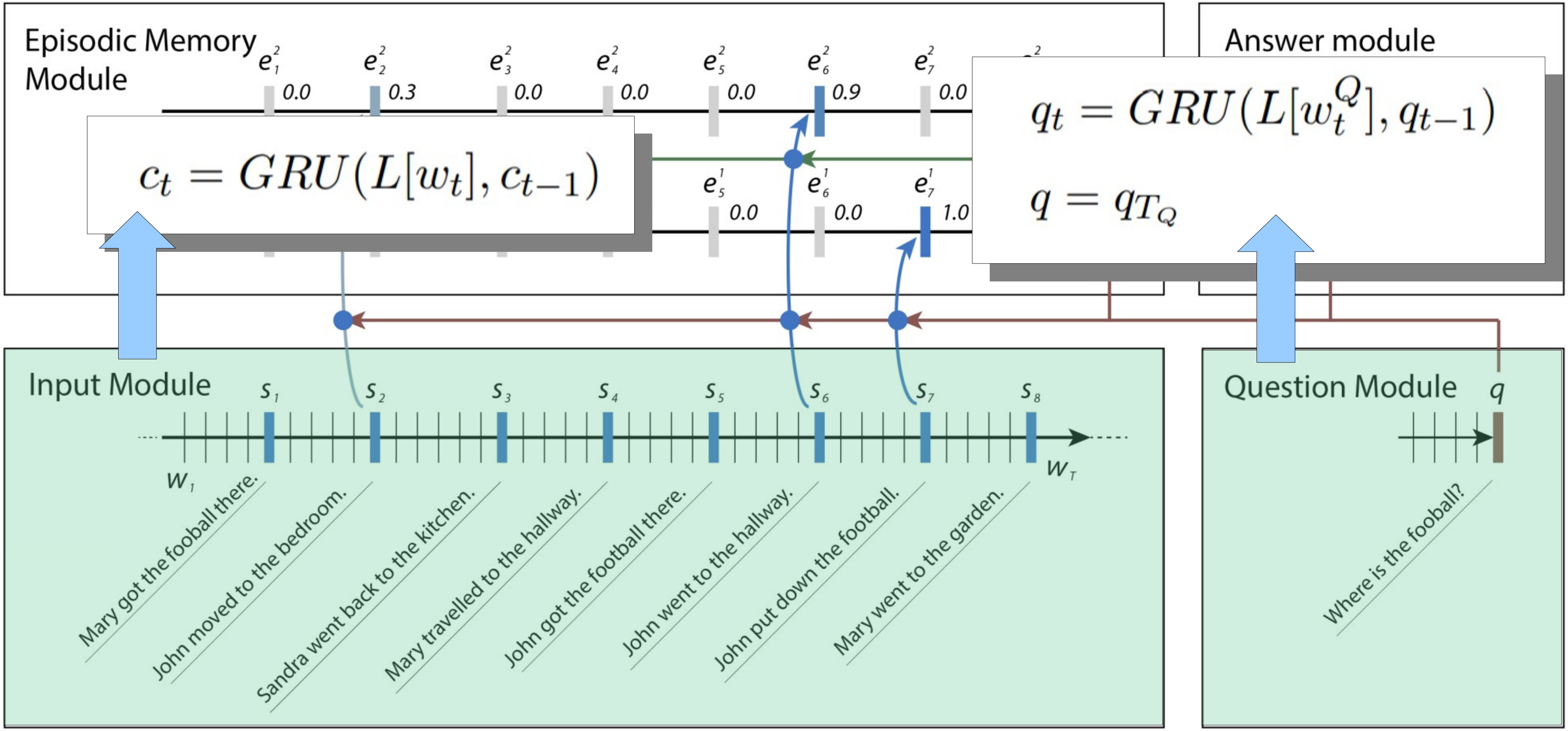


- Neural network based framework
- Internal memory representation (memory vector *m*)
- Uses lot of Gated Recurrent Units
- End-to-end

