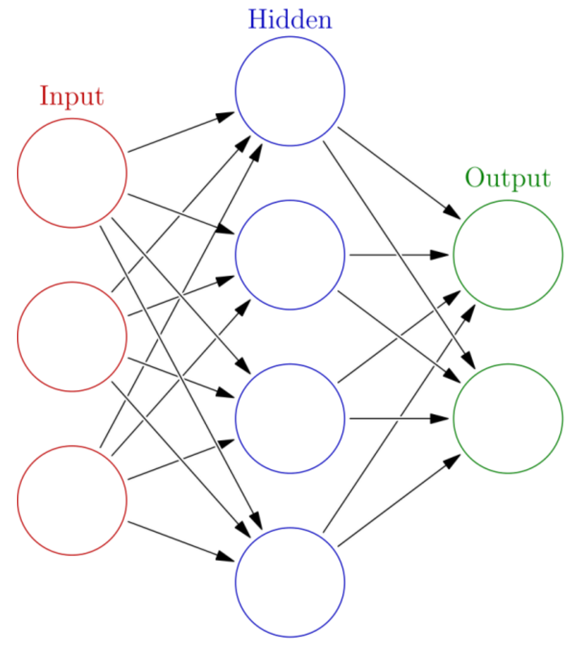
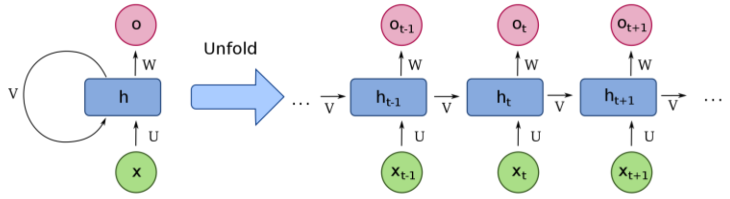
Pointer Networks

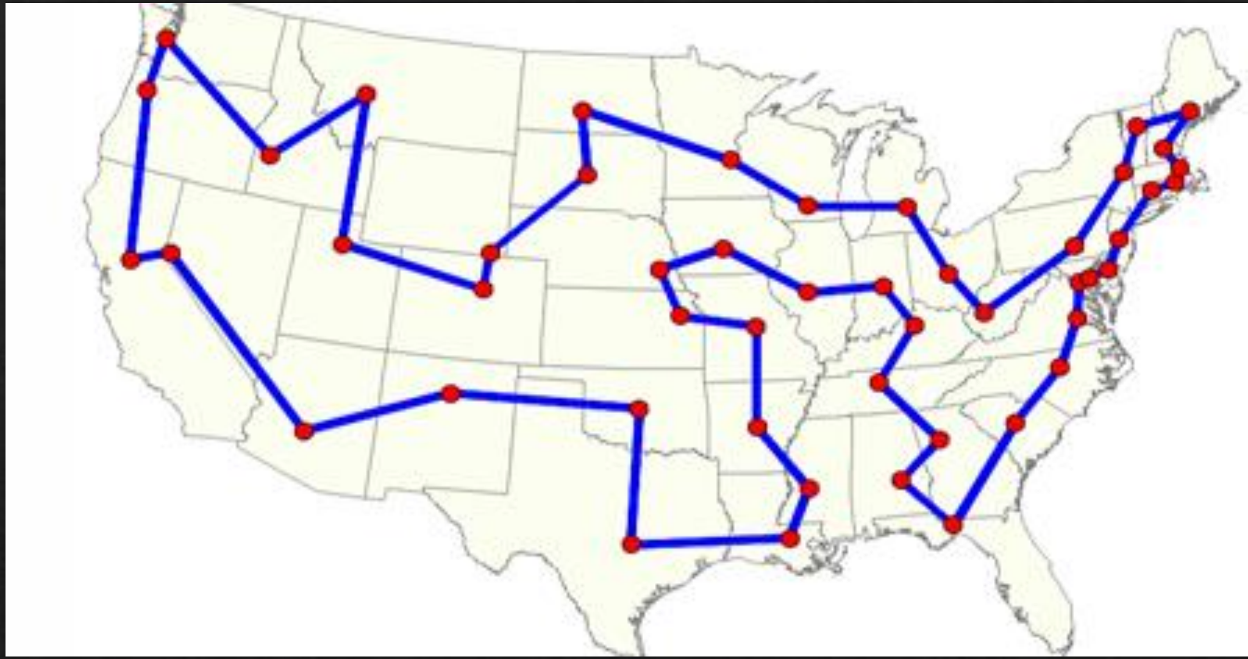
Michał Filipiuk,
CORE #8, 06.11.2020

Why do we need a new kind of neural networks?





