

Notes:

The first picture has an error which is the weights of RNN should always be the same one, no Wh1h2 etc anymore just one Whh or Wxh or wyh.

Training time:

(1) Use an affine transformation to compute the initial hidden state

from the image features. This should produce an array of shape (N, H)

(2) Use a word embedding layer to transform the words in captions\_in

from indices to vectors, giving an array of shape (N, T, W).

(3) Use either a vanilla RNN or LSTM (depending on self.cell\_type) to

process the sequence of input word vectors and produce hidden state

vectors for all timesteps, producing an array of shape (N, T, H).

(4) Use a (temporal) affine transformation to compute scores over the

vocabulary at every timestep using the hidden states, giving an

array of shape (N, T, V).

(5) Use (temporal) softmax to compute loss using captions\_out, ignoring

the points where the output word is <NULL> using the mask above.

Test time:

(1) Embed the previous word using the learned word embeddings

(2) Make an RNN step using the previous hidden state and the embedded

current word to get the next hidden state.

(3) Apply the learned affine transformation to the next hidden state to

get scores for all words in the vocabulary

(4) Select the word with the highest score as the next word, writing it

to the appropriate slot in the captions variable

LSTM Backward stages: 