MS&E 236 / CS 225: Lecture 2
TSP and RNNs

April 2, 2024

1 Plan for today

- Traveling salesman problem (TSP)
- ML models for TSP
  - Sequence-to-sequence recurrent neural networks (RNNs)
  - Long-short-term-memories (LSTMs)

2 Traveling salesman problem (TSP)

- Input: Network with n nodes represents a map with n cities
  - c_{ij} = distance from node i to j
- Goal: find the shortest distance tour passing through each node exactly once
- How many tours are there? Say we start at node 1
  - $n-1$ choices for the next node, $n-2$ choices for the node after that, ...
  - $(n-1)!$ tours
- One of the most famous NP-hard problems
  - In theory, can't find an optimal tour much faster than trying out all $(n-1)!$ tours
  - Major challenge problem in computer science and optimization for 70+ years
- Many heuristics for this problem (don't provably return optimal solution)
  - Hand-designed, based on human intuition. Can a ML model do better?

3 Sequence-to-sequence recurrent neural networks (RNNs)

- Inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^{d_0}$
  - E.g., word embeddings, cities on a map with $d_0 = 2, \ldots$

- Outputs $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m \in \mathbb{R}^{d_1}$
  - E.g., translation in a foreign language, tour of the cities, ...
- First step: encoder RNN

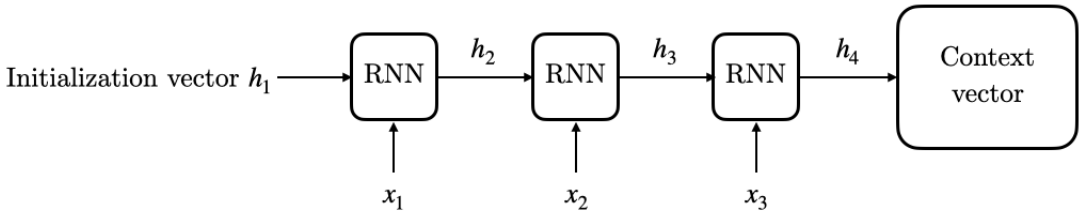

Figure 1: Encoder RNN

  - Compute hidden states $\boldsymbol{h}_t \in \mathbb{R}^{d_h}$, where $\boldsymbol{h}_t = \phi(W^{(hh)}\boldsymbol{h}_{t-1} + W^{(hx)}\boldsymbol{x}_{t-1})$
    * ϕ is a non-linearity, $W^{(hh)} \in \mathbb{R}^{d_h \times d_h}, W^{(hx)} \in \mathbb{R}^{d_h \times d_0}$ are weight matrices
- Second step: decoder RNN

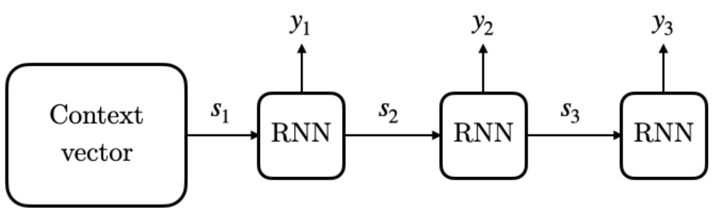

Figure 2: Decoder RNN

  - Compute outputs $\boldsymbol{y}_t = \mathrm{softmax}(W^{(sy)}\boldsymbol{s}_t)$
  - Compute hidden states $\boldsymbol{s}_t$, where $\boldsymbol{s}_t = \phi(W^{(ss)}\boldsymbol{s}_{t-1})$
- Pros: can transform sequence of arbitrary length to sequence of arbitrary length
- Cons:
  - Motivation for long-short-term-memories (LSTMs):
    * Information from early in the sequence lost later in the sequence
    * Issues with exploding and vanishing gradients (see, e.g., CS244N)
  - Motivation for pointer networks for discrete optimization:
    * No reason output will preserve combinatorial structure

3.1 LSTMs

- Key idea: cell state $\boldsymbol{c}_t$. Like a conveyor belt that runs down entire chain
  - Very easy for information to flow along it with minimal change
- Gates regulate how much information is added to or removed from cell state
  - E.g., if cell state includes gender of previous subject, forget that if new subject