Dynamic Memory Network



Dynamic Memory Network

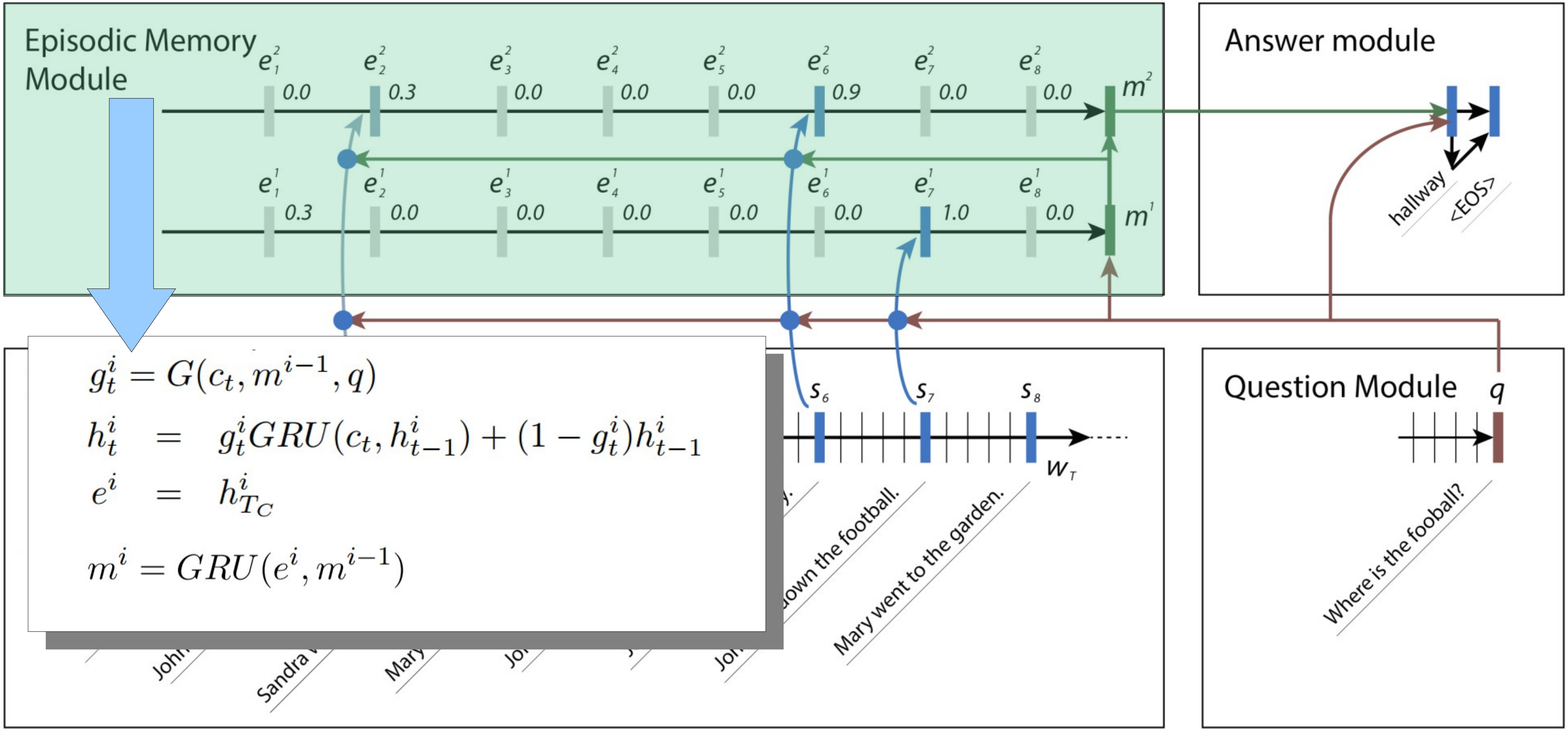


$$c_t = GRU(L[w_t], c_{t-1})$$

$$q_t = GRU(L[w_t^Q], q_{t-1})$$

$$q = q_{T_Q}$$

Dynamic Memory Network



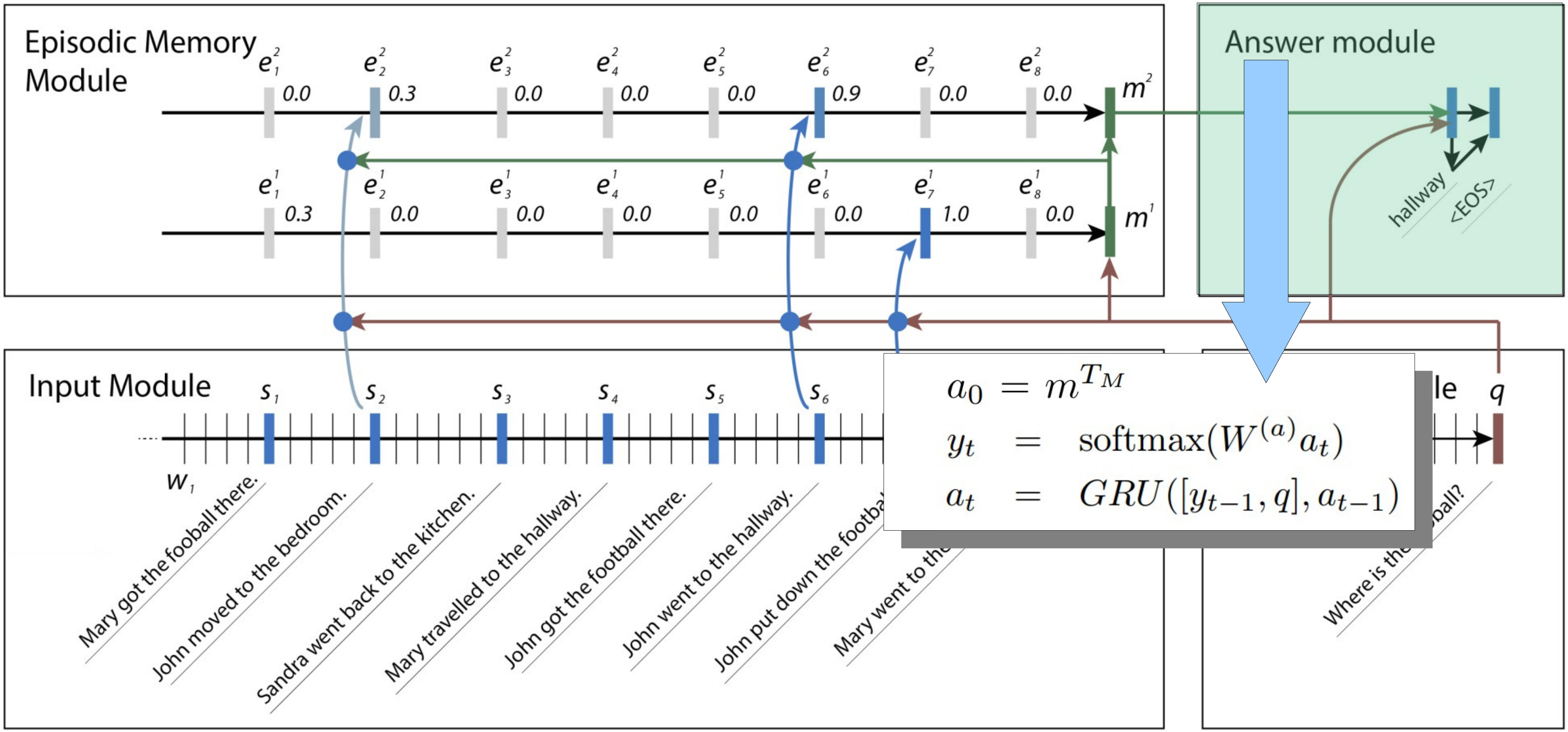
$$g_t^i = G(c_t, m^{i-1}, q)$$

$$h_t^i = g_t^i GRU(c_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i$$

$$e^i = h_{T_C}^i$$

$$m^i = GRU(e^i, m^{i-1})$$

Dynamic Memory Network



$$a_0 = m^{T_M}$$

$$y_t = \text{softmax}(W^{(a)} a_t)$$

$$a_t = GRU([y_{t-1}, q], a_{t-1})$$

Implementation details

- Supervised classification problem
- Model is end-to-end!
- Trained via back propagation through time and gradient descent
- Use L_2 regularisation

Goal?

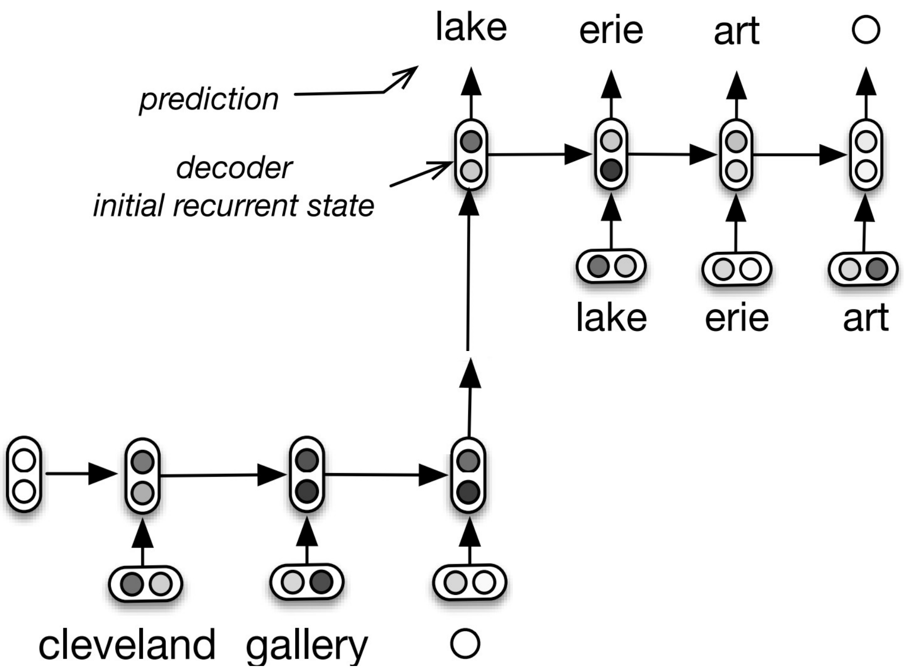
- Given a text and a question as input, generate a sentence as an answer

