Pointer Networks

Oriol Vinyals, Meire Fortunato and Navdeep Jaitly

Presented by: Israa Alqassem

Background

Recurrent Neural Networks (RNN) &

Sequence to Sequence Models (Seq2Seq)

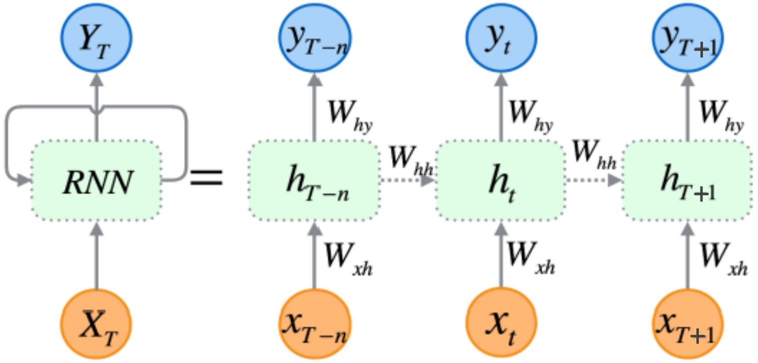
RNN

- RNN is a generalization of feedforward neural networks to sequences
  - $(x_1, \dots, x_T)$: input sequence of length T
  - $(y_1, \dots, y_T)$: output sequence of length T

$$h_t = sigm\left(W^{hx} x_t + W^{hh} h_{t-1}\right)$$

$$y_t = W^{yh} h_t$$

How to apply an RNN to problems whose input and the output sequences have different lengths with complicated and non-monotonic relationships?



Seq2Seq Model

- Estimate the conditional probability

$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T)$$

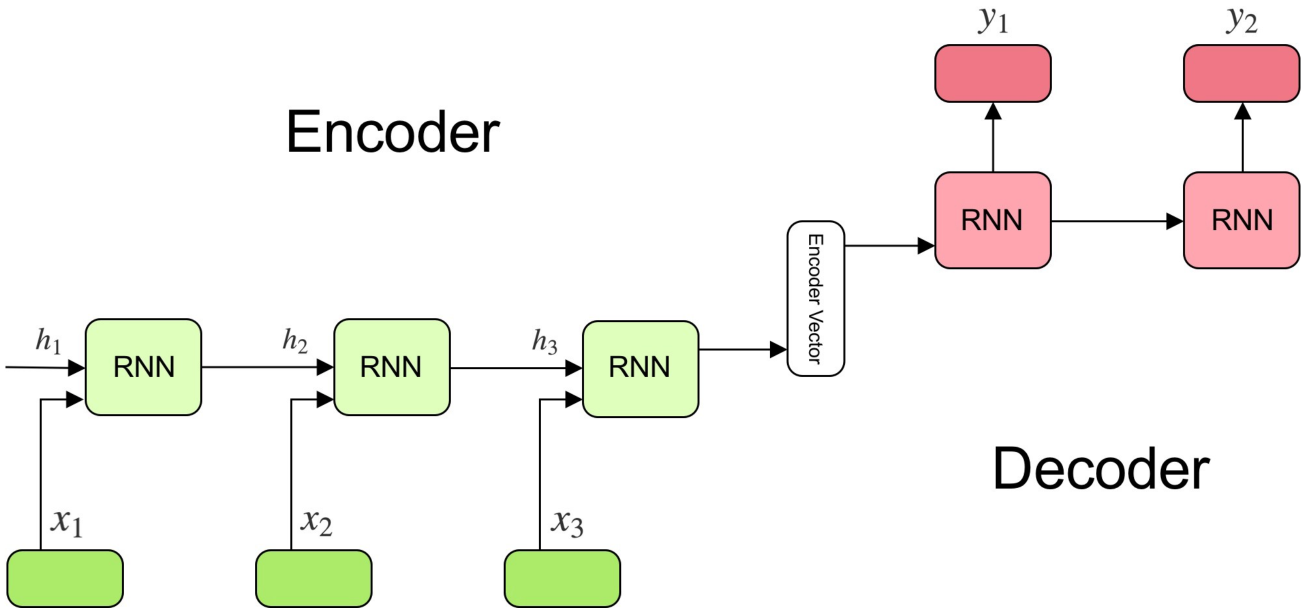
- First, obtaining the fixed dimensional representation $\boldsymbol{v}$ of the input sequence given by the last hidden state

- Then, compute the probability of $y_1, \dots, y_{T'}$ with an initial hidden state is set to the representation $\boldsymbol{v}$ of $x_1, \dots, x_T$

