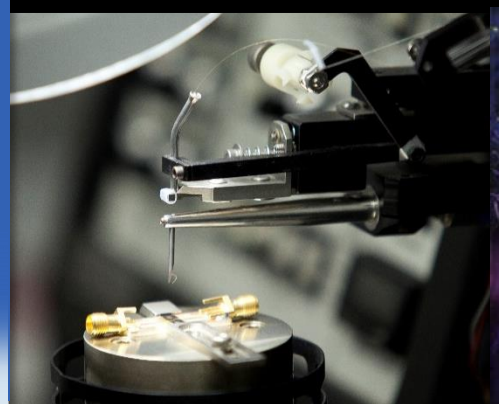
Centro Brasileiro de Pesquisas Físicas

Redes Neurais profundas e aplicações Deep Learning

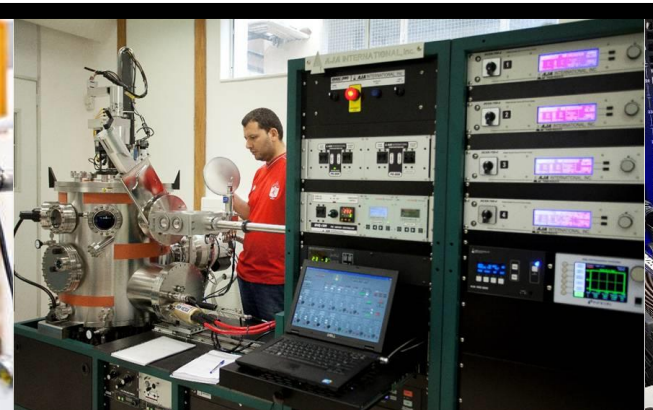
Clécio Roque De Bom – debom@cbpf.br

CBPF

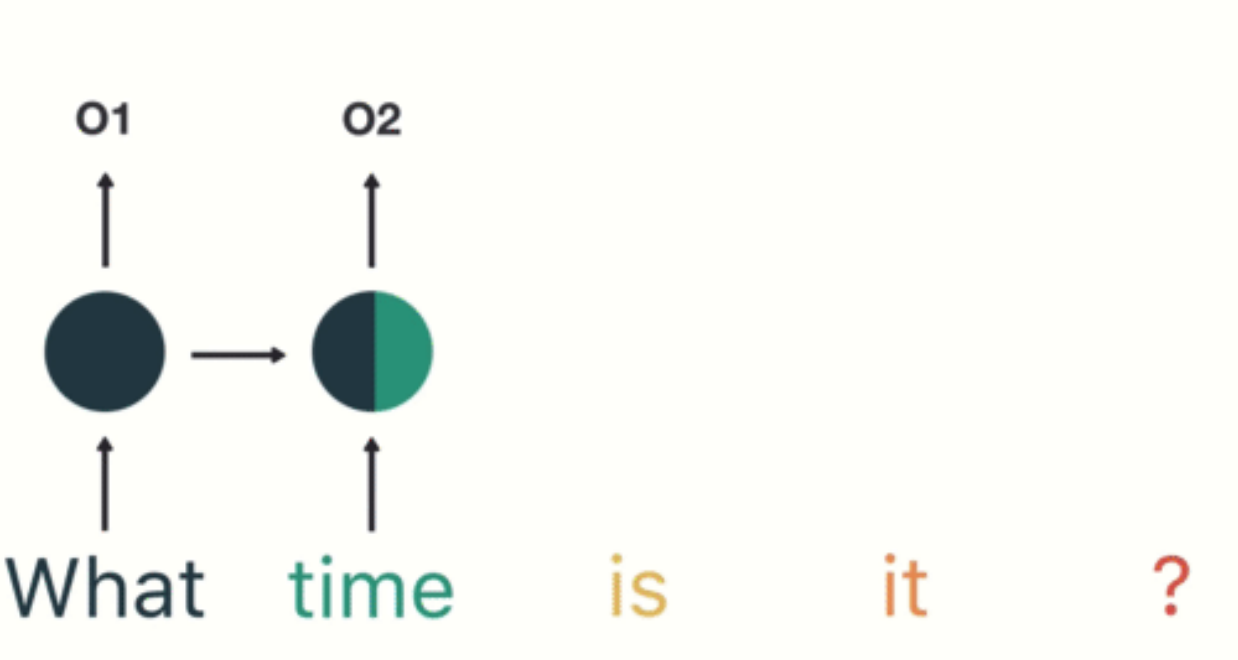
