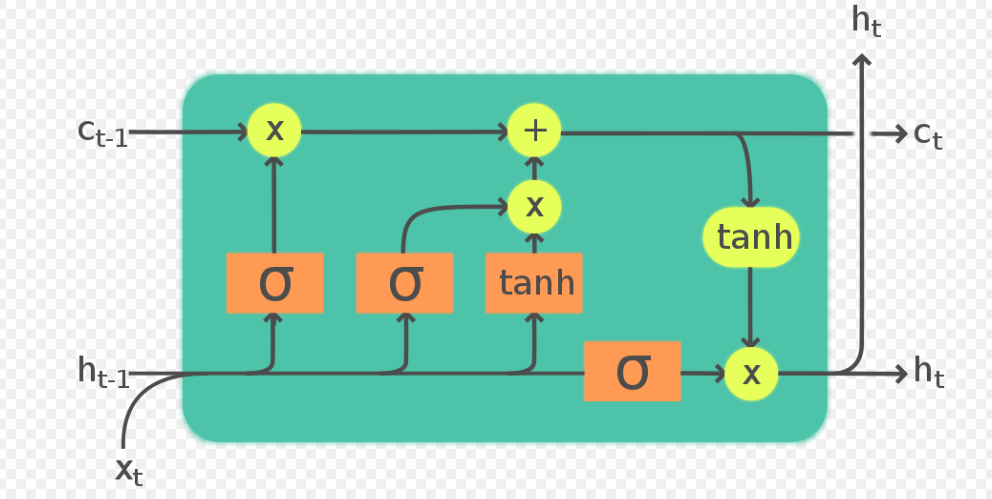
1.

(a)

i.



ii. gradient vanishing

iii.

input through word embedding could become sentence vector

output we use logical sentence form to represent parsing tree, where each phrase could be represented as vector too

(c)

i. each word is represented as a vector, all the vectors have the same length, stack the word vectors and generate a N\*M matrix, where N is the length, M is the vector length

ii.

000

010

000

111

1/9\*111

111

iii.

A pooling function is applied to each map to reduce dimensionality and number of parameters

1-max pooling

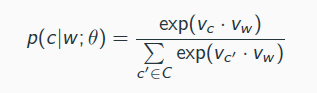
(d)

Gradient descent: calculate the difference between model output and real output, during back propagation apply this bias to update the parameters in the model.

Gradient descent use all the training data to update the parameters, which requires more time; while stochastic gradient descent each time randomly pick one training data to update the parameters, which saves a lot of time.

3.

(a)



Where Vc is the context vector and vw is the word vector

We seek parameter values (i.e., vector representations for both words and contexts) such that the dot product vw \_ vc associated with “good" word-context pairs is maximized.

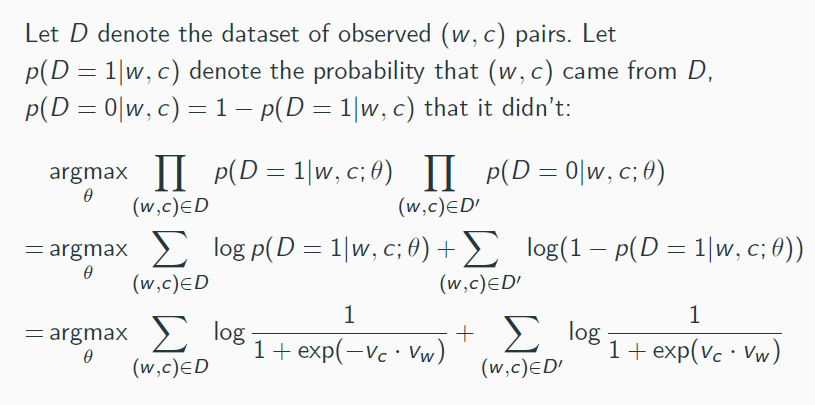
(b)

Because the denominator needs to compute the sum of all pot product between context vectors and word vector, whose size equal to vocabulary size, can easily be in the millions (too many outputs to evaluate)

Negative sampling (not understand)

Instead of computing the full output layer based on the hidden layer, evaluate only the output neuron that represents the positive class and a few randomly sampled neurons.

(c)



(d)

(e)

Go through a fully connected layer

(f)

Encoder-decoder