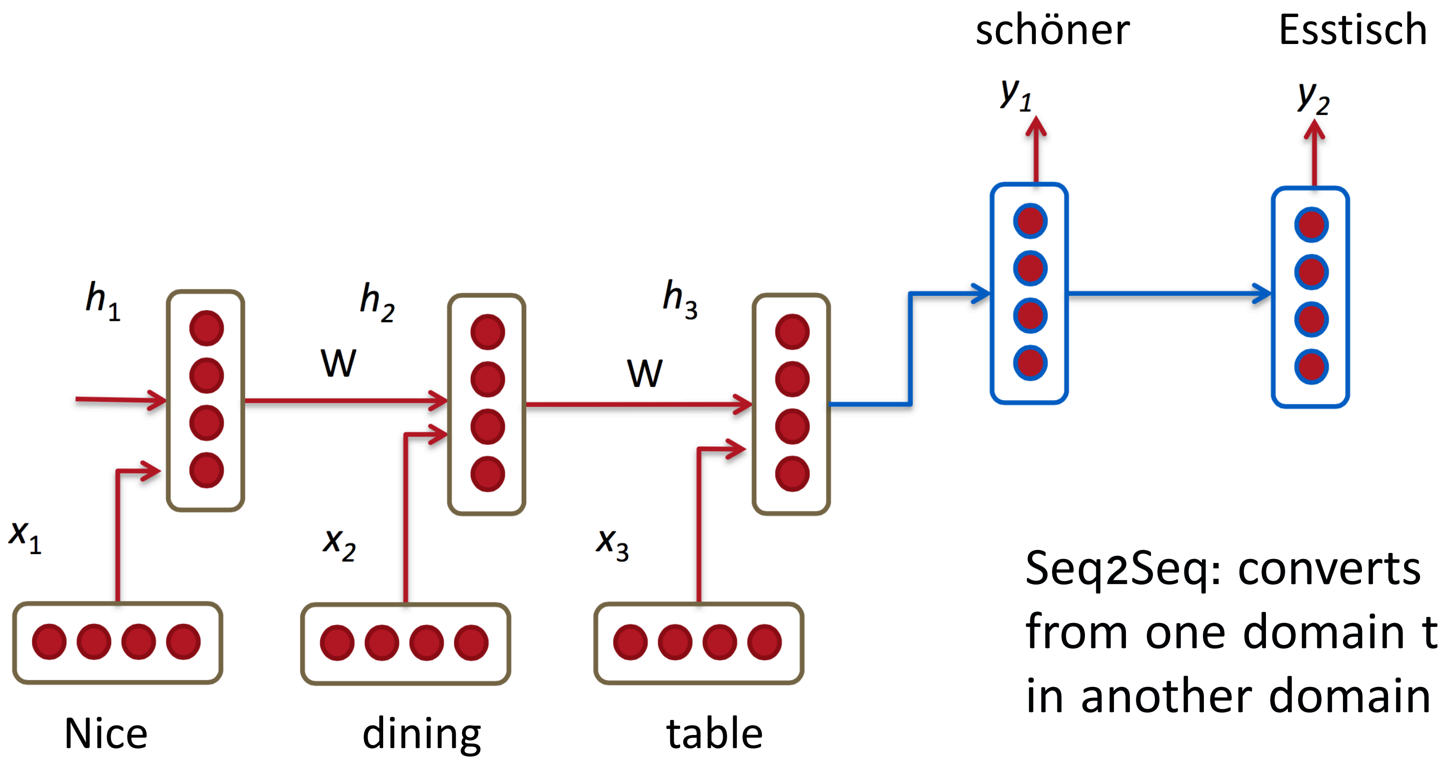
$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t=1}^{t=T'} p(y_t | v, y_1, \dots, y_{t-1})$$

Seq2Seq

1. Encoder: encodes the input sequence into a fixed length vector called a context vector
2. Decoder: predicts the output sequence using the context vector



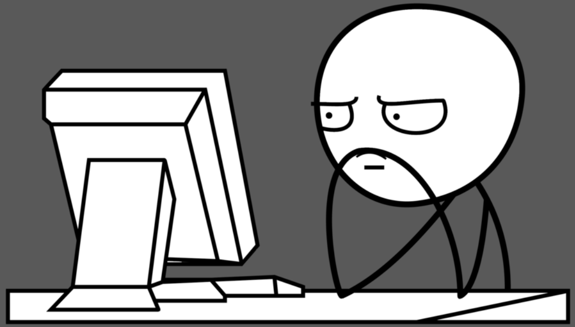
Seq2Seq



Seq2Seq: converts sequences from one domain to sequences in another domain

Varying output dictionary

A new Family of Problems



Unlike Seq2Seq, the target vocabulary of the output labels is NOT fixed

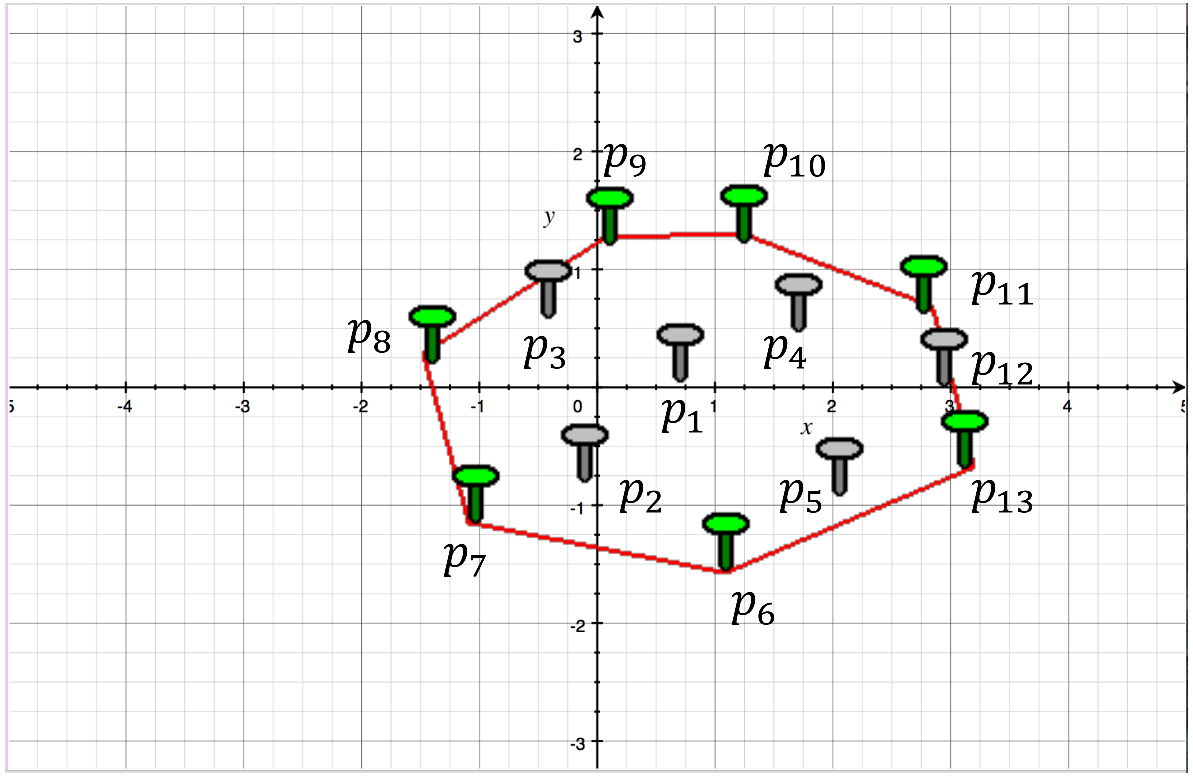
- Given a set $P = \{p_1, p_2, \dots, p_n\}$, find the (unique) minimal convex set containing P

