

Lecture 5 Q&A

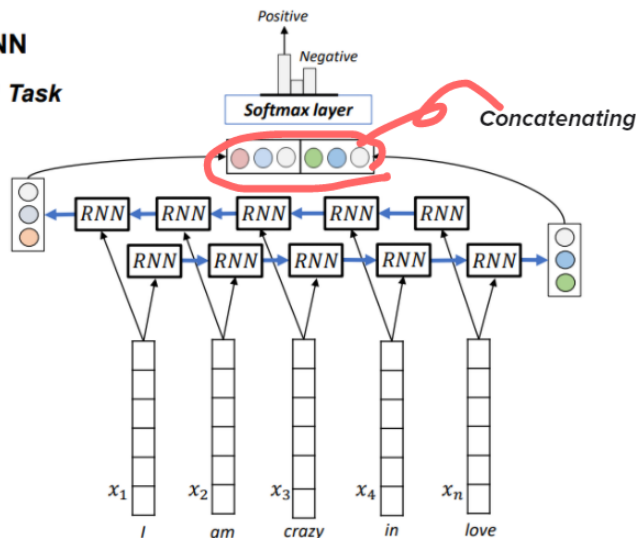
30 March 2021

How can we stack two bi-direction output?

Beware to use the term "Stack". as there is a "Stack RNN or LSTM" when we stack two or more layers on top of each layer.

Bi-RNN

N to 1 Task



Concatenating?

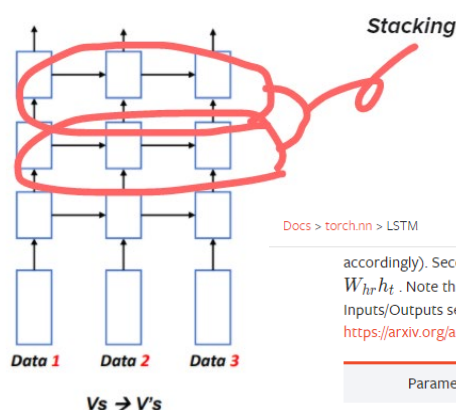
It means stacking end to end rather than on top of one another

For example, concatenate $[x_1, x_2, x_3]$, and $[x_4, x_5, x_6] \Rightarrow [x_1, x_2, x_3, x_4, x_5, x_6]$

4 Data Transformation for Deep Learning NLP



Stacking RNN



Docs > torch.nn > LSTM

Stacking?

Generally stacked LSTM or RNN refers to multiple layers (on top of each layer).

In Pytorch, you can simply define this in your model definition or RNN or LSTM layer has parameter called "num_layers". You can specify how many layers you want to stack by using this "num_layers" parameter.

accordingly). Second, the output hidden state of each layer will be multiplied by a learnable projection matrix: $h_t = W_{hr}h_t$. Note that as a consequence of this, the output of LSTM network will be of different shape as well. See Inputs/Outputs sections below for exact dimensions of all variables. You can find more details in <https://arxiv.org/abs/1402.1128>.

Parameters

- **input_size** – The number of expected features in the input x
- **hidden_size** – The number of features in the hidden state h
- **num_layers** – Number of recurrent layers. E.g., setting `num_layers=2` would mean stacking two LSTMs together to form a *stacked LSTM*, with the second LSTM taking in outputs of the first LSTM and computing the final results. Default: 1
- **bias** – If `False`, then the layer does not use bias weights b_{ih} and b_{hh} . Default: `True`
- **batch_first** – If `True`, then the input and output tensors are provided as (batch, seq, feature). Default: `False`
- **dropout** – If non-zero, introduces a *Dropout* layer on the outputs of each LSTM layer except the last layer, with dropout probability equal to `dropout`. Default: 0
- **bidirectional** – If `True`, becomes a bidirectional LSTM. Default: `False`
- **proj_size** – If > 0 , will use LSTM with projections of corresponding size. Default: 0

[LSTM — PyTorch 1.8.1 documentation](#)

***What if a word has different meaning in different situation?
(lexicons)***

How would you control for changes in language over time? E.g. if you were doing a sentiment analysis of newspaper articles over the last century.

That is very interesting about language. This problem causes that using same embedding for different time period may not be appropriate.

There is a specific word embedding for this purpose, that you will learn later this unit. It is called "ELMO" (Contextual Language Embedding)



Yes, that ELMO. Want to know more? wait for later lecture or read the paper → [\[1802.05365\] Deep contextualized word representations \(arxiv.org\)](#)