clearnightsrthebest.com

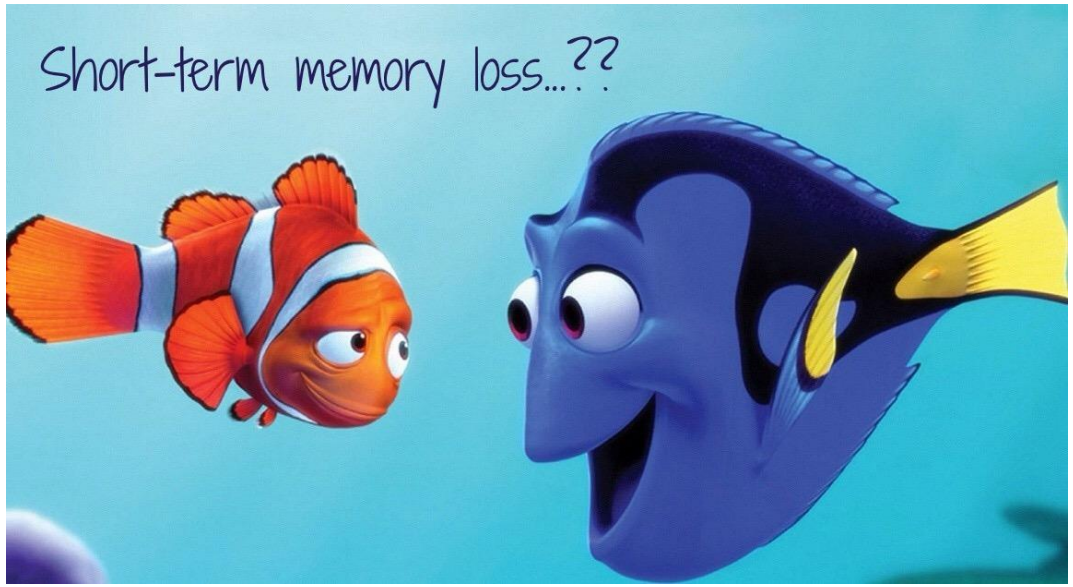




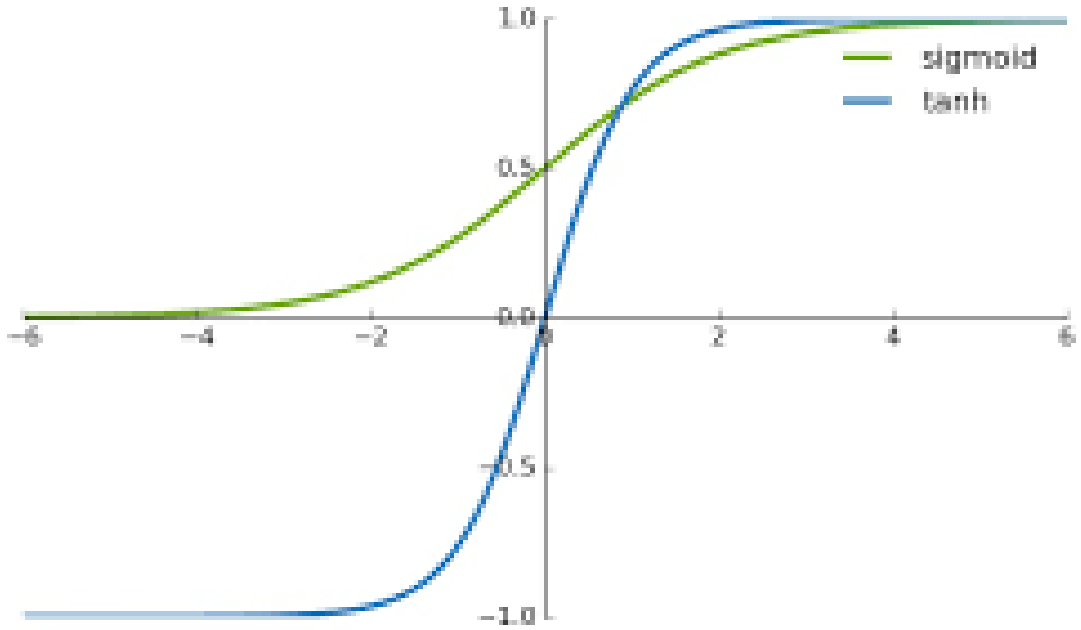



Recurrent Neural Nets