Current architectures don’t fit all problems



For example they don't fit combinatorial optimization problems like travelling salesman problem

Pointer Networks

Oriol Vinyals* Google Brain

Meire Fortunato* Department of Mathematics, UC Berkeley

Navdeep Jaitly Google Brain

Sequence-to-sequence approach

Input – sequence of vectors

Network

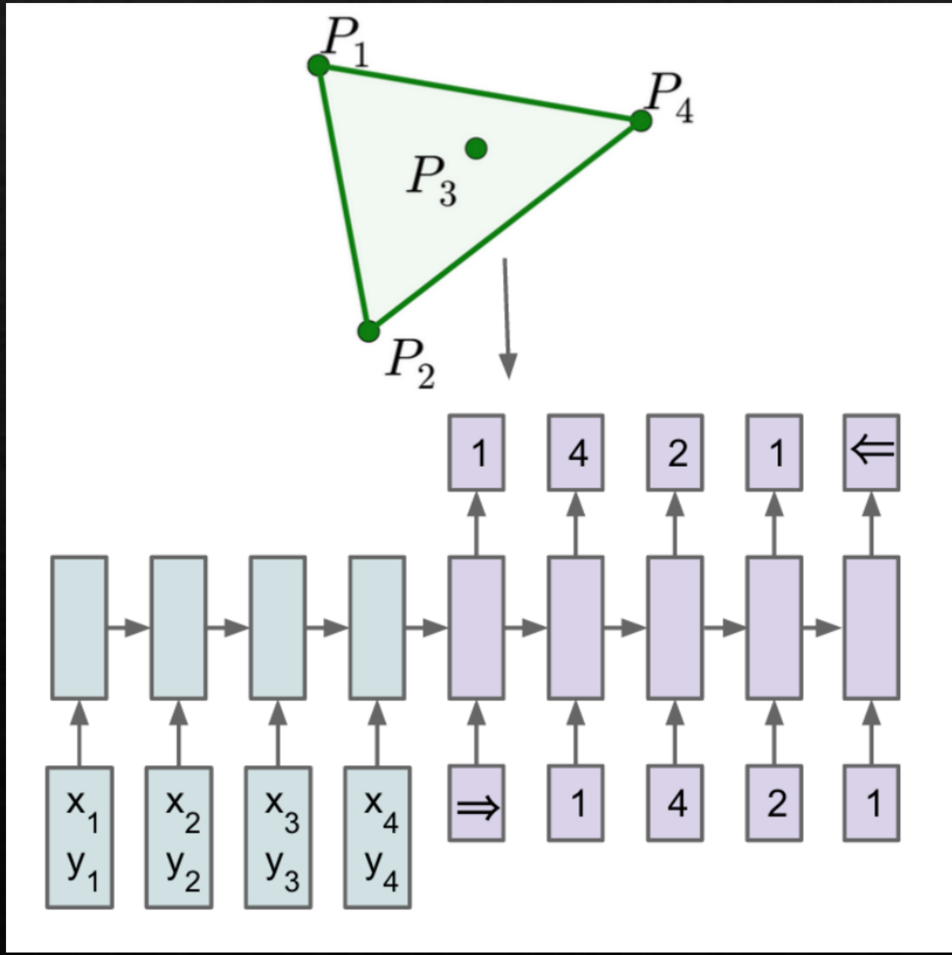
Length of output

Probability of getting C_i

$$p(\mathcal{C}^{\mathcal{P}}|\mathcal{P};\theta) = \prod_{i=1}^{m(\mathcal{P})} p_\theta(C_i|\underbrace{C_1,\ldots,C_{i-1}}_{\text{Already chosen indices}},\mathcal{P};\theta)$$

Output – sequence of indices

$$\theta^* = \arg\max_{\theta} \sum_{\mathcal{P},\mathcal{C}^{\mathcal{P}}} \log p(\mathcal{C}^{\mathcal{P}}|\mathcal{P};\theta)$$



