Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks

by Kai Sheng Tai, Richard Socher, Christopher D. Manning

Daniel Perez
tuvistavie

CTO @ Claude Tech
M2 @ The University of Tokyo

October 2, 2017

Distributed representation of words

Idea

Encode each word using a vector in $\mathbb{R}^d$, such that words with similar meanings are close in the vector space.

Representing sentences

Limitation

Good representation of words is not enough to represent sentences

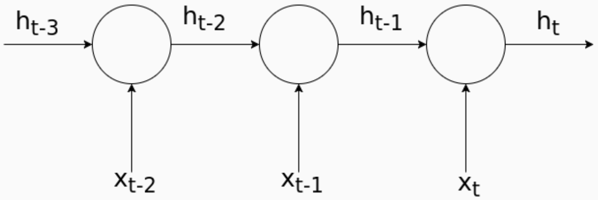
The man driving the aircraft is speaking.

vs

The pilot is making an announce.

Idea

Add state to the neural network by reusing the last output as an input of the model

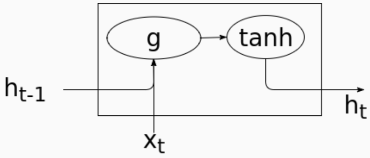


Basic RNN cell

In a plain RNN, h_t is computed as follow

$$h_t = \tanh(Wx_t + Uh_{t-1} + b)$$

given, $g(x_t, h_{t-1}) = Wx_t + Uh_{t-1} + b$,

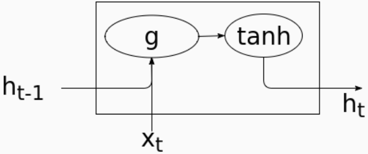


Basic RNN cell

In a plain RNN, h_t is computed as follow

$$h_t = \tanh(Wx_t + Uh_{t-1} + b)$$

given, $g(x_t, h_{t-1}) = Wx_t + Uh_{t-1} + b$,



Issue

Because of vanishing gradients, gradients do not propagate well through the network: impossible to learn long-term dependencies

Long short-term memory (LSTM)

Goal

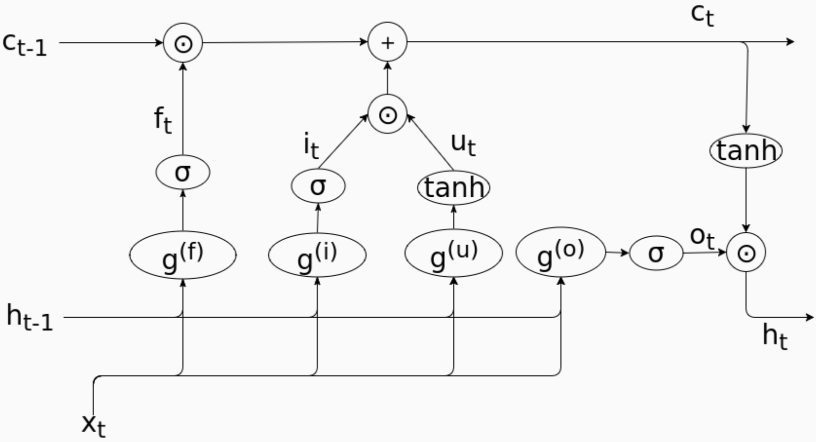
Improve RNN architecture to learn long term dependencies

Main ideas

- Add a memory cell which does not suffer vanishing gradient
- Use gating to control how information propagates

LSTM cell

Given $g^n(x_t, h_{t-1}) = W^{(n)}x_t + U^{(n)}h_{t-1} + b^{(n)}$



Structure of sentences

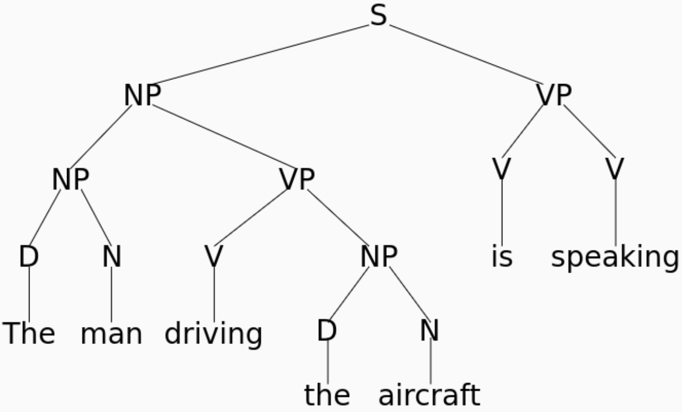
Sentences are not a simple linear sequence.

The man driving the aircraft is speaking.

Structure of sentences

Sentences are not a simple linear sequence.

The man driving the aircraft is speaking.

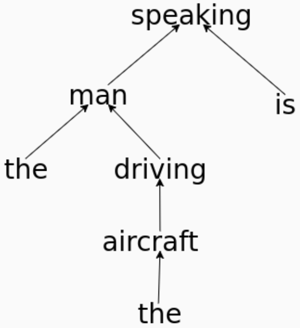


Constituency tree

Structure of sentences

Sentences are not a simple linear sequence.

The man driving the aircraft is speaking.