  This require the ability to retrieve facts and reason over them, but also to be able to generate a sentence

- General idea to solve this problem:
  - Use DMN as it showed multiple skills like deduction or coreference resolution
  - Combine it with a sentence generative model

First modification: Encoder-Decoder [2]

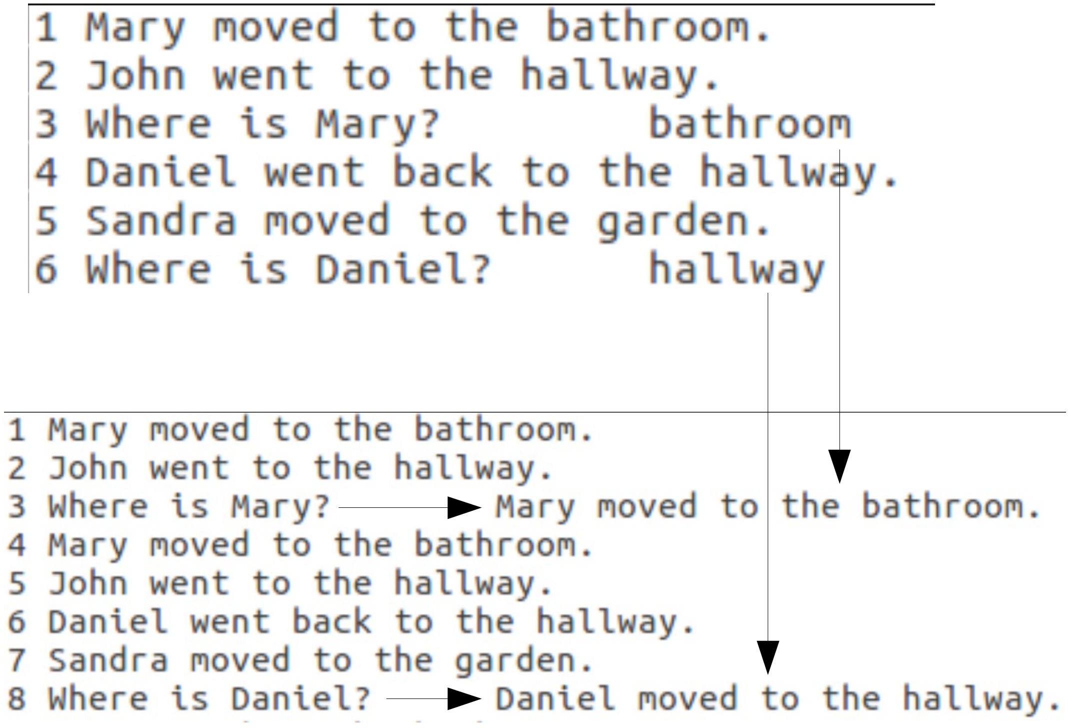
- Use DMN for all the reasoning part
- Use the Encoder-Decoder architecture to generate sentences



- Idea: see the answer module as a decoder
- It can now produce a fixed & limited number of words.

Implementation details

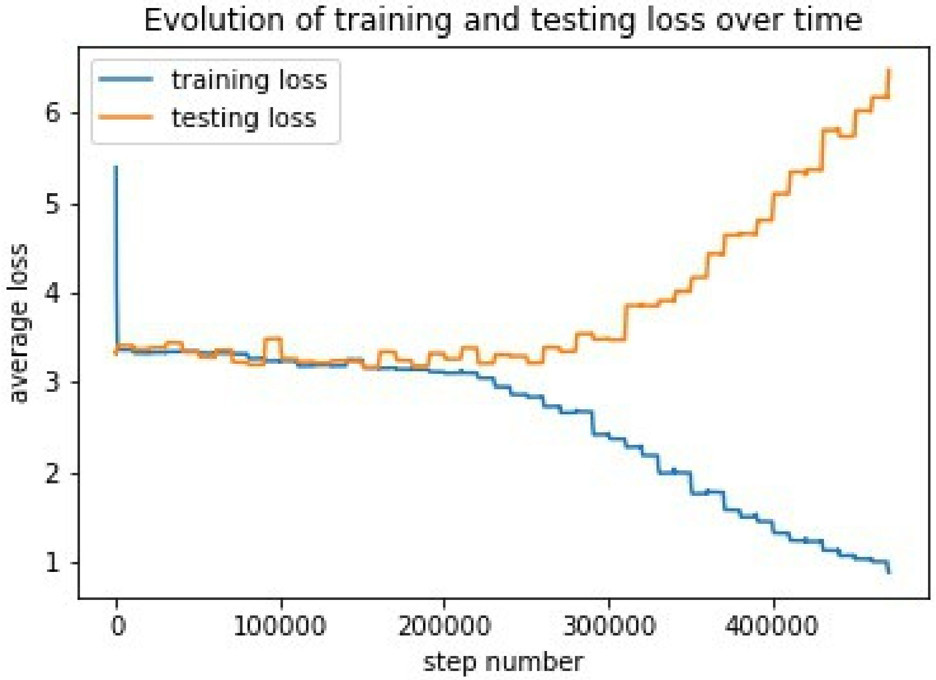
- Modification of the bAbi data set
- Simple sentences as answers
- Use GloVe to have word embeddings
- Answers always have the same architecture

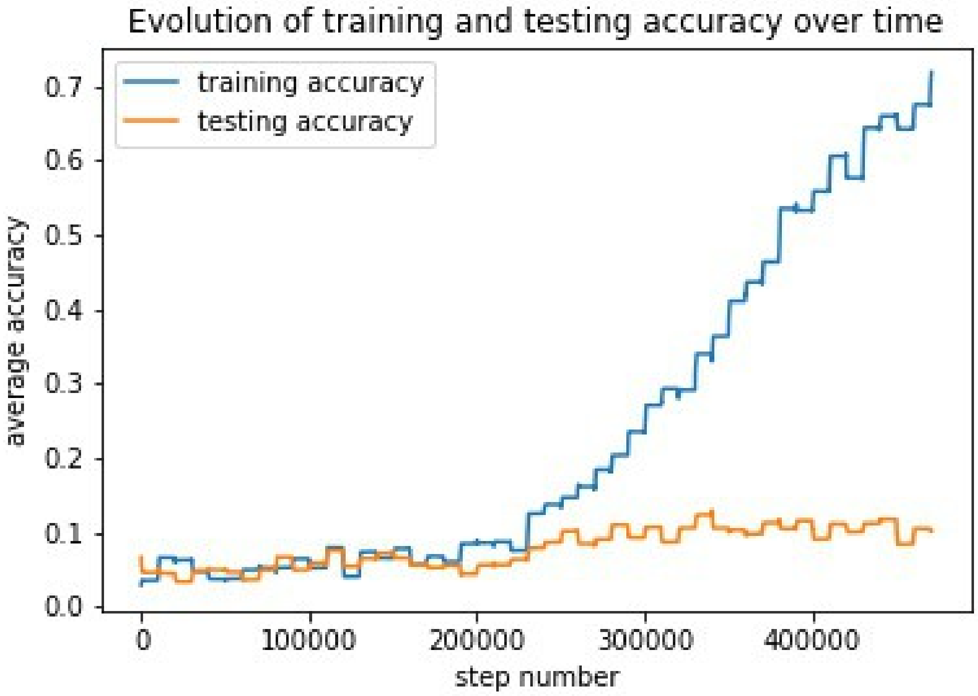


```
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary?          bathroom
4 Daniel went back to the hallway.
5 Sandra moved to the garden.
6 Where is Daniel?        hallway
```

```
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary? ──────► Mary moved to the bathroom.
4 Mary moved to the bathroom.
5 John went to the hallway.
6 Daniel went back to the hallway.
7 Sandra moved to the garden.
8 Where is Daniel? ────► Daniel moved to the hallway.
```