Convex Hull

$$C^p = \{C_1, \dots, C_{m(P)}\}$$

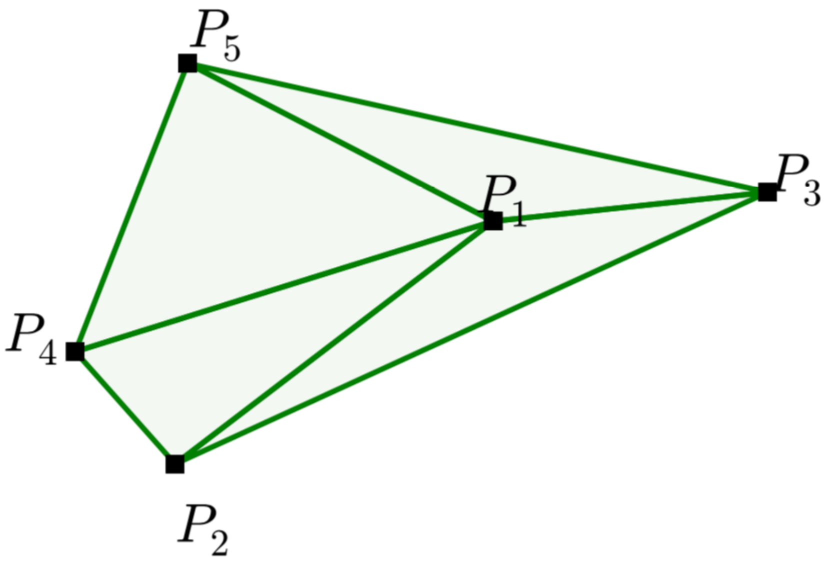
$$m_P \in [1, n]$$

- The outputs are pointers to the input elements
- Variable length input/output



Another combinatorial search problem…

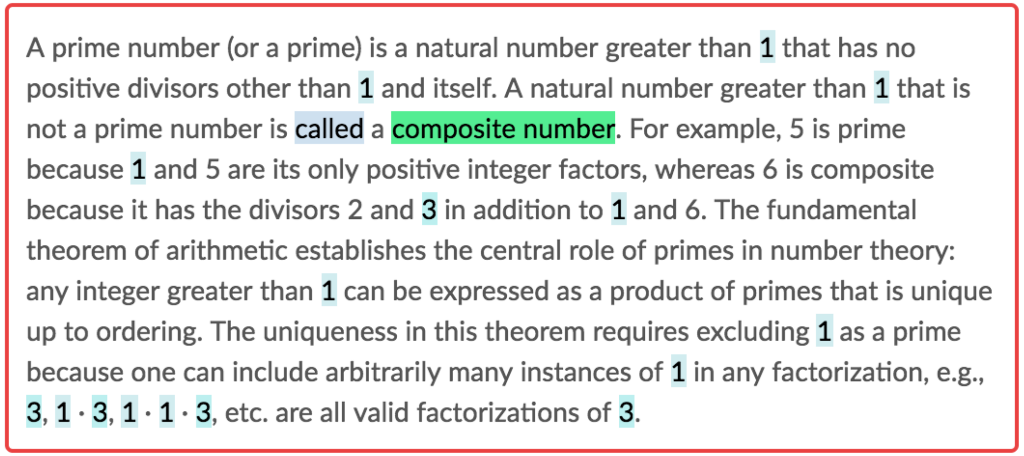
Delaunay Triangulation



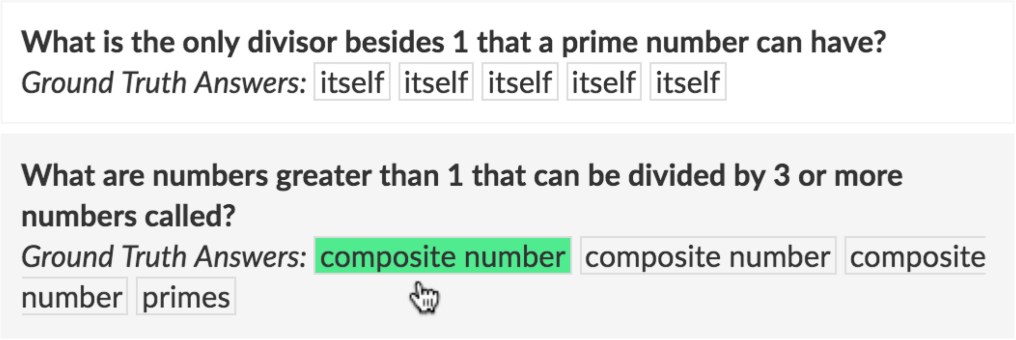
(b) Input $\mathcal{P} = \{P_1, \ldots, P_5\}$, and the output $\mathcal{C}^{\mathcal{P}} = \{\Rightarrow, (1,2,4), (1,4,5), (1,3,5), (1,2,3), \Leftarrow\}$ representing its Delaunay Triangulation.