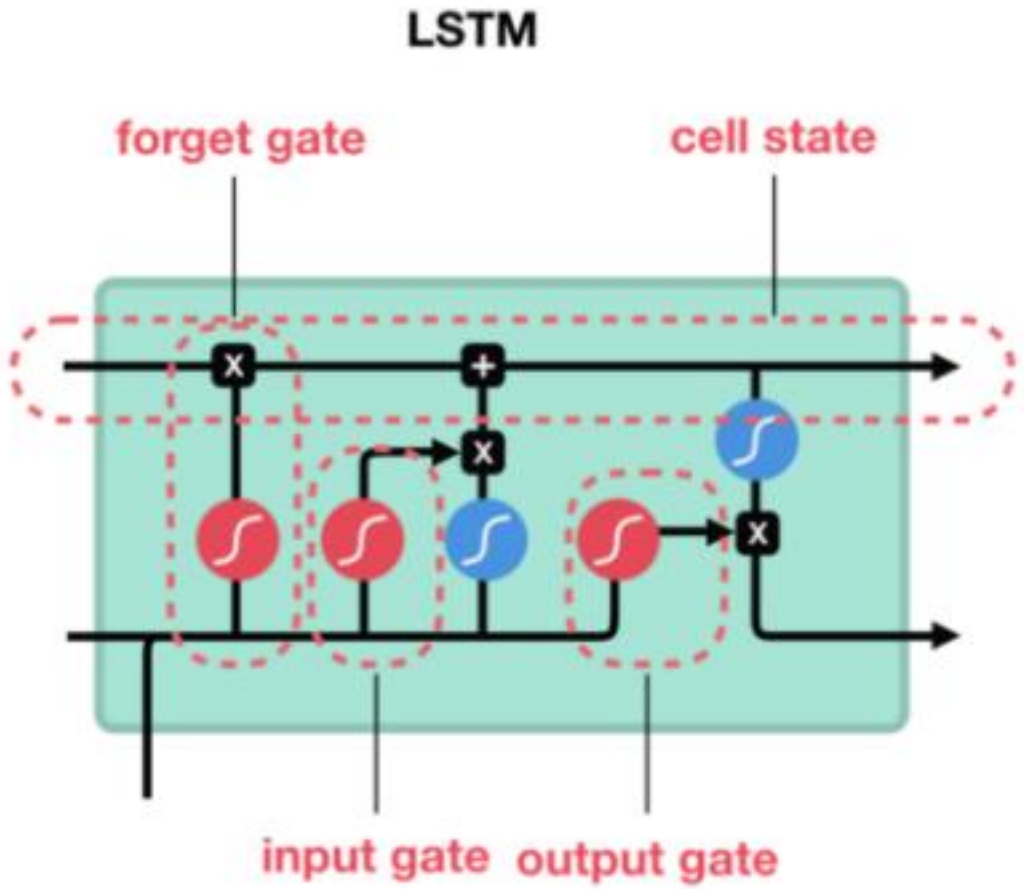




Remembering and forgeting

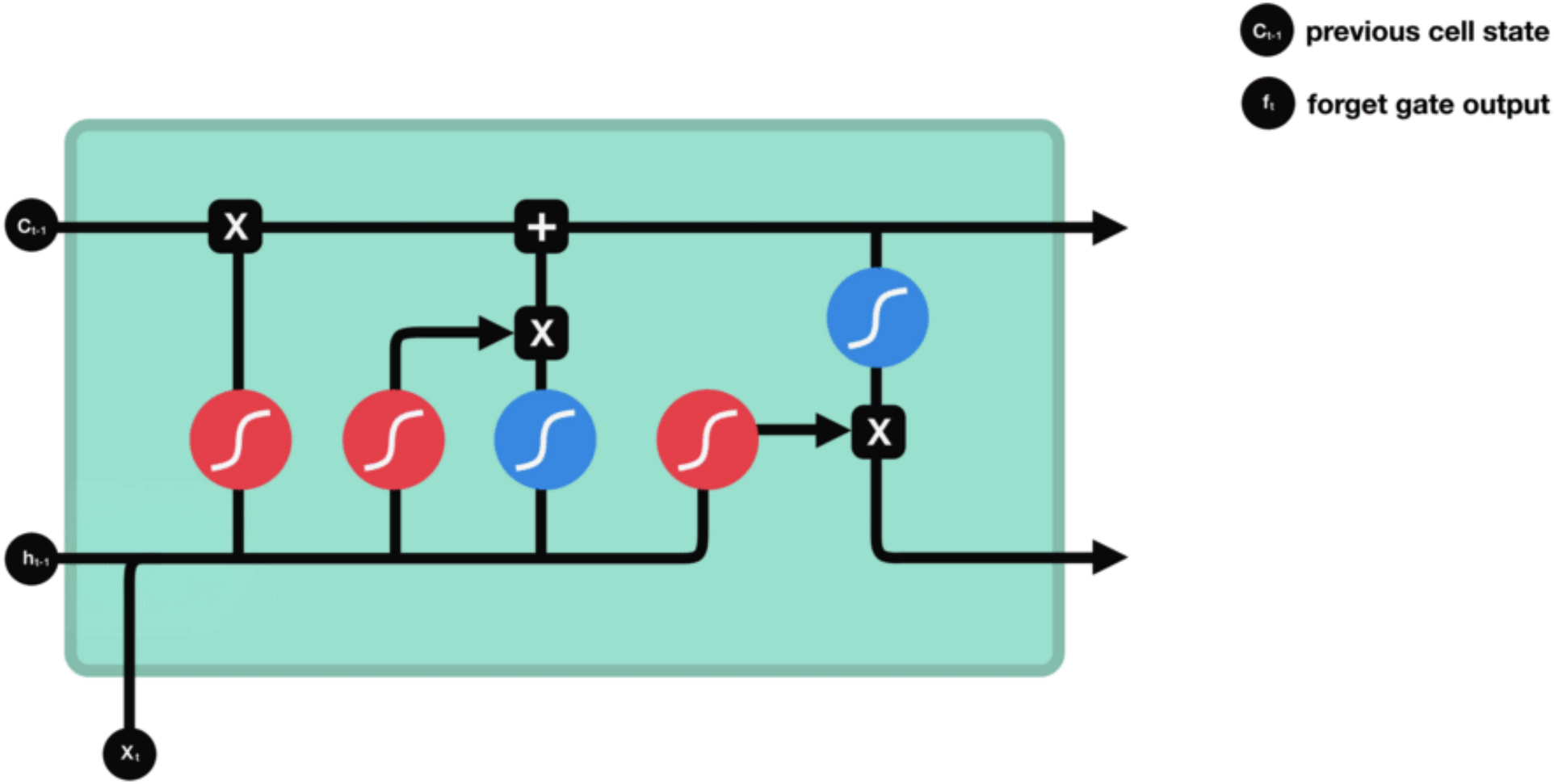


Remembering and forgeting