Dependency tree

Tree-structured LSTMs

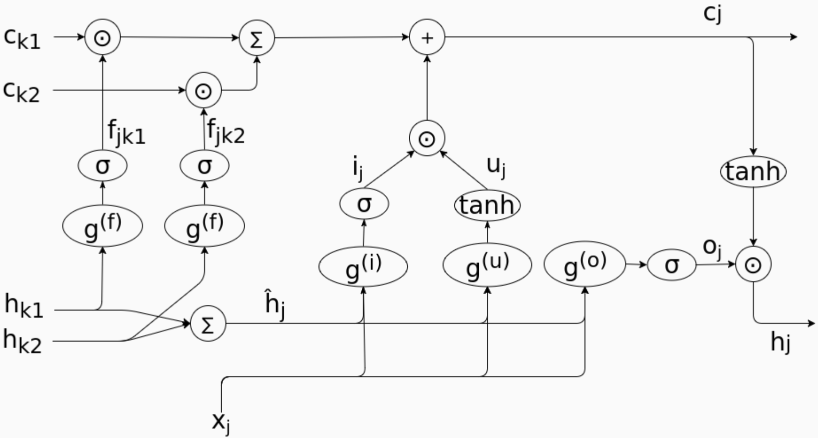
Goal

Improve encoding of sentences by using their structures

Models

- Child-sum tree LSTM
  Sums over all the children of a node: can be used for any number of children
- N-ary tree LSTM
  Use different parameters for each node: better granularity, but maximum number of children per node must be fixed

Children outputs and memory cells are summed



Child-sum tree LSTM at node j with children k_1 and k_2

Child-sum tree LSTM

Properties

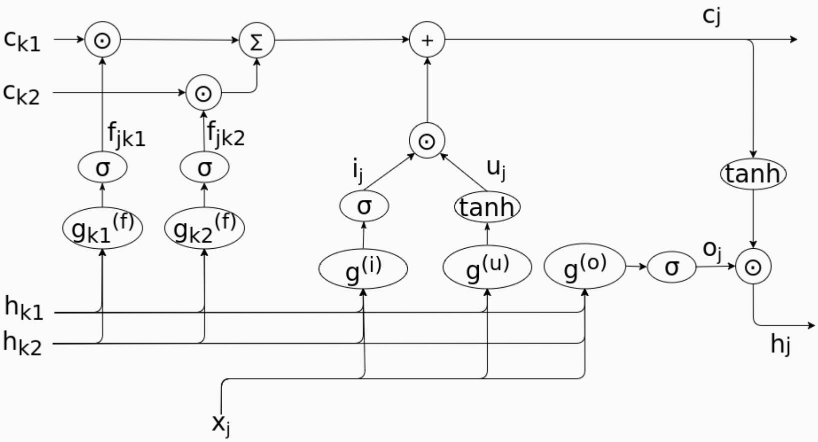
- Does not take into account children order
- Works with variable number of children
- Shares gates weight (including forget gate) between children

Application

Dependency Tree-LSTM: number of dependents is variable

N-ary tree LSTM

Given $g_k^{(n)}(x_t, h_{l_1}, \cdots, h_{l_N}) = W^{(n)}x_t + \sum_{l=1}^{N} U_{kl}^{(n)} h_{jl} + b^{(n)}$



Binary tree LSTM at node j with children k_1 and k_2

N-ary tree LSTM

Properties

- Each node must have at most N children
- Fine-grained control on how information propagates
- Forget gate can be parameterized so that siblings affect each other

Application

Constituency Tree-LSTM: using a binary tree LSTM

Sentiment classification

Task

Predict sentiment $\hat{y}_j$ of node j

Sub-tasks

- Binary classification
- Fine-grained classification over 5 classes

Method

- Annotation at node level
- Uses negative log-likelihood error

$$\hat{p}_\theta(y|\{x\}_j) = \text{softmax}\left(W^{(s)} h_j + b^{(s)}\right)$$

$$\hat{y}_j = \arg\max_y \hat{p}_\theta(y|\{x\}_j)$$

Sentiment classification results

Constituency Tree-LSTM performs best on fine-grained sub-task

Method	**Fine-grained**	**Binary**
CNN-multichannel	47.4	**88.1**
LSTM	46.4	84.9
Bidirectional LSTM	49.1	87.5
2-layer Bidirectional LSTM	48.5	87.2
Dependency Tree-LSTM	48.4	85.7
Constituency Tree-LSTM		
- randomly initialized vectors	43.9	82.0
- Glove vectors, fixed	49.7	87.5
- Glove vectors, tuned	**51.0**	88.0

Semantic relatedness

Task

Predict similarity score in $[1, K]$ between two sentences

Method

Similarity between sentences L and R annotated with score $\in [1, 5]$

- Produce representations h_L and h_R
- Compute distance h_+ and angle $h_\times$ between h_L and h_R
- Compute score using fully connected NN

$$h_s = \sigma\left(W^{(\times)} h_\times + W^{(+)} h_+ + b^{(h)}\right)$$
$$\hat{p}_\theta = \text{softmax}\left(W^{(p)} h_s + b^{(p)}\right)$$
$$\hat{y} = r^T \hat{p}_\theta \qquad\qquad r = [1, 2, 3, 4, 5]$$

- Error is computed using KL-divergence

Semantic relatedness results

Dependency Tree-LSTM performs best for all measures

Method	**Pearson's** r	**MSE**
LSTM	0.8528	0.2831
Bidirectional LSTM	0.8567	0.2736
2-layer Bidirectional LSTM	0.8558	0.2762
Constituency Tree-LSTM	0.8582	0.2734
Dependency Tree-LSTM	**0.8676**	**0.2532**

Summary

- Tree-LSTMs allow to encode tree topologies
- Can be used to encode sentences parse trees
- Can capture longer and more fine-grained words dependencies

References

Christopher Olah.
Understanding lstm networks, 2015.

Kai Sheng Tai, Richard Socher, and Christopher D Manning.
Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks.
2015.