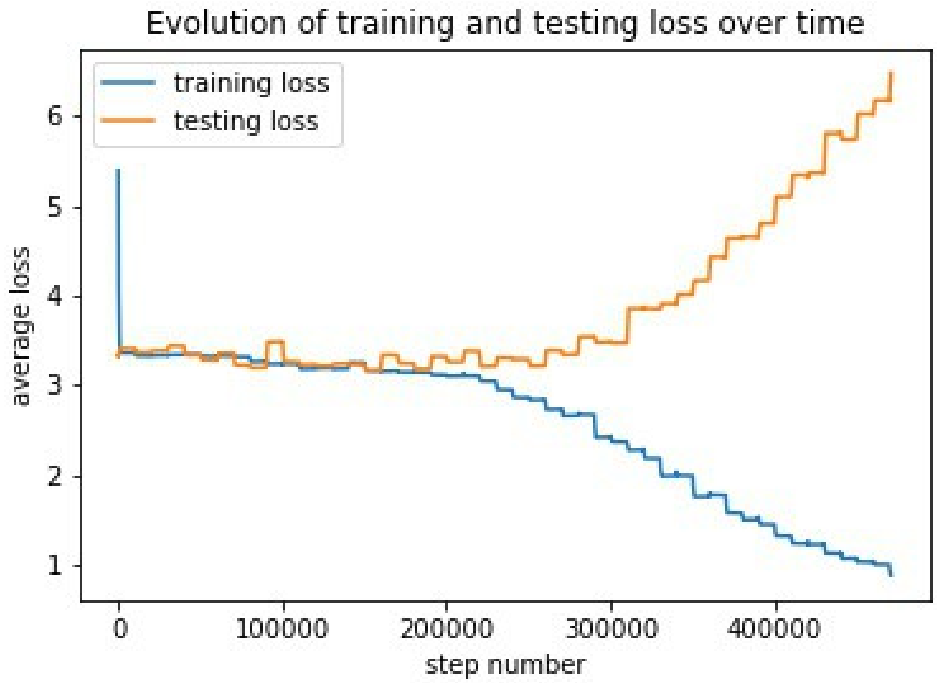
<Subject> moved to the <Place>. (ex: "Marie moved to the office.")