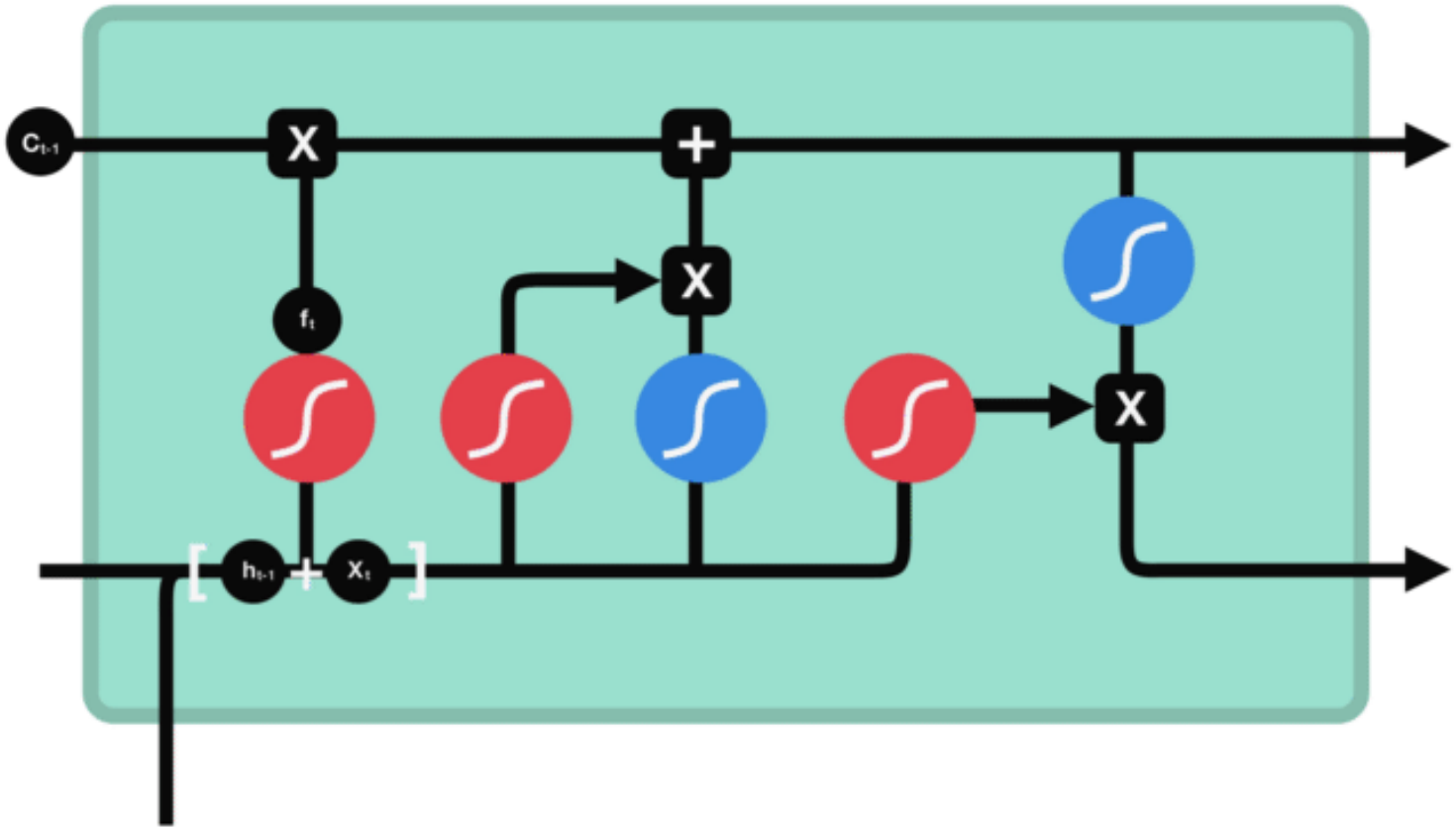
https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Long Short Term Memory