SQuAD Example

Stanford **Qu**estion **A**nswering **D**ataset (SQuAD)



A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. A natural number greater than 1 that is not a prime number is called a composite number. For example, 5 is prime because 1 and 5 are its only positive integer factors, whereas 6 is composite because it has the divisors 2 and 3 in addition to 1 and 6. The fundamental theorem of arithmetic establishes the central role of primes in number theory: any integer greater than 1 can be expressed as a product of primes that is unique up to ordering. The uniqueness in this theorem requires excluding 1 as a prime because one can include arbitrarily many instances of 1 in any factorization, e.g., 3, 1 · 3, 1 · 1 · 3, etc. are all valid factorizations of 3.



What is the only divisor besides 1 that a prime number can have?
Ground Truth Answers: itself itself itself itself itself

What are numbers greater than 1 that can be divided by 3 or more numbers called?
Ground Truth Answers: composite number composite number composite number primes

..but Seq2Seq models can’t solve these problems

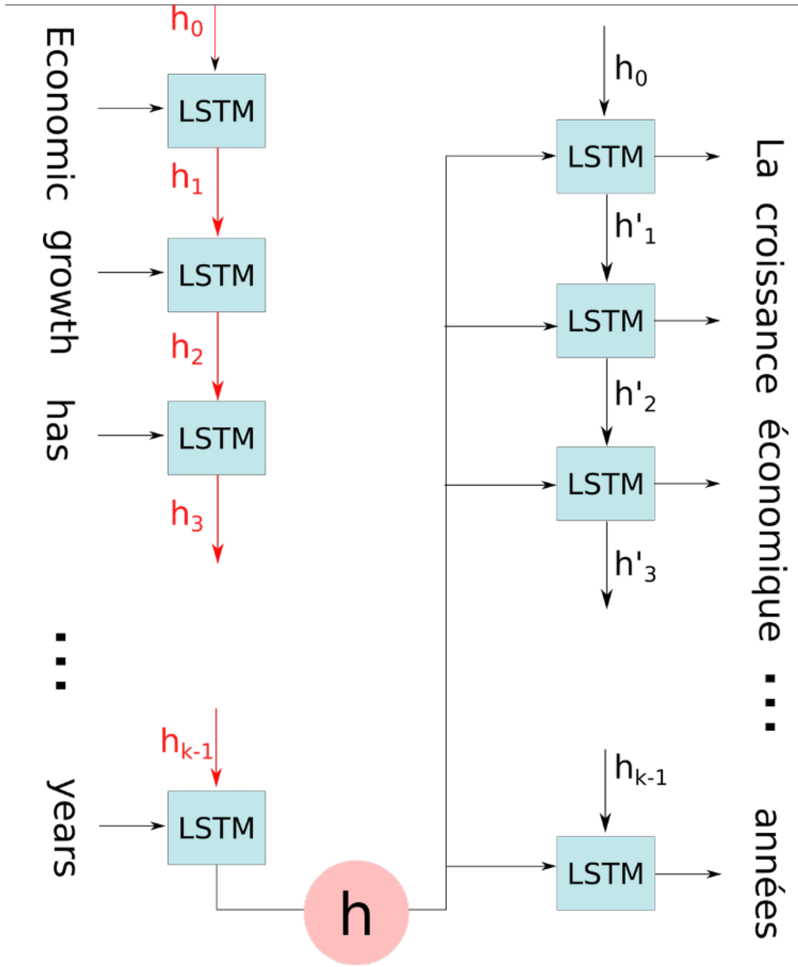
Limitations of Seq2Seq Models

- The dictionary of output labels have to be fixed
- The context vector is fixed for varied length input
- Diminishing influence of previous states on later states

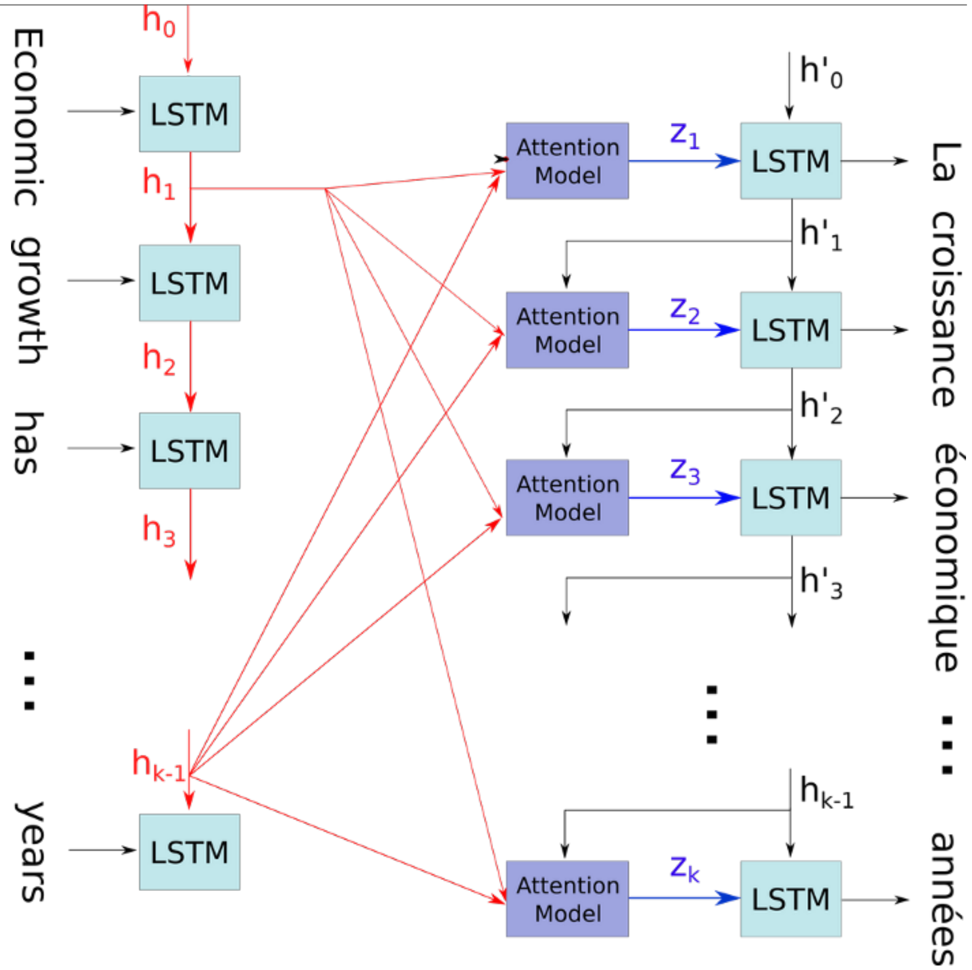
Attention Mechanism

- Blend the encoder states to propagate extra information to the decoder
- Create a weighted combination of all input states
  - Different weights for different position in output labels
- Major performance boost for data with longer term dependencies

Attention model for translation



vs.