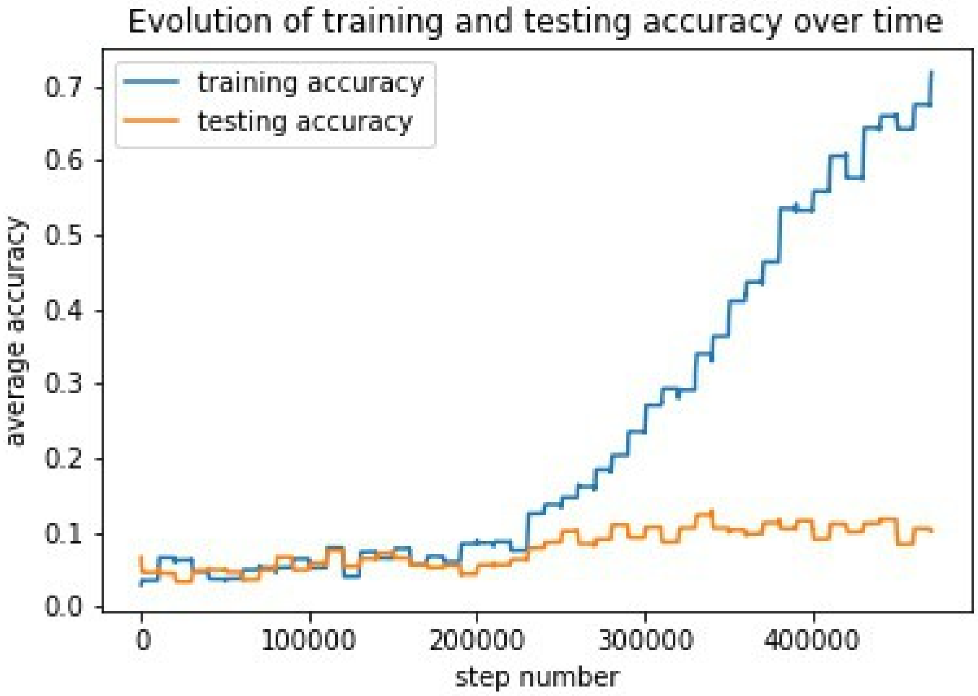


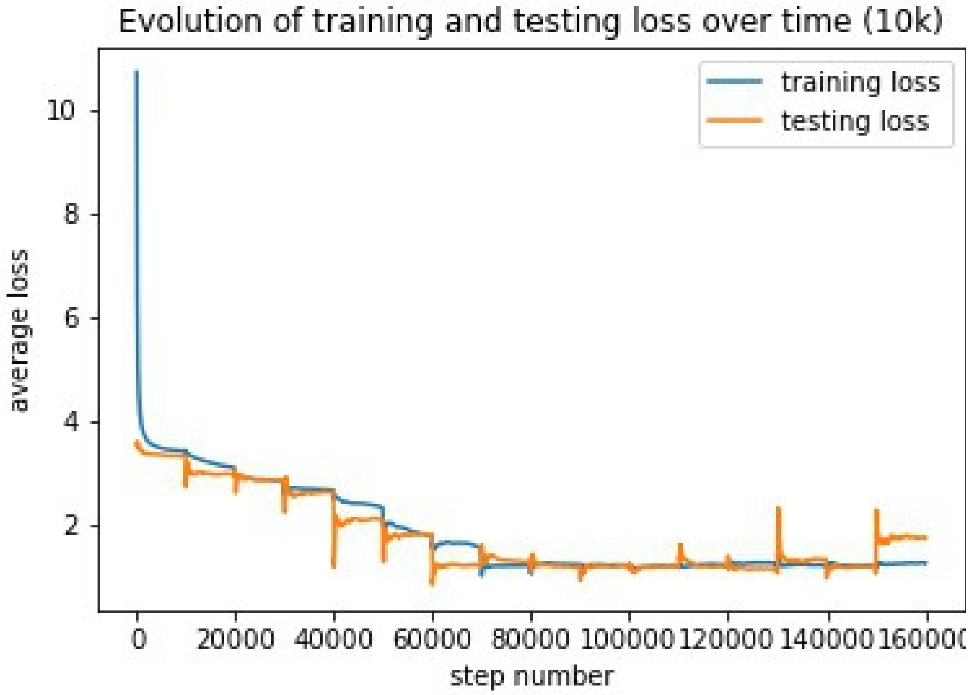


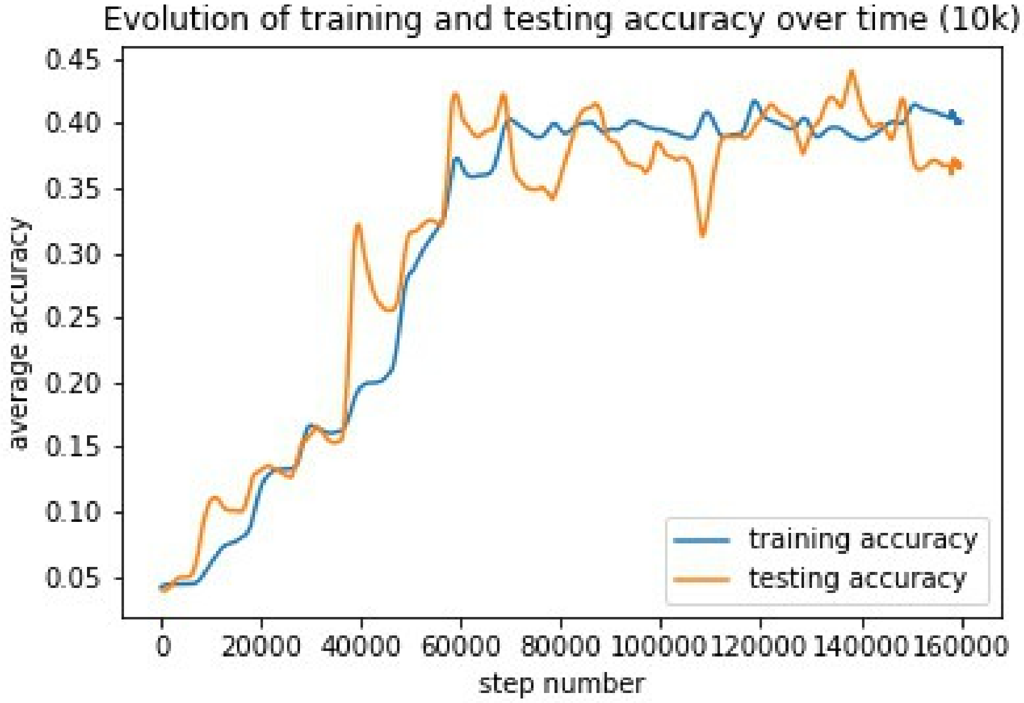
- Only 1000 exemples
- Over-fitting

- Accuracy: either 1 if the model predicted the right sentence or 0!


Evolution of training and testing loss over time
training loss
testing loss
average loss
1
2
3
4
5
6
0
100000
200000
300000
400000
step number


Evolution of training and testing accuracy over time
training accuracy
testing accuracy
average accuracy
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
100000
200000
300000
400000
step number


Evolution of training and testing loss over time (10k)
training loss
testing loss
average loss
2
4
6
8
10
0
20000
40000
60000
80000
100000
120000
140000
160000
step number


Evolution of training and testing accuracy over time (10k)
training accuracy
testing accuracy
average accuracy
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0
20000
40000
60000
80000
100000
120000
140000
160000
step number

First version of the Encoder-Decoder: analyse of the results

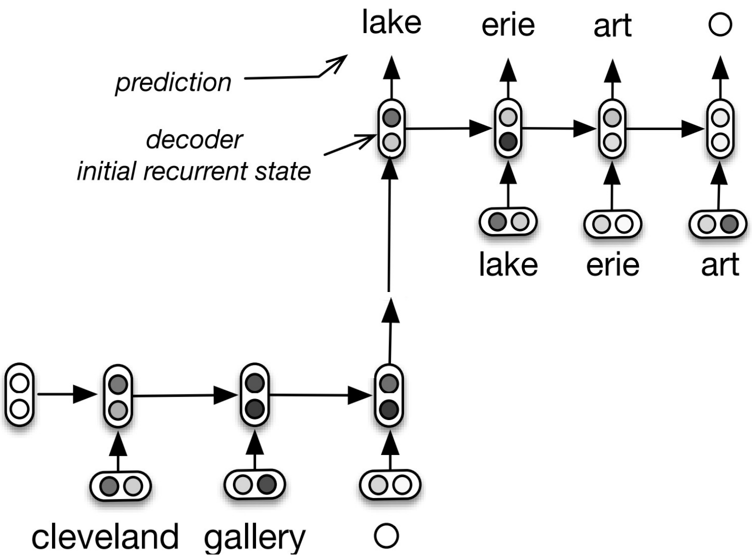
- High tendencie to overfit
- However, with a lot of data, seems to work

- Models focus on learning the easy part of the sentence: reasonning ability is compromised

<Subject> went to the <Place>. (ex: "Marie went to the office.")

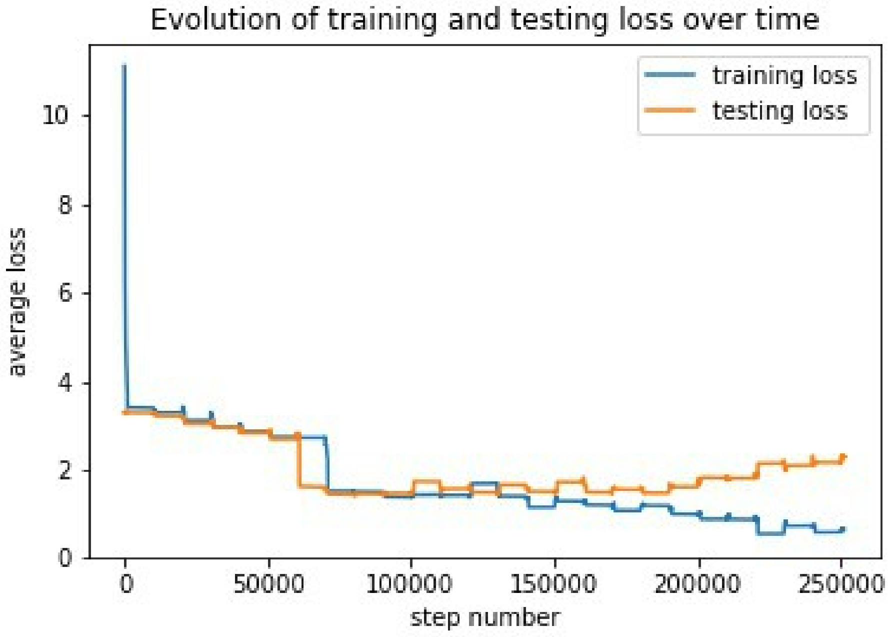
Encoder-Decoder – Why does it fail?