Forget Gate

Long Short Term Memory



C_{t-1} previous cell state

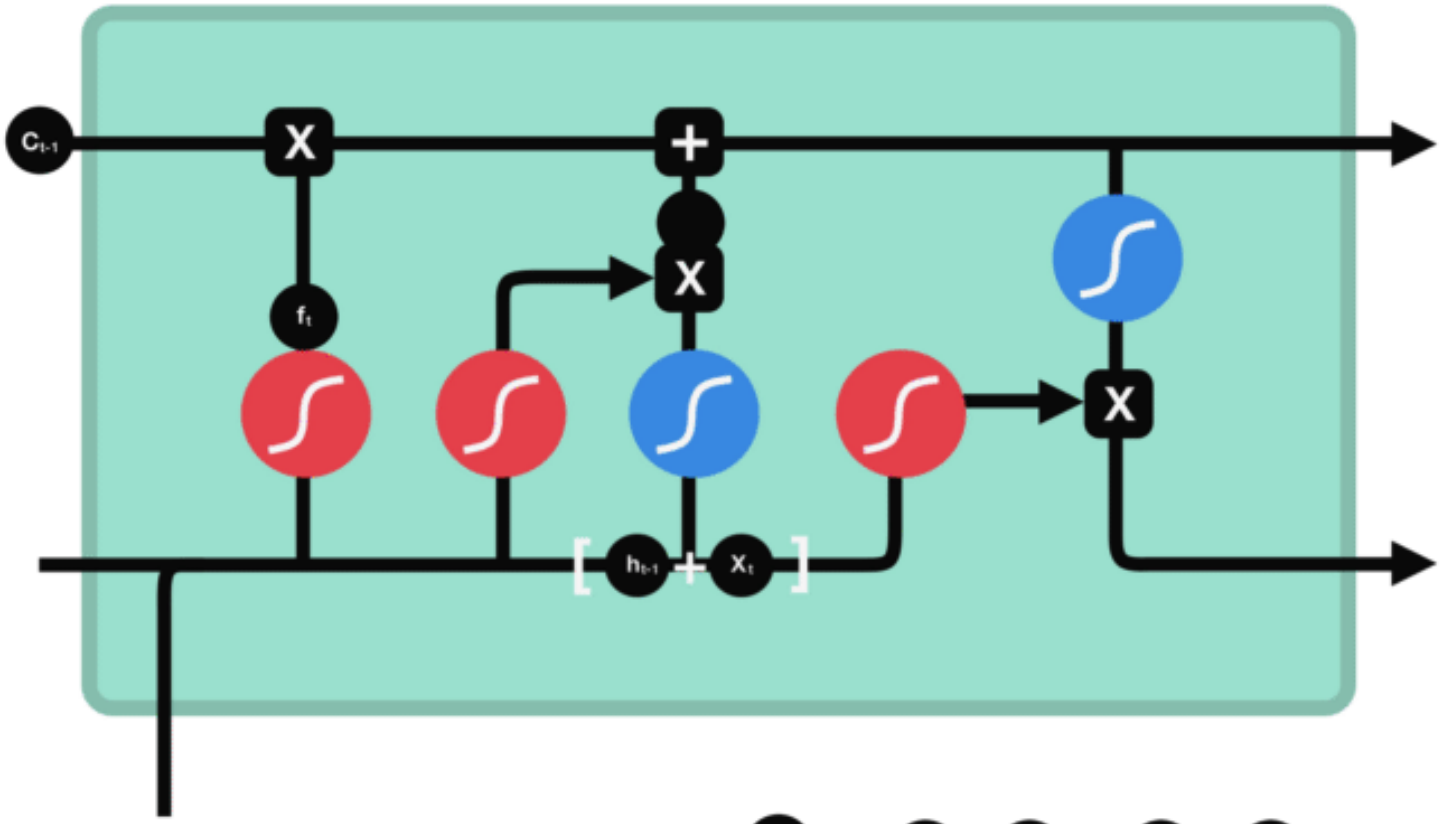
f_t forget gate output

i_t input gate output

$\tilde{C}_t$ candidate

Input Gate

Long Short Term Memory



C_{t-1} previous cell state

f_t forget gate output

i_t input gate output

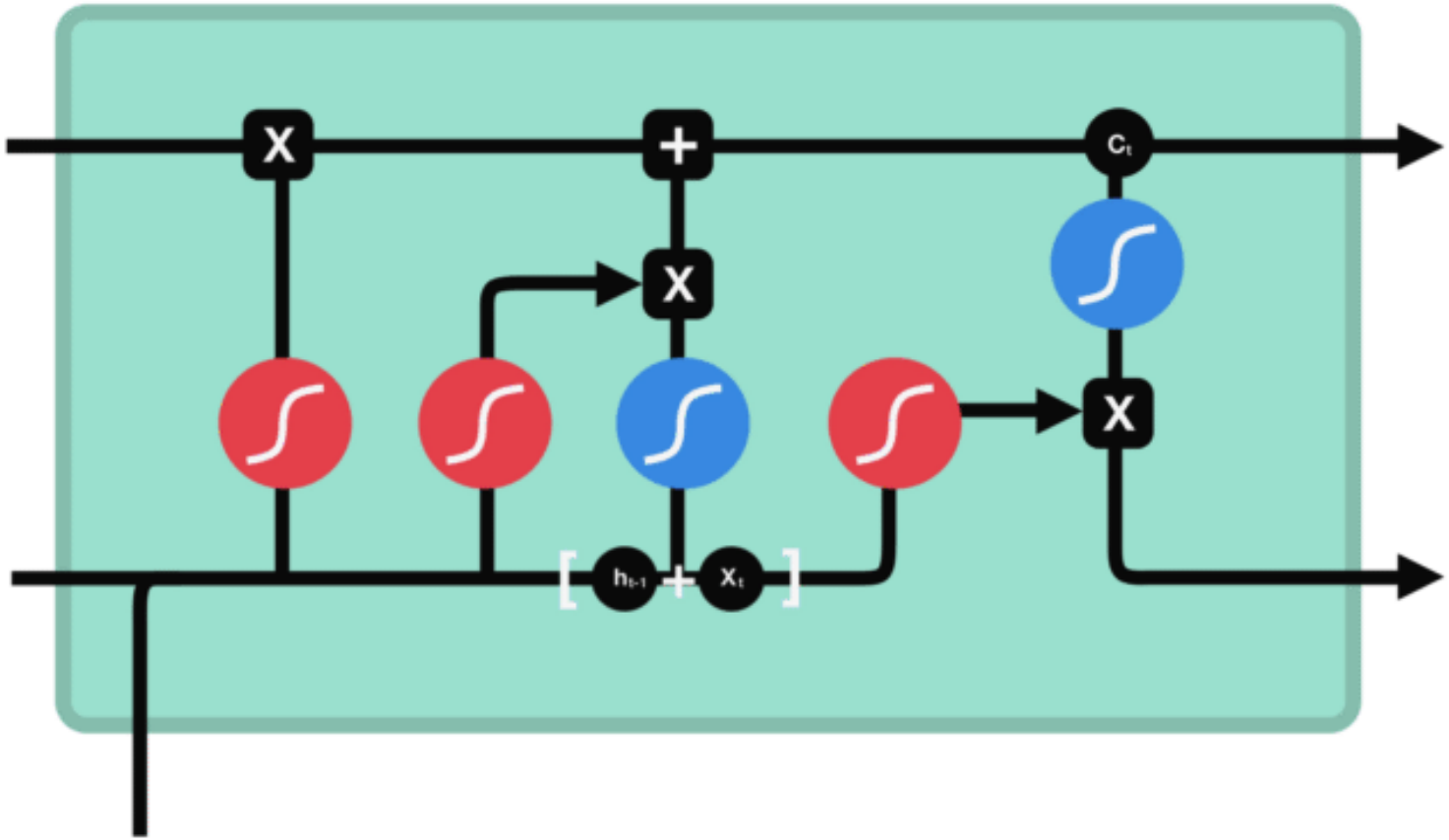
$\tilde{C}_t$ candidate

C_t new cell state

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Cell State

Long Short Term Memory



C_{t-1} previous cell state

f_t forget gate output

i_t input gate output

$\tilde{C}_t$ candidate

C_t new cell state

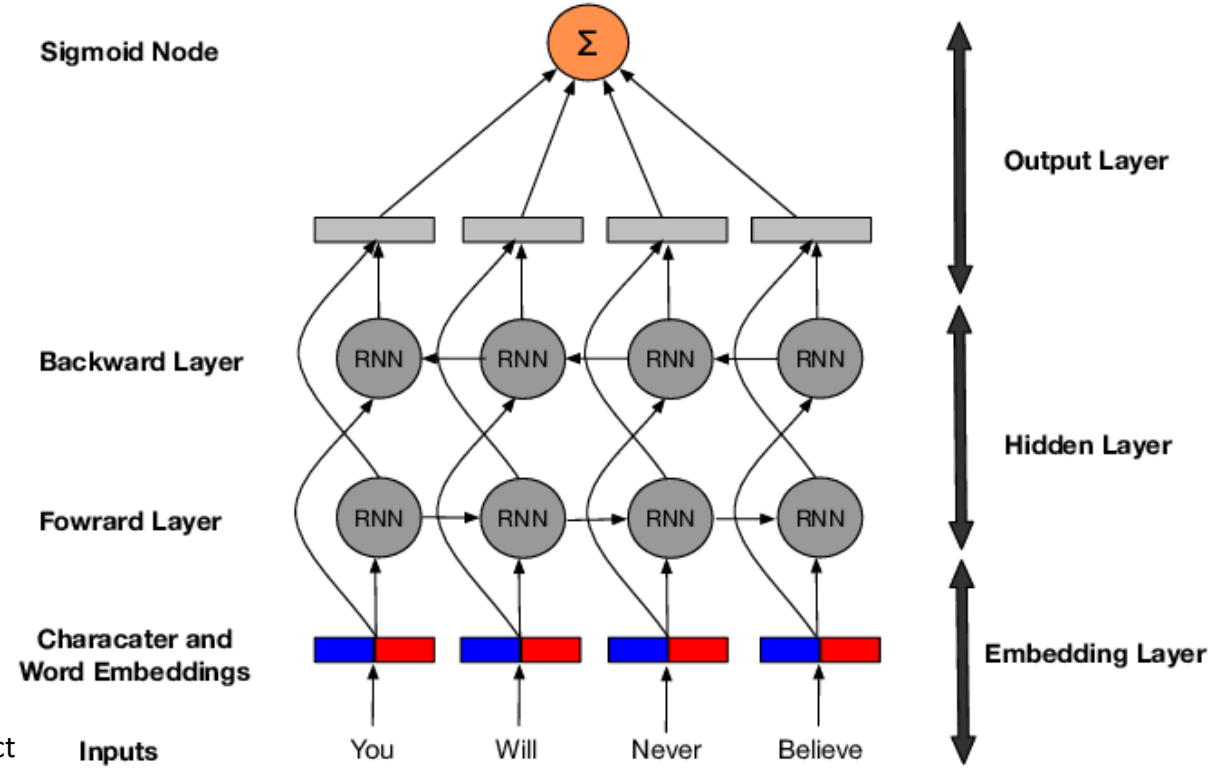
o_t output gate output

h_t hidden state

New Hidden State – Output Gate

Bi (Directional) (RNN) Long Short Term Memory

He said , "Teddy bears are on sale!"