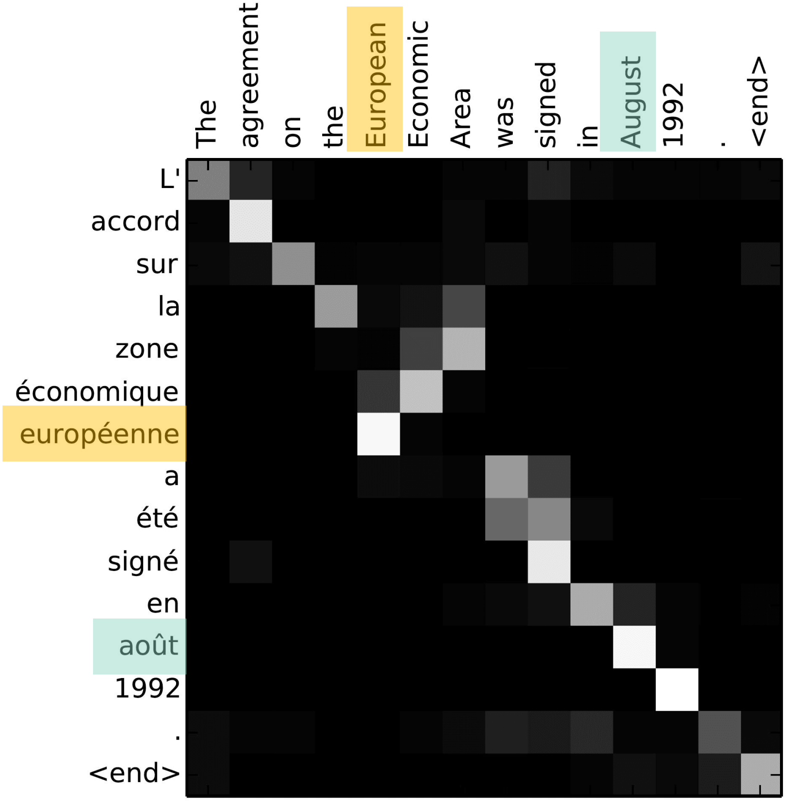


Attention model

- Encoder hidden states: $e_1, \dots, e_n$
- Decoder hidden states: $d_1, \dots, d_m,\ m \in [1, n]$
- For LSTM RNNs, we use the state after the output gate has been component-wise multiplied by the cell activation
- We compute the attention vector at each output time i as follows:
  - $u_j^i = v^T \tanh\left(W_1 e_j + W_2 d_i\right)$
  - $a_j^i = softmax(u_j^i)$
  - $d_i' = \sum_{j=1}^{n} a_j^i e_j$

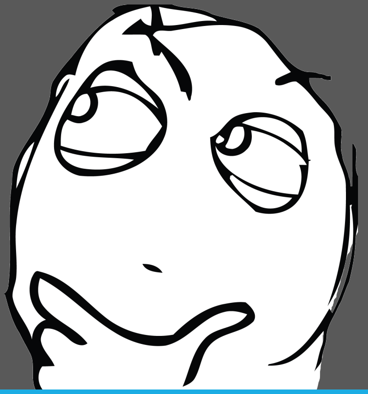
Visualized Attention

- Which part of input is relevant for this part of output?



Attention mechanism is to blend the encoder states to propagate extra information to the decoder

…instead, use the attention vector as pointers to the input elements



Pointer Networks

In a nutshell…

a) The neural architecture learns to predict conditional probability of an output sequence