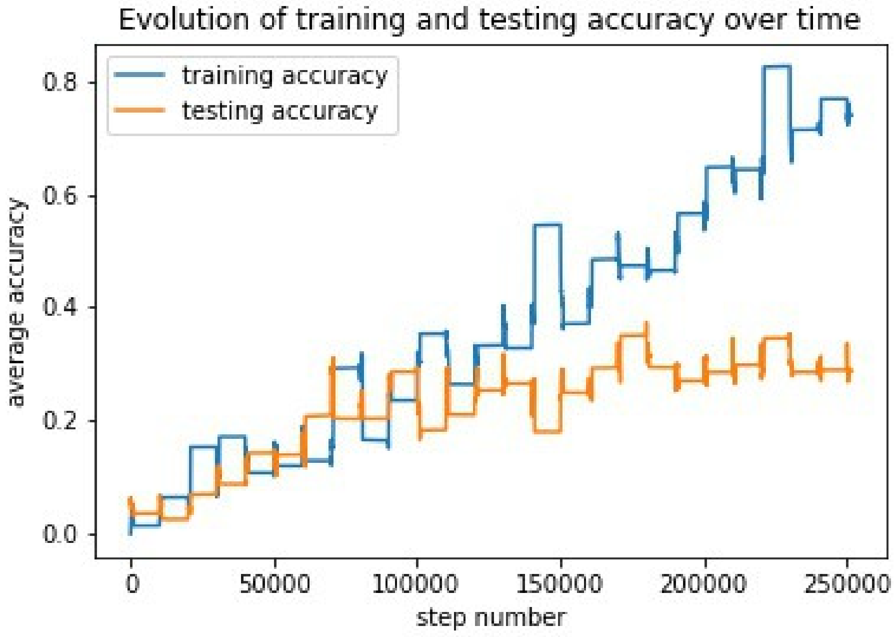
- The GRU decoder is hard to train.
  - Idea: make it easier for the model
- Small modification of the answer module: Add the original memory vector as input



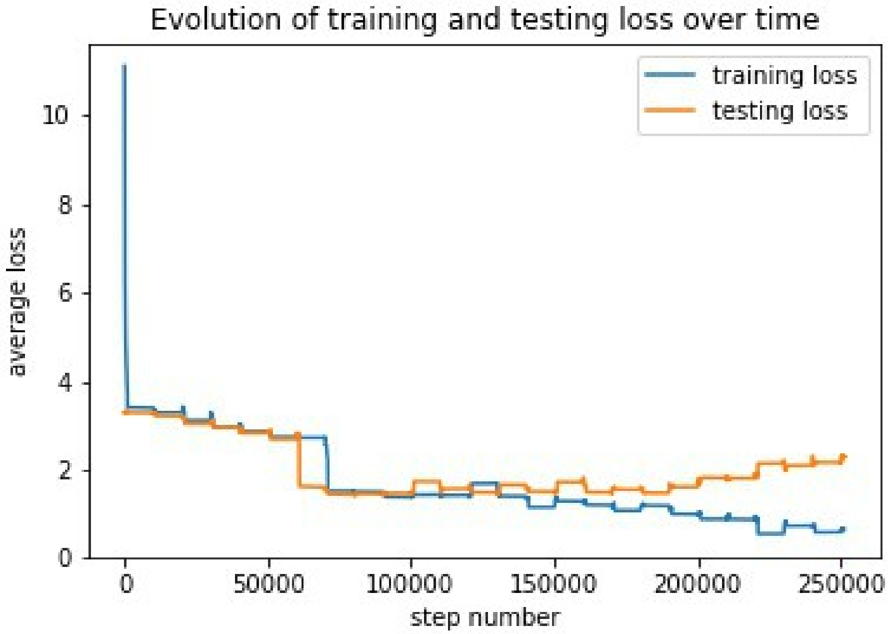
$$y_t = \text{softmax}(W^{(a)} a_t)$$
$$a_t = GRU([y_{t-1}, q], a_{t-1})$$
$$\downarrow$$
$$a_t = GRU([y_{t-1}, q, a_0], a_{t-1})$$