Content Based Input Attention

Learnable parameters

Hidden state of decoder

$$u_j^i = v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n)$$

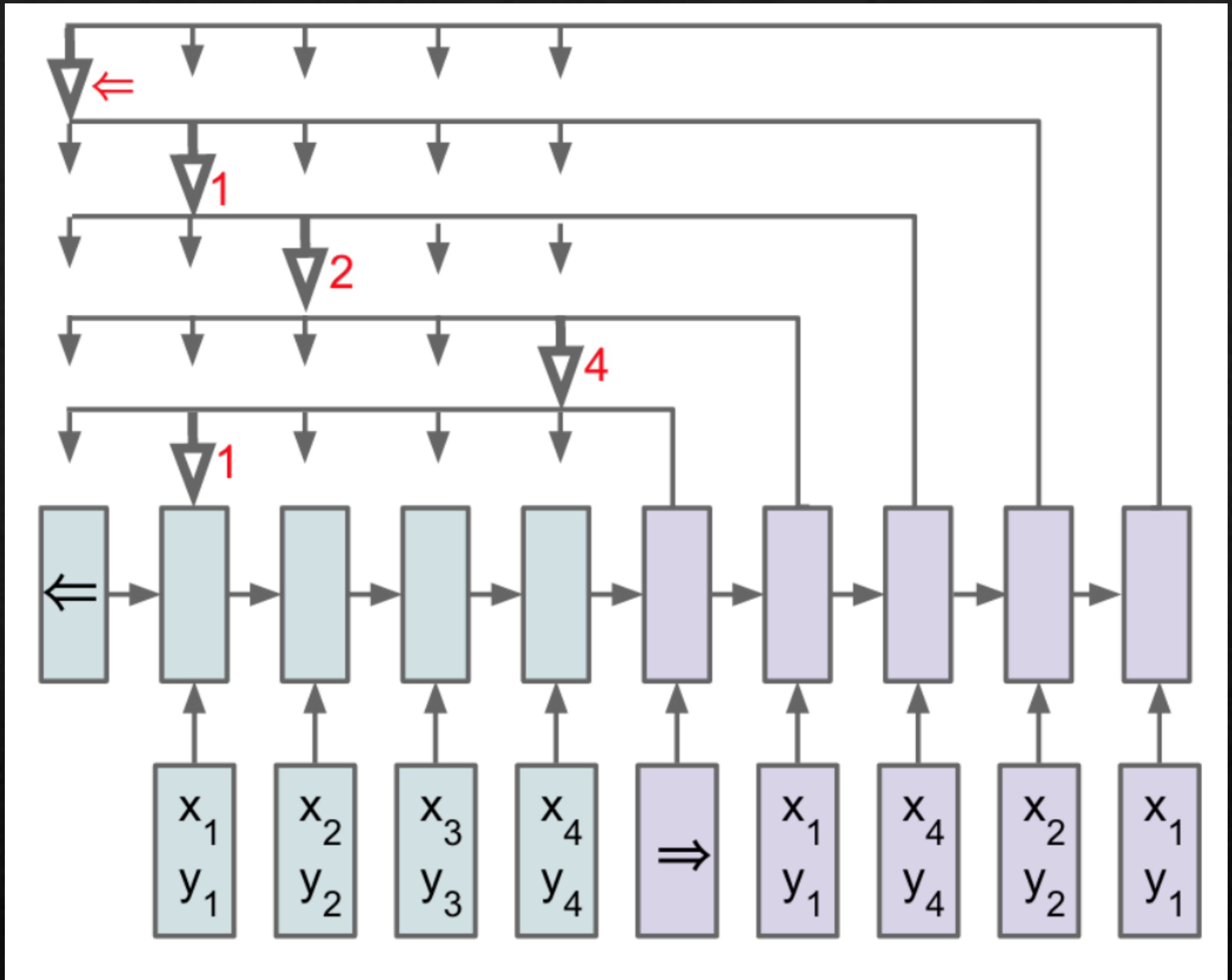
$$a_j^i = \mathrm{softmax}(u_j^i) \quad j \in (1, \ldots, n)$$