b) Output sequence contains pointers to input elements

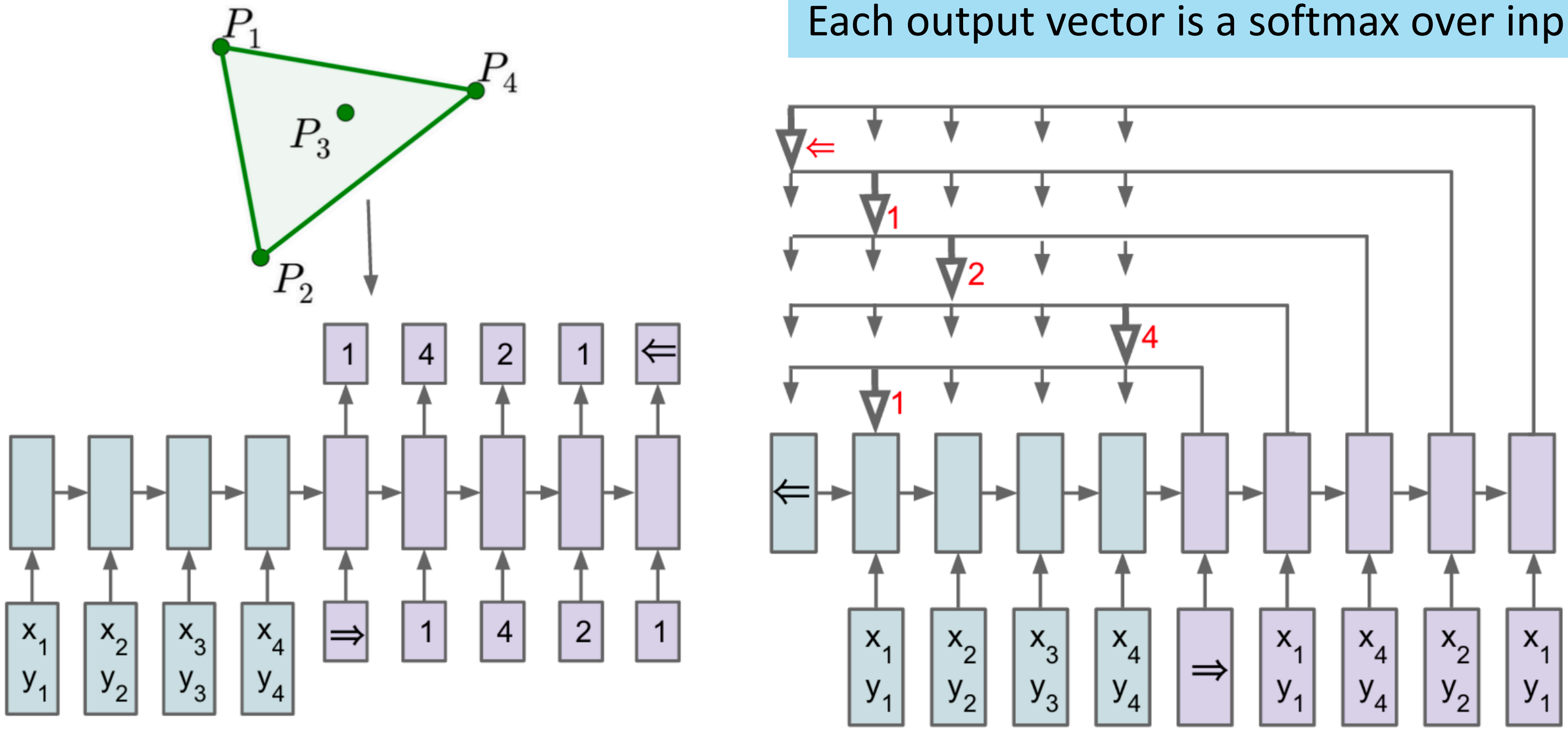
c) The output dictionary (i.e., solution space or search space) depends on the input elements

d) The model uses attention to point to input indices that will be selected as outputs

e) The model improves over seq2seq with attention by allowing output dictionary to be dependent on the inputs

The Model

Each output vector is a softmax over input elements



(a) Sequence-to-Sequence

(b) Ptr-Net

Pointer Networks

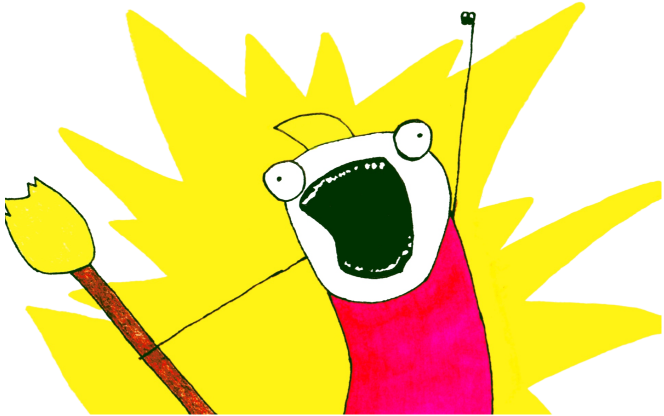
An encoding RNN converts the input sequence to a code

That is fed to the decoding network

It produces a vector that modulates a content-based attention mechanism over inputs

The output of the attention mechanism is a softmax distribution with dictionary size equal to the length of the input