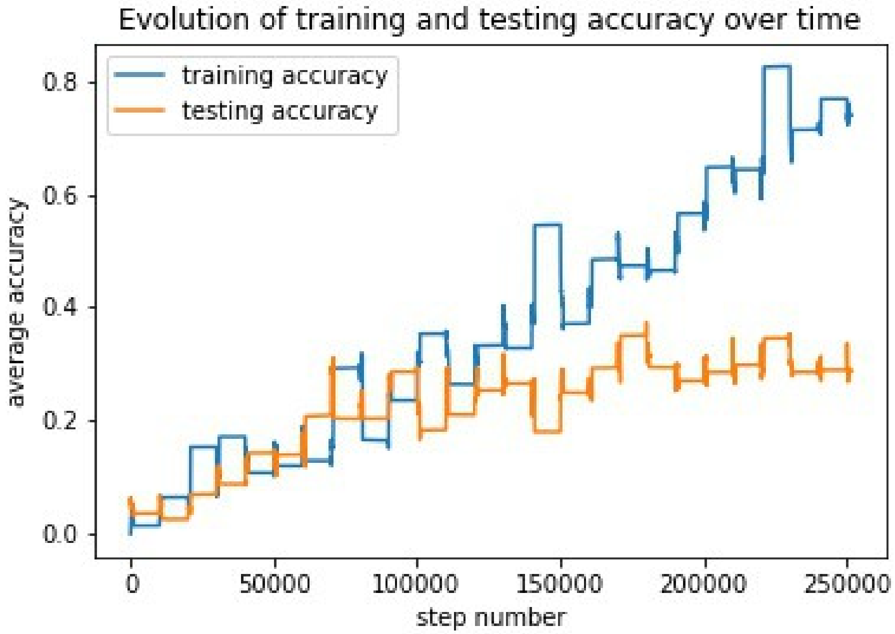
2. Encoder-Decoder modification

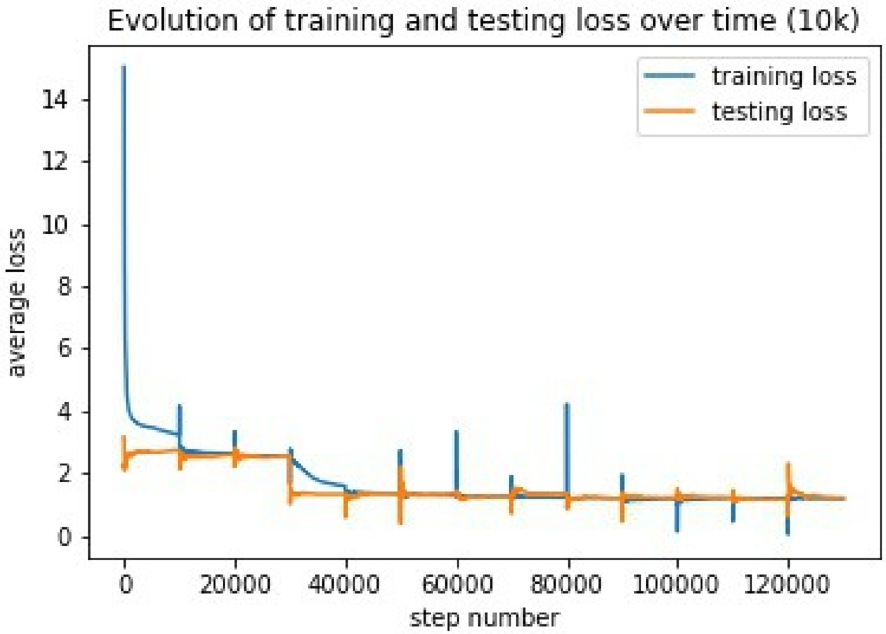


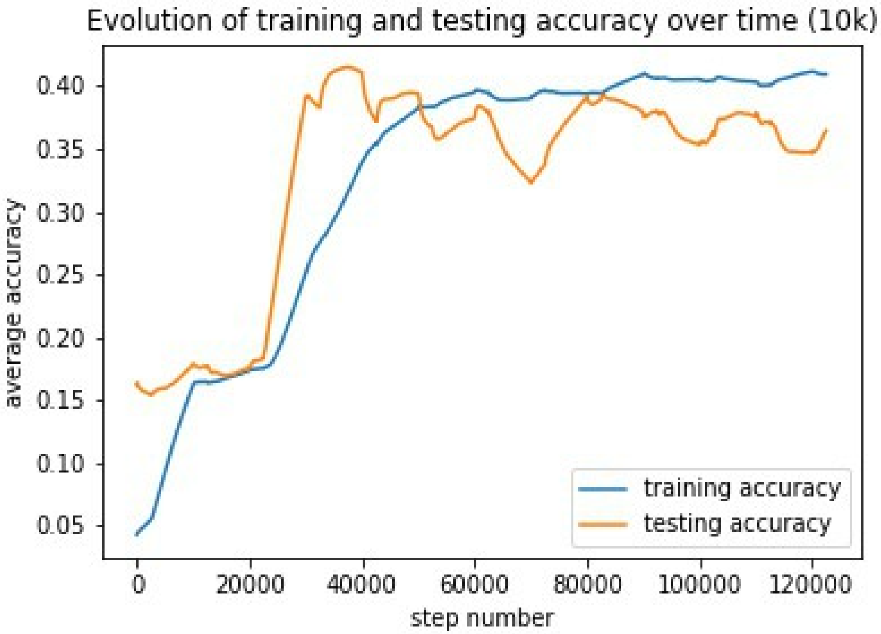


- Only 1000 exemples
- Over-fitting, again


Evolution of training and testing loss over time


Evolution of training and testing accuracy over time

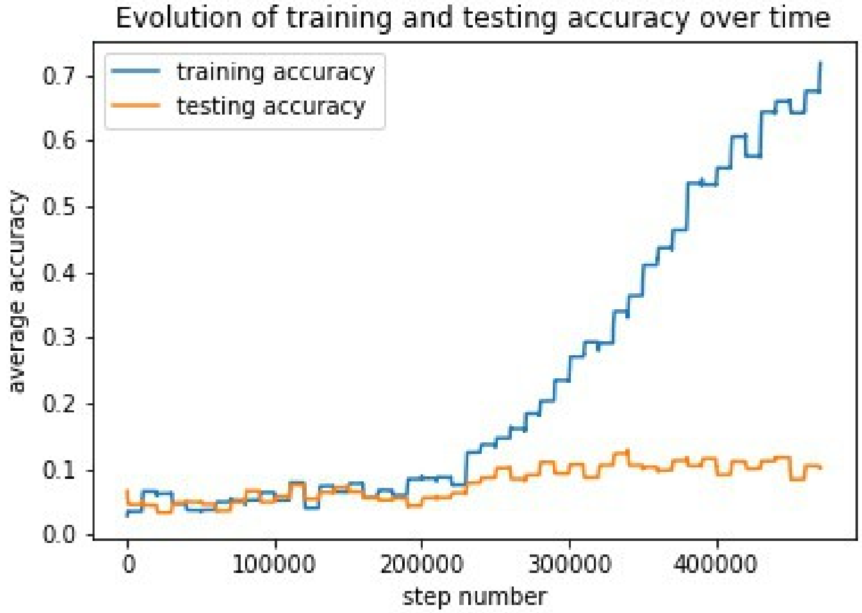

Evolution of training and testing loss over time (10k)


Evolution of training and testing accuracy over time (10k)