Anand, Ankesh, Tanmoy Chakraborty, and Noseong Park. "We used neural networks to detect clickbaits: You won't believe what happened next!." European Conference on Information Retrieval. Springer, Cham, 2017.

https://arxiv.org/abs/1612.01340

Variable length

```
from keras.models import Sequential
from keras.layers import LSTM, Dense, TimeDistributed
from keras.utils import to_categorical
import numpy as np

model = Sequential()

model.add(LSTM(32, return_sequences=True, input_shape=(None, 5)))
model.add(LSTM(8, return_sequences=True))
model.add(TimeDistributed(Dense(2, activation='sigmoid')))

print(model.summary(90))

model.compile(loss='categorical_crossentropy',
              optimizer='adam')
```

Variable length

```
def train_generator():
    while True:
        sequence_length = np.random.randint(10, 100)
        x_train = np.random.random((1000, sequence_length, 5))
        # y_train will depend on past 5 timesteps of x
        y_train = x_train[:, :, 0]
        for i in range(1, 5):
            y_train[:, i:] += x_train[:, :-i, i]
        y_train = to_categorical(y_train > 2.5)
        yield x_train, y_train

model.fit_generator(train_generator(), steps_per_epoch=30, epochs=10,
verbose=1)

#example from https://datascience.stackexchange.com/questions/26366/training-an-rnn-with-examples-of-
different-lengths-in-keras
```

TimeDistributed layer

`TimeDistributed` class

```
tf.keras.layers.TimeDistributed(layer, **kwargs)
```

This wrapper allows to apply a layer to every temporal slice of an input.