The output is a subset of the input



Formally,

RNNs with Attention

$$u_j^i = v^T \tanh\left(W_1 e_j + W_2 d_i\right)$$

$$a_j^i = softmax(u_j^i)$$

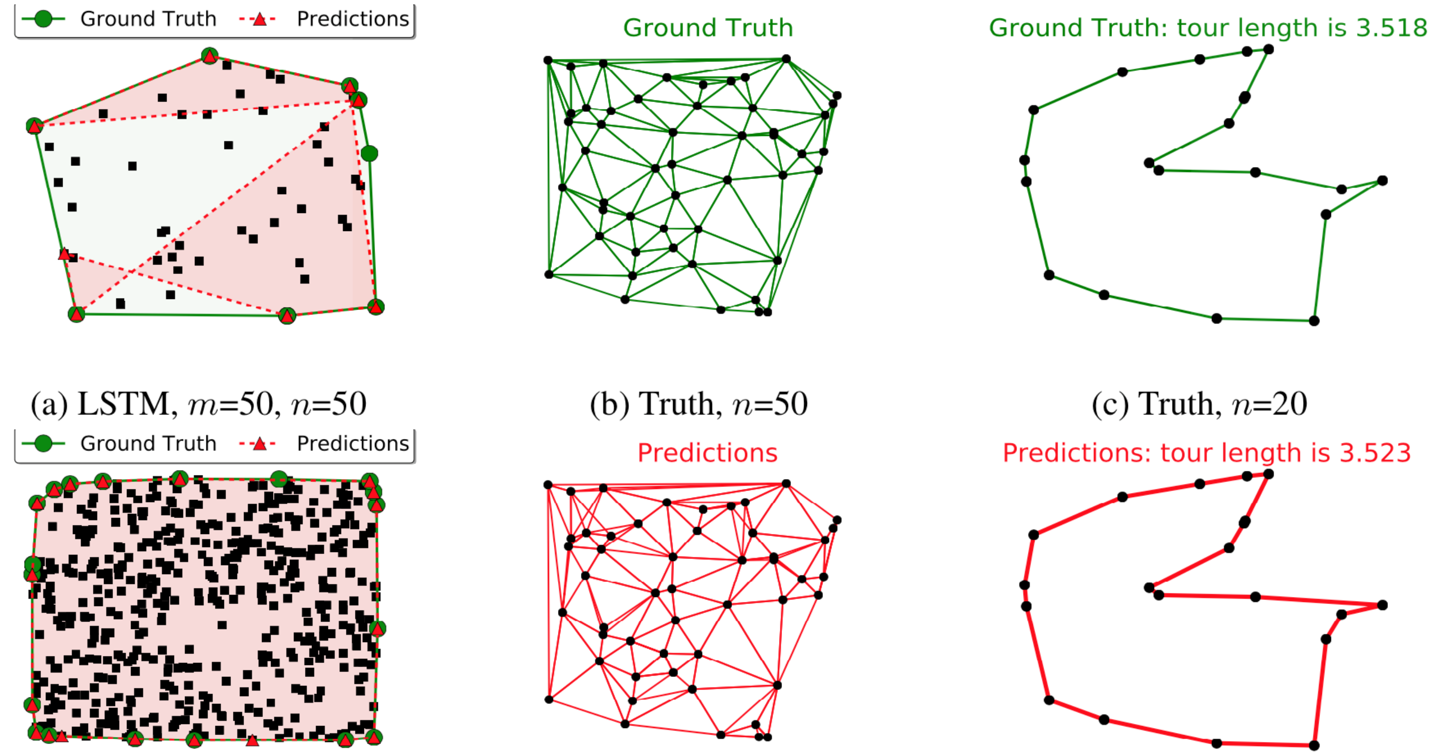
$$d_i' = \sum_{j=1}^{n} a_j^i e_j$$

Pointer Networks

$$u_j^i = v^T \tanh\left(W_1 e_j + W_2 d_i\right)$$

$$a_j^i = softmax(u_j^i)$$

Results



(a) LSTM, m=50, n=50

(b) Truth, n=50

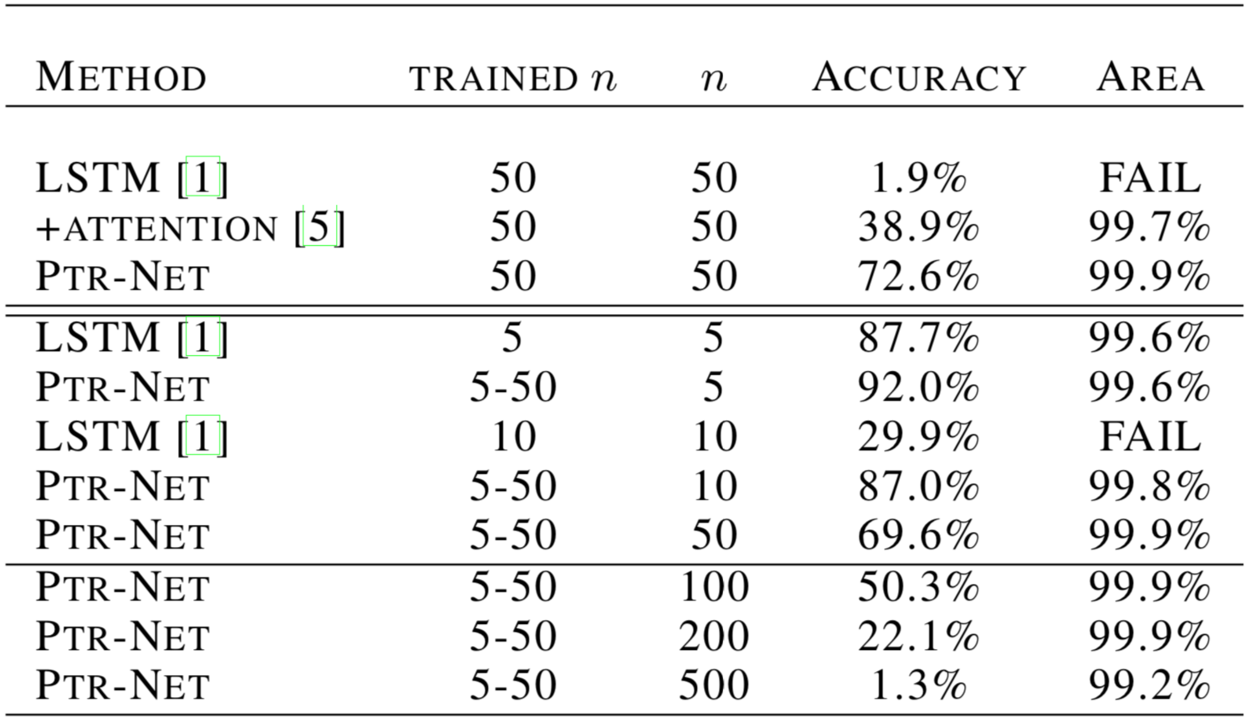
(c) Truth, n=20

(d) Ptr-Net, m=5-50, n=500

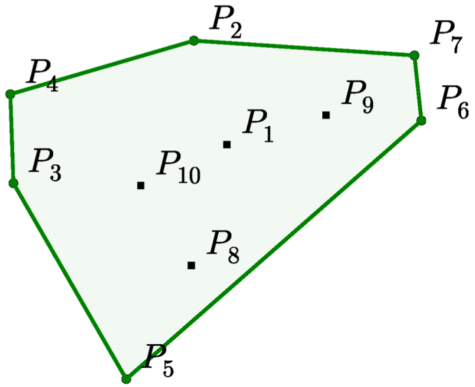
(e) Ptr-Net , m=50, n=50

(f) Ptr-Net , m=5-20, n=20

Convex Hull

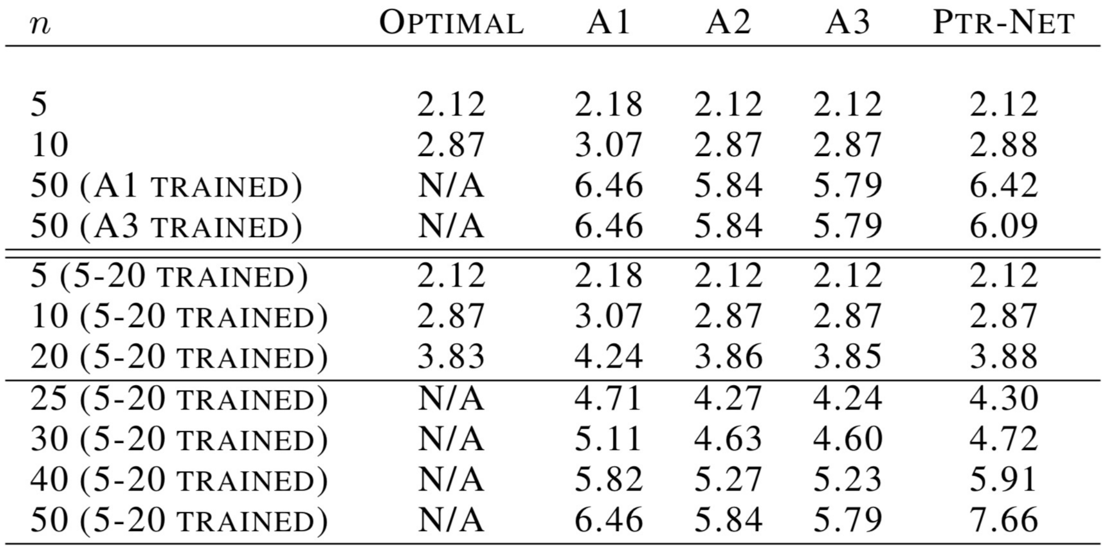


Method	trained n	n	Accuracy	Area
LSTM [1]	50	50	1.9%	FAIL
+attention [5]	50	50	38.9%	99.7%
Ptr-Net	50	50	72.6%	99.9%
LSTM [1]	5	5	87.7%	99.6%
Ptr-Net	5-50	5	92.0%	99.6%
LSTM [1]	10	10	29.9%	FAIL
Ptr-Net	5-50	10	87.0%	99.8%
Ptr-Net	5-50	50	69.6%	99.9%
Ptr-Net	5-50	100	50.3%	99.9%
Ptr-Net	5-50	200	22.1%	99.9%
Ptr-Net	5-50	500	1.3%	99.2%

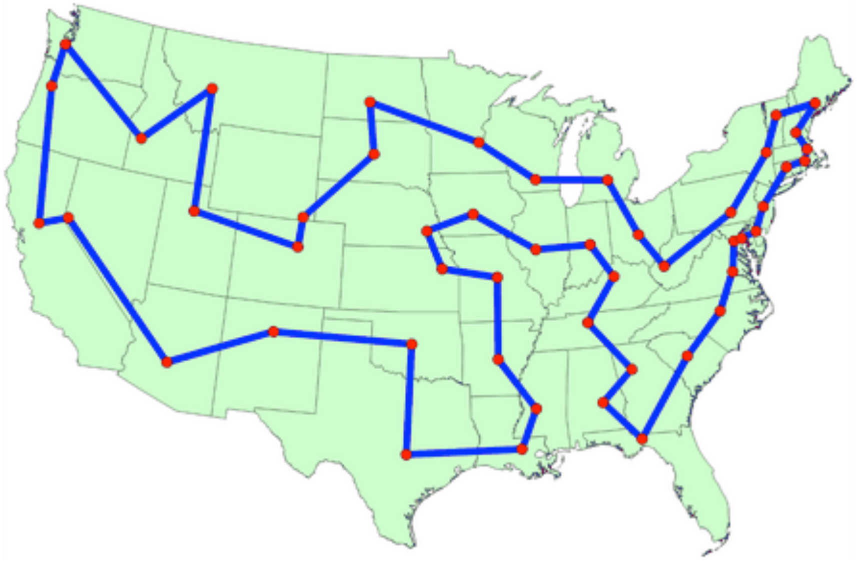