Evolution of accuracy on a small dataset

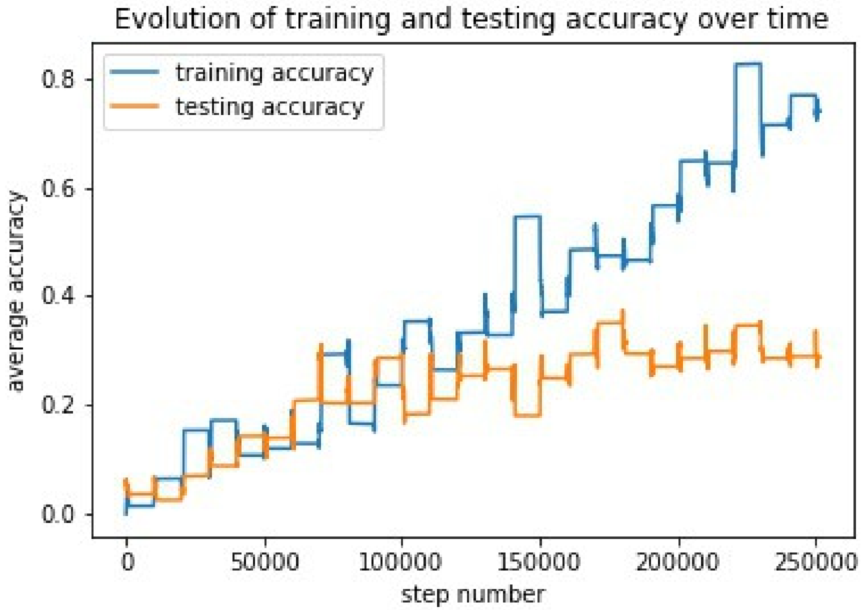
First modification

Testing accuracy is caped at ~10%

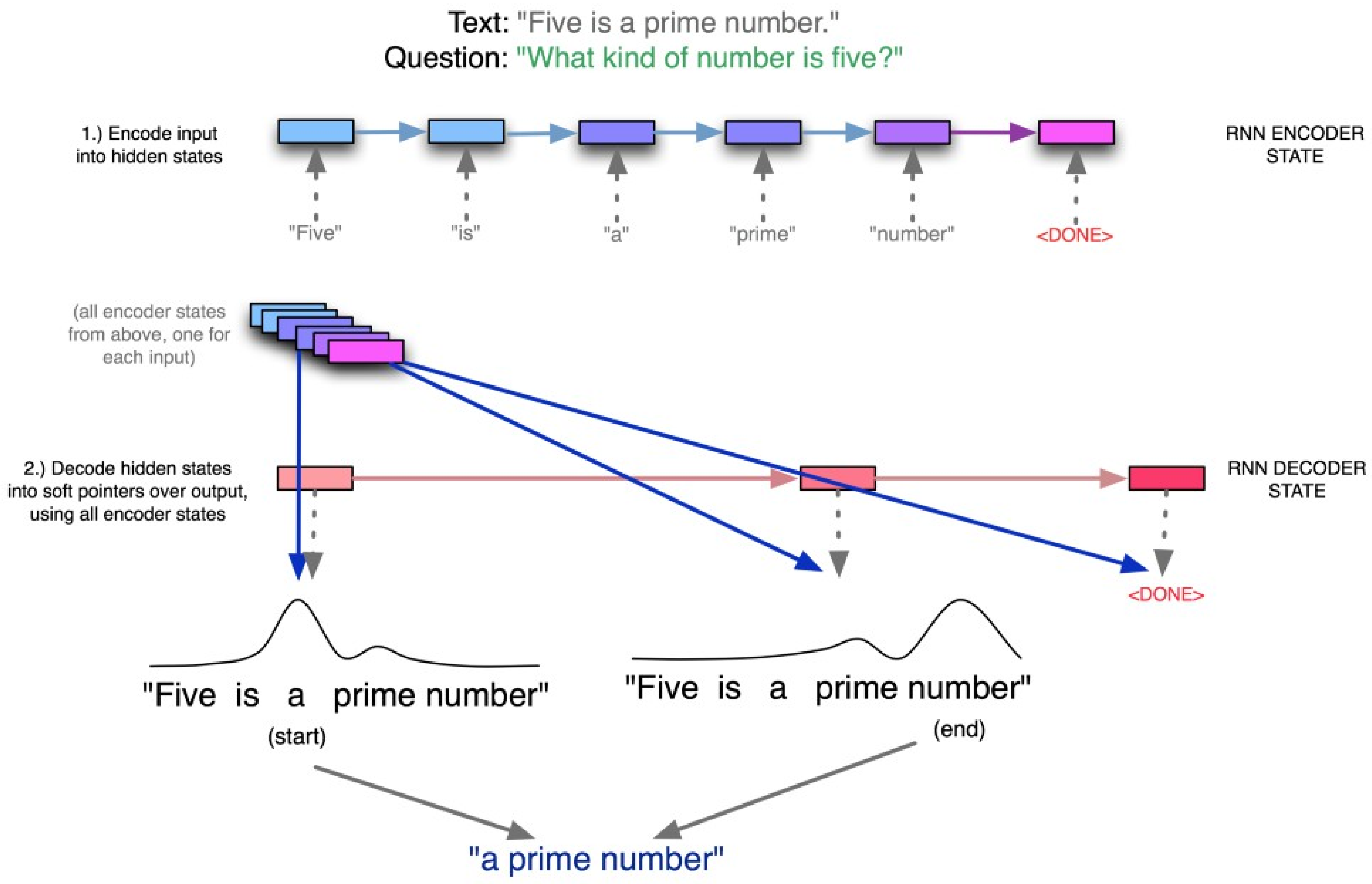


With the memory vector as input

Testing accuracy goes up to ~25%

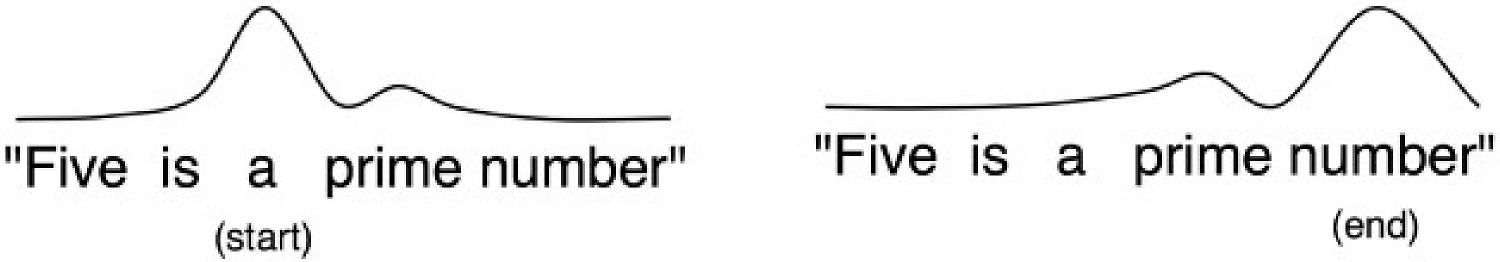


Pointer Nets - Quickly


Text: "Five is a prime number."
Question: "What kind of number is five?"
1.) Encode input into hidden states
RNN ENCODER STATE
"Five"
"is"
"a"
"prime"
"number"
<DONE>
(all encoder states from above, one for each input)
2.) Decode hidden states into soft pointers over output, using all encoder states
RNN DECODER STATE
<DONE>
"Five is a prime number"
(start)
"Five is a prime number"
(end)
"a prime number"

Pointer network modification [3]

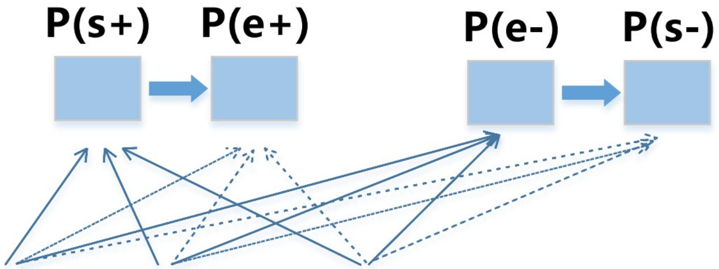
- GRU Decoder is hard to train, but pointer networks should be easy.
- Hypothesis: the answer is in the text
- The goal is to find where it is, i.e. Produce a start and end idx



Implementation details

- SQuAD data set – complex and diverse
  - Multiple topics from wikipedia
  - Every answer to each question is a segment of text
- Two-directional prediction: produce start idx, then use it to produce end idx & redo it the other way, then merge

Prediction Layer



$$P(s+) = softmax(W^{(s+)} m^{T_M})$$

$$P(e+) = softmax(W^{(e+)} m^{T_M} + W^{(c+)} c_{s+})$$

Pointer network modification: results

- Right now, not learning...

Conclusion

- Goal: Modify DMN to produce multiple word answer

  Need multiple abilities: fact retrieving and resoning, but also sentence generation

- Encoder-Decoder Architecture

  Promising results, but decoder LSTM are too hard to train

- Pointer Network Architecture

  Need more refining

- Other Architectures? Merge of the previous ones?

Referencs

- [1] Ask Me Anything: Dynamic Memory Network for Question Answering, 5 Mar 2016
- [2] A Hierarchical Recurrent Encoder-Decoder for Generative Context-Aware Query Suggestion, 8 Jul 2015
- [3] Exploring Question Understanding and Adaptation in Neural-Network-Based Question Answering, 25 Mar 2017

Questions?