ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
CAMPUS DI FORLÌ

# 91258 / B0385
# Natural Language Processing
## Lesson 17. Bidirectional RNN →

### Long Short-Term Memory Networks

Alberto Barrón-Cedeño
a.barron@unibo.it

26/11/2025

## Table of Contents





## Left and right context

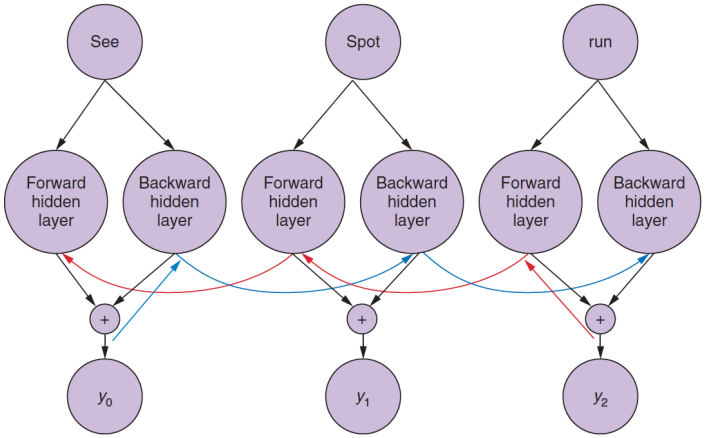
Not only the previous context is important to understand the *current* token

They wanted to pet the dog whose fur was brown.

- Descriptions and relevant information often come later
- A standard RNN neglects information from the *future*



## Bidirectional recurrent neural network




- We *arrange* 2 RNNs:
  - one takes the input as usual
  - the other takes the backward input
  - ⊕ means concatenation

## Implementation difference

```
 # Adding one bidirectional recurrent layer

model.add(Bidirectional(SimpleRNN(
    num_neurons,
    return_sequences=True),
    input_shape=(maxlen, embedding_dims))
    )
```

Let us see



## BiRNN zoom into results

Accuracies after 2 epochs

| units | Acc | $\text{Acc}_{val}$ |
|---|---|---|
| 50 | 0.8156 | 0.7662 |
| 40 | 0.8244 | 0.7540 |
| 30 | 0.8259 | 0.7874 |
| 20 | 0.8072 | 0.8076 |
| 10 | 0.8007 | 0.8016 |
| 5 | 0.7973 | 0.8006 |
| 1 | 0.7070 | 0.7822 |

* Reminder: we had used 50 units last time for the RNN



## LSTMs



## Short effect from the past

The effect of token $x_t$ *dilutes* significantly as soon as in $t + 2$

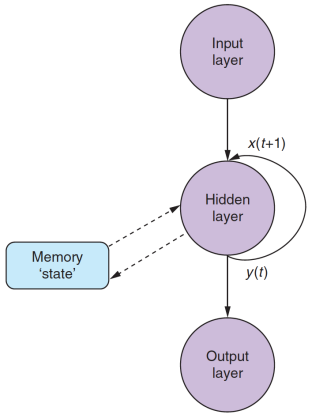
Consider the following —fairly plausible— texts...

The young woman went to the movies with her friends.

The young woman found a free ticket on the ground and went to the movies.

- In both cases, went is the main verb
- A (Bi)RNN would hardly reflect that in the second case
- We need an architecture able to "remember" the entire input

## State: the memory of an LSTM




- The memory state contains attributes
- The attributes are updated with every instance
- The *rules* of the state are trained NNs

Now we have two learning objectives:

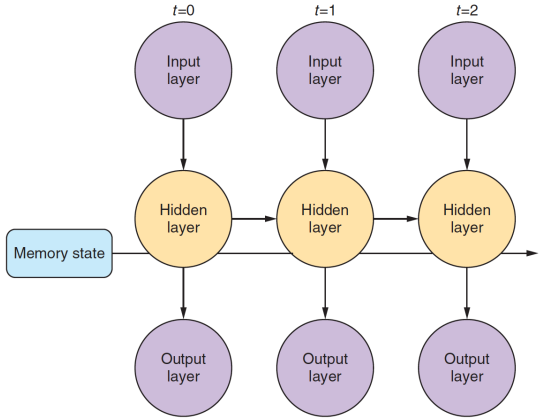
- Learn to predict the target labels
- Learn to identify what has to be *remembered*

(Lane et al., 2019, p. 276)