Hidden states of encoder

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

Weighted average

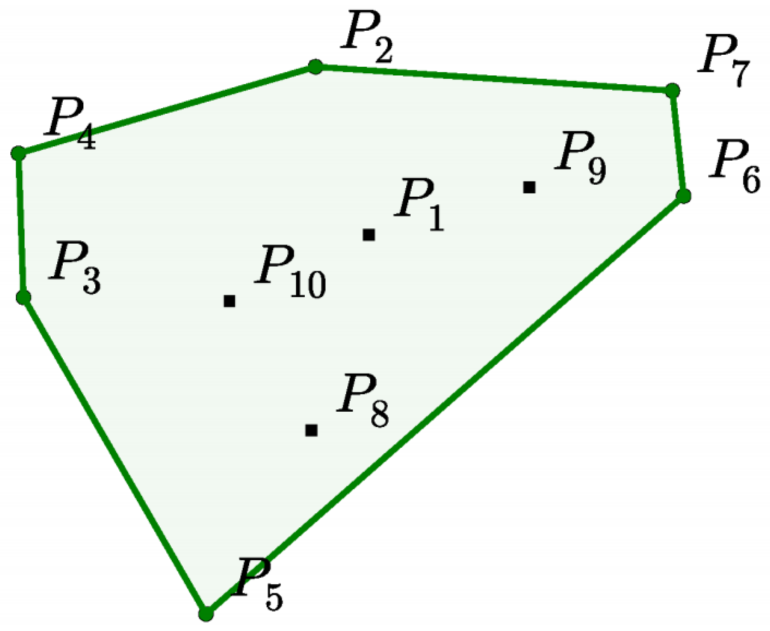
Pointer Network



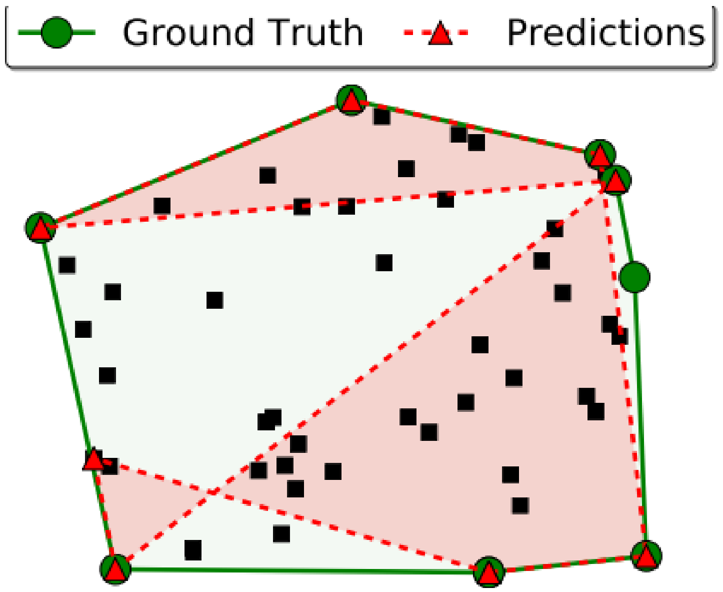
$$
\begin{aligned}
u_j^i &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n) \\
p(C_i | C_1, \ldots, C_{i-1}, \mathcal{P}) &= \mathrm{softmax}(u^i)
\end{aligned}
$$

Problems

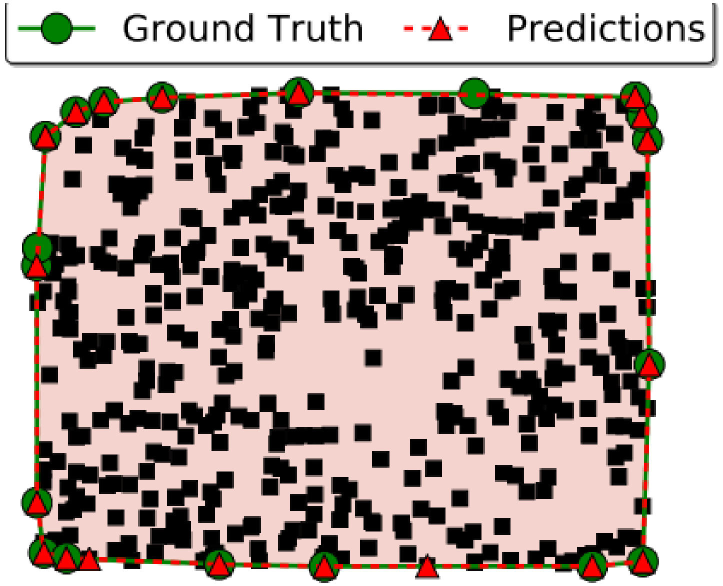
Convex Hull



(a) 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.