Input $\mathcal{P} = \{P_1, \ldots, P_{10}\}$, and the output sequence $\mathcal{C}^{\mathcal{P}} = \{\Rightarrow, 2, 4, 3, 5, 6, 7, 2, \Leftarrow\}$ representing its convex hull.

Traveling Salesman Problem



n	OPTIMAL	A1	A2	A3	PTR-NET
5	2.12	2.18	2.12	2.12	2.12
10	2.87	3.07	2.87	2.87	2.88
50 (A1 TRAINED)	N/A	6.46	5.84	5.79	6.42
50 (A3 TRAINED)	N/A	6.46	5.84	5.79	6.09
5 (5-20 TRAINED)	2.12	2.18	2.12	2.12	2.12
10 (5-20 TRAINED)	2.87	3.07	2.87	2.87	2.87
20 (5-20 TRAINED)	3.83	4.24	3.86	3.85	3.88
25 (5-20 TRAINED)	N/A	4.71	4.27	4.24	4.30
30 (5-20 TRAINED)	N/A	5.11	4.63	4.60	4.72
40 (5-20 TRAINED)	N/A	5.82	5.27	5.23	5.91
50 (5-20 TRAINED)	N/A	6.46	5.84	5.79	7.66



Thanks for your Attention!

References

- Vinyals, Oriol, Meire Fortunato, and Navdeep Jaitly. "Pointer networks." (2015)
- Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." (2014)
- Vaswani, Ashish, et al. "Attention is all you need." (2017)
- Wang, Shuohang, and Jing Jiang. "Machine comprehension using match-lstm and answer pointer." (2016)

- Some of the slides in this presentation were adapted from the presentation "Pointer Networks And their uses" by Priyansh Trivedi. SDA Reading Group