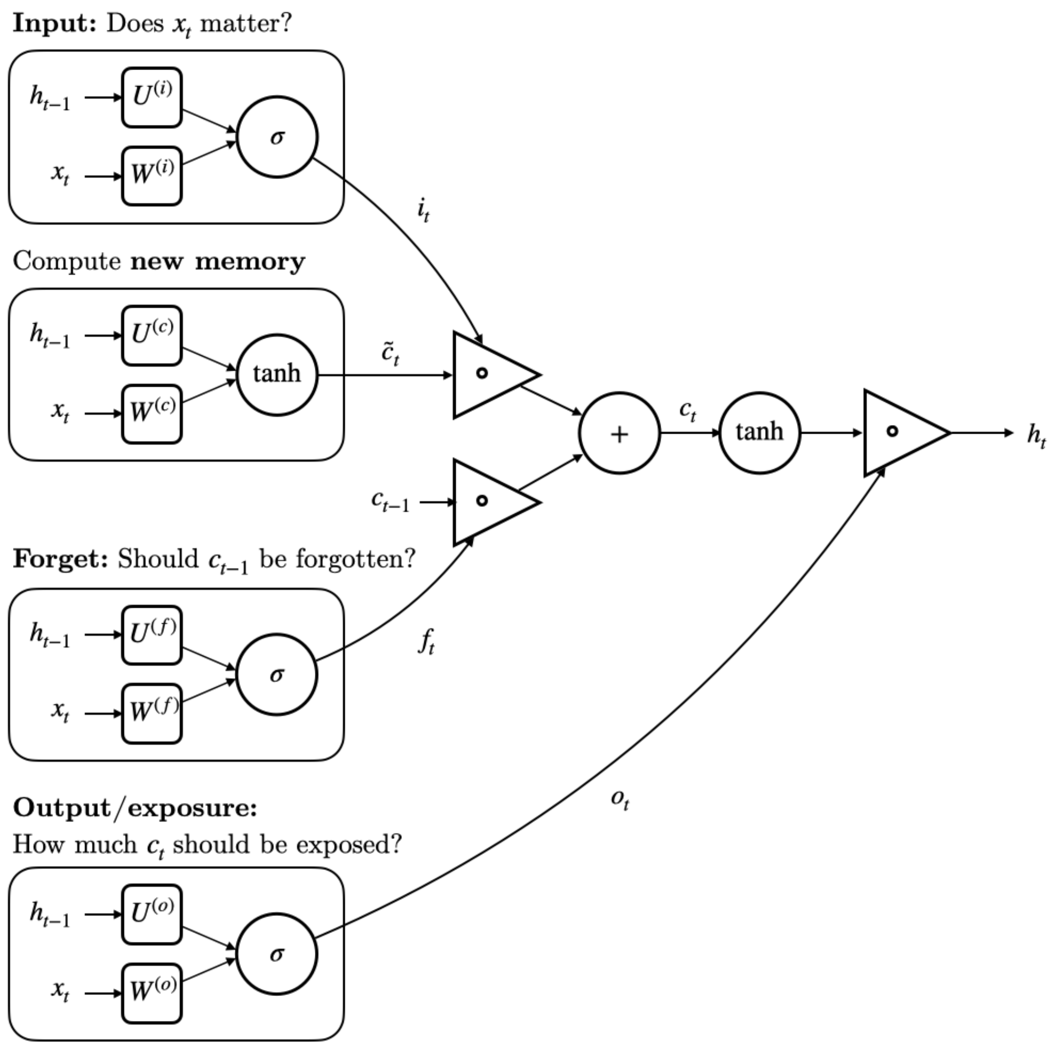

Figure 3: LSTM

3.2 Attention

- Key insight: Decoder RNN only uses single context vector to produce output. However,
  - Different parts of input have different levels of significance
  - Different parts of output may have varying dependence on particular parts of input
- Attention mechanism: Output $\boldsymbol{y}_t$ will depend on $\boldsymbol{s}_t$ and entire sequence $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{n+1}$
  - We'll see an example specific to pointer networks

4 Next time

- Pointer networks: use LSTMs to predict *permutations*
  - E.g., order of nodes in TSP tour