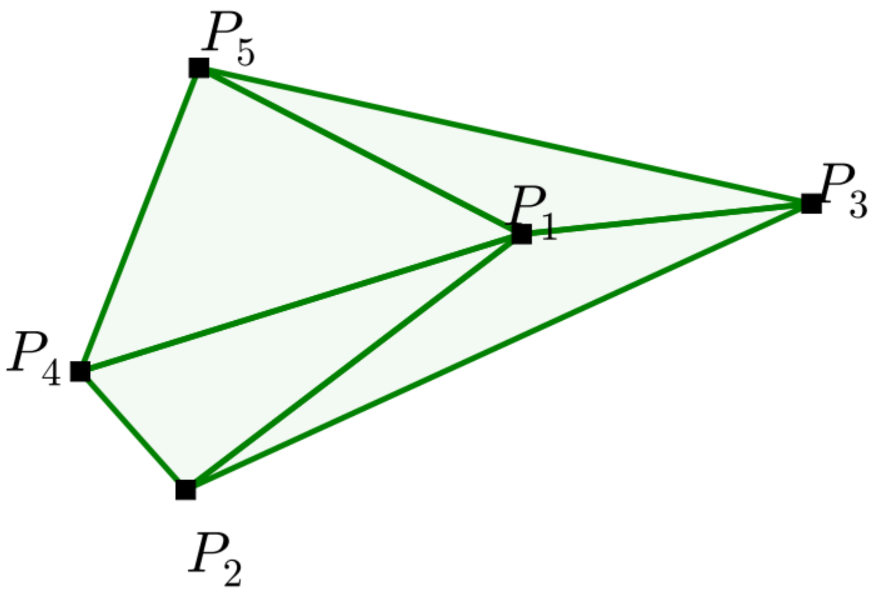
(a) LSTM, m=50, n=50



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%

- Accuracy: Number of test cases, where predicted sequence of points represent the convex hull
- Area: Ratio of area of predicted hull to the ground truth hull

Delaunay Triangulation

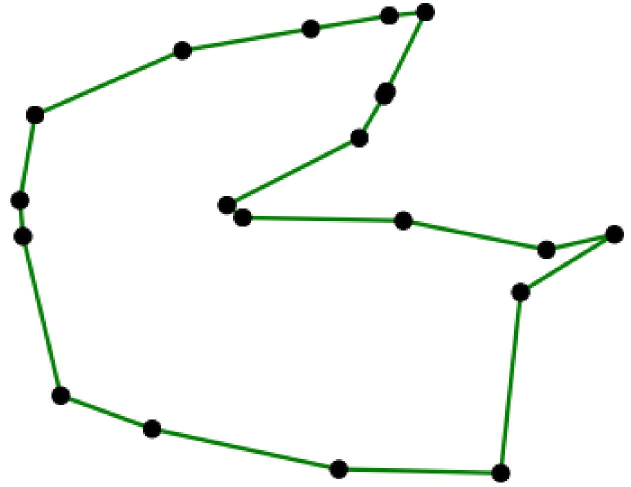


(b) Input $\mathcal{P} = \{P_1, \ldots, P_5\}$, and the output $\mathcal{C}^{\mathcal{P}} = \{\Rightarrow, (1,2,4), (1,4,5), (1,3,5), (1,2,3), \Leftarrow\}$ representing its Delaunay Triangulation.

Travelling Salesman Problem

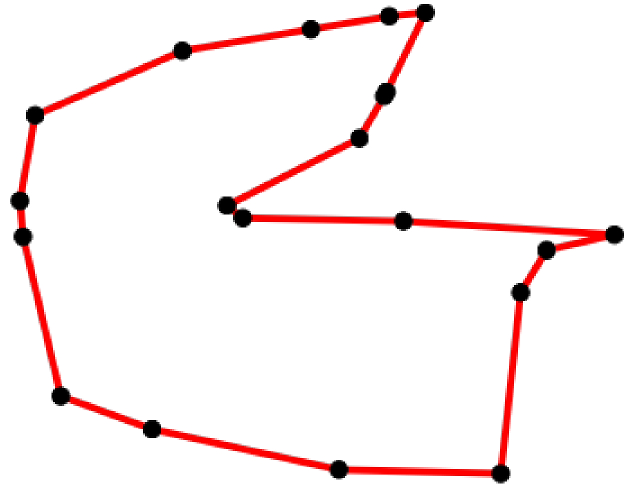
	2^N*N^2	N^2	N^2	N^3	
n	OPTIMAL	A1	A2	A3	PTR-NET
5	2.12	2.18	2.12	2.12	2.12
10	2.87	3.07	2.87	2.87	2.88
50 (A1 TRAINED)	N/A	6.46	5.84	5.79	6.42
50 (A3 TRAINED)	N/A	6.46	5.84	5.79	6.09
5 (5-20 TRAINED)	2.12	2.18	2.12	2.12	2.12
10 (5-20 TRAINED)	2.87	3.07	2.87	2.87	2.87
20 (5-20 TRAINED)	3.83	4.24	3.86	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

Ground Truth: tour length is 3.518



(c) Truth, n=20

Predictions: tour length is 3.523



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

Bibliography

- https://arxiv.org/pdf/1506.03134.pdf
- https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks
- https://en.wikipedia.org/wiki/Delaunay_triangulation
- https://en.wikipedia.org/wiki/Convex_hull
- https://en.wikipedia.org/wiki/Travelling_salesman_problem

ML in PL is back!

https://forms.gle/EuYGXQnezBARTpXN6