## Unrolled LSTM

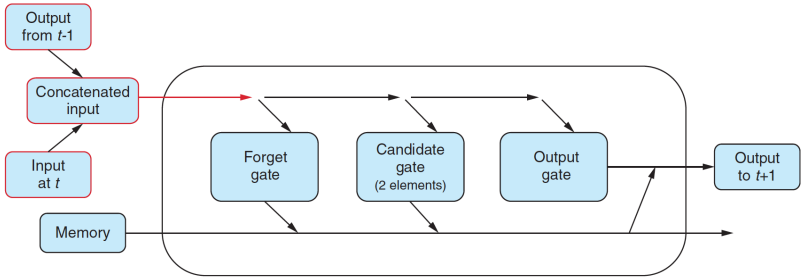
- Activation from $t-1$ plus memory state
- The memory state sends a vector with the state of each LSTM cell, of cardinality number_of_units




(Lane et al., 2019, p. 277)



## The LSTM cell (layer)

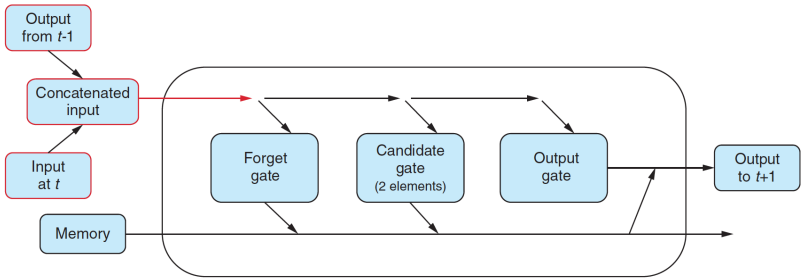



Input: $\text{output}_{t-1} \oplus \text{input}_t$

Gates: a FF layer + an activation function each



## LSTM Forget Gate




Input: $[x_{[t,0]}, x_{[t,1]}, \ldots, x_{[t,299]}, h_{[t-1,0]}, h_{[t-1,1]}, \ldots h_{[t-1,49]}, 1]$

Forget: How much of the memory should be erased —forgetting long-term dependencies as new ones arise

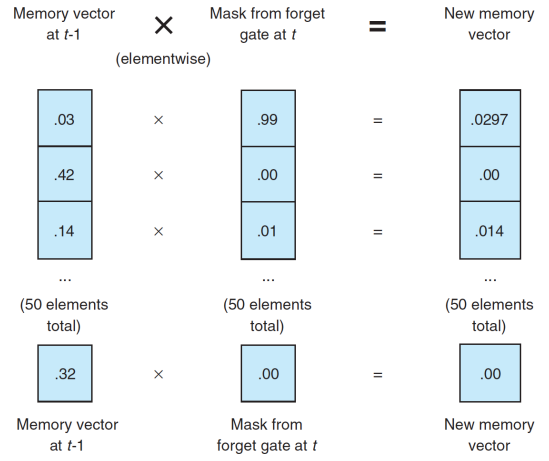
$351 * 50 = 17,550$ parameters

Feed-forward NN with sigmoid activation function: $[0, 1]$

(Lane et al., 2019, p. 280)

# LSTM Forget Gate

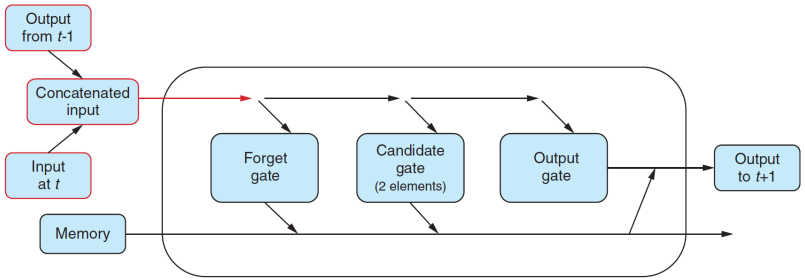
Forget is a mask:




(Lane et al., 2019, p. 282)



# LSTM Candidate Gate

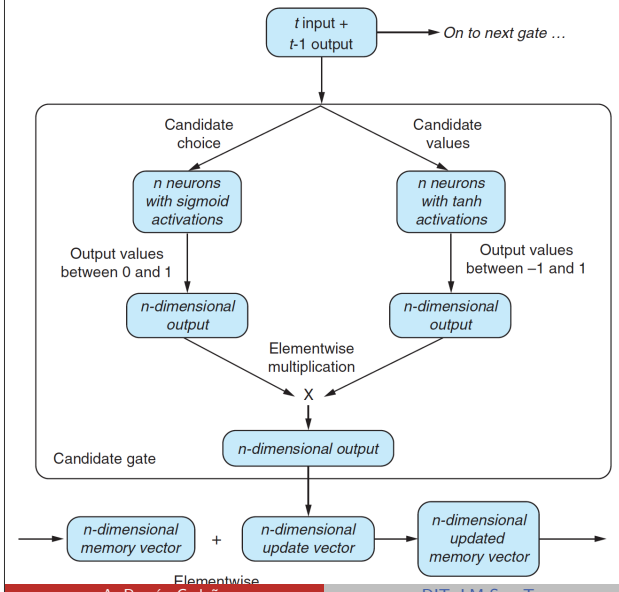



Input: $[x_{[t,0]}, x_{[t,1]}, \ldots, x_{[t,299]}, h_{[t-1,0]}, h_{[t-1,1]}, \ldots h_{[t-1,49]}, 1]$

Candidate: How much to augment the memory —what to remember and where to do it



# LSTM Candidate Gate



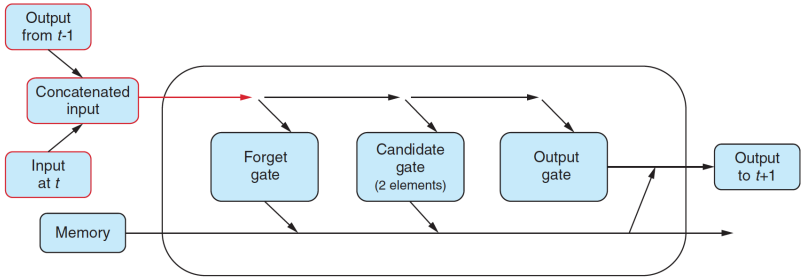

Candidate choice
Which values should be updated (∼forget)

Candidate values
Computes those new values

(Lane et al., 2019, p. 283)



# LSTM Output Gate

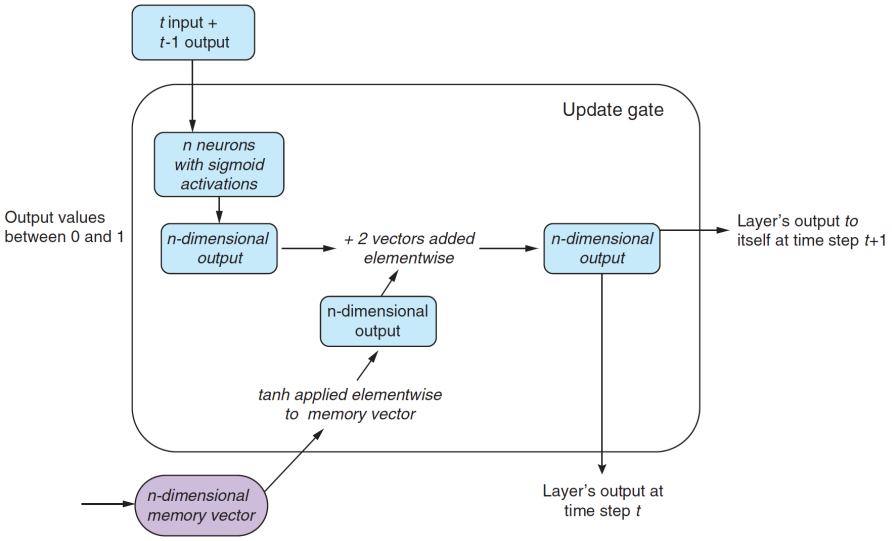



Input: $[x_{[t,0]}, x_{[t,1]}, \ldots, x_{[t,299]}, h_{[t-1,0]}, h_{[t-1,1]}, \ldots h_{[t-1,49]}, 1]$

Output: produces the output vector —both for the actual task and back to the memory

- sigmoid to the input
- tanh to the state

## LSTM Output Gate




* The figure says "added". It is a product

(Lane et al., 2019, p. 284)



## LSTM: Wrapping Up

- The *main* network uses the output of the memory in the same fashion as in a RNN
- The memory *decides* what to keep/feed to the network
- The weights of the memory are also learned by back-propagation

Let us see



## LSTM: Result

| arch | units | Acc | $\text{Acc}_{val}$ |
|---|---|---|---|
| BiRNN | 50 | 0.8156 | 0.7662 |
| BiRNN | 40 | 0.8244 | 0.7540 |
| BiRNN | 30 | 0.8259 | 0.7874 |
| BiRNN | 20 | 0.8072 | 0.8076 |
| BiRNN | 10 | 0.8007 | 0.8016 |
| BiRNN | 5 | 0.7973 | 0.8006 |
| BiRNN | 1 | 0.7070 | 0.7822 |
| LSTM | 50 | 0.8692 | **0.8678** |



## References

Lane, H., C. Howard, and H. Hapkem
2019. *Natural Language Processing in Action*. Shelter Island, NY: Manning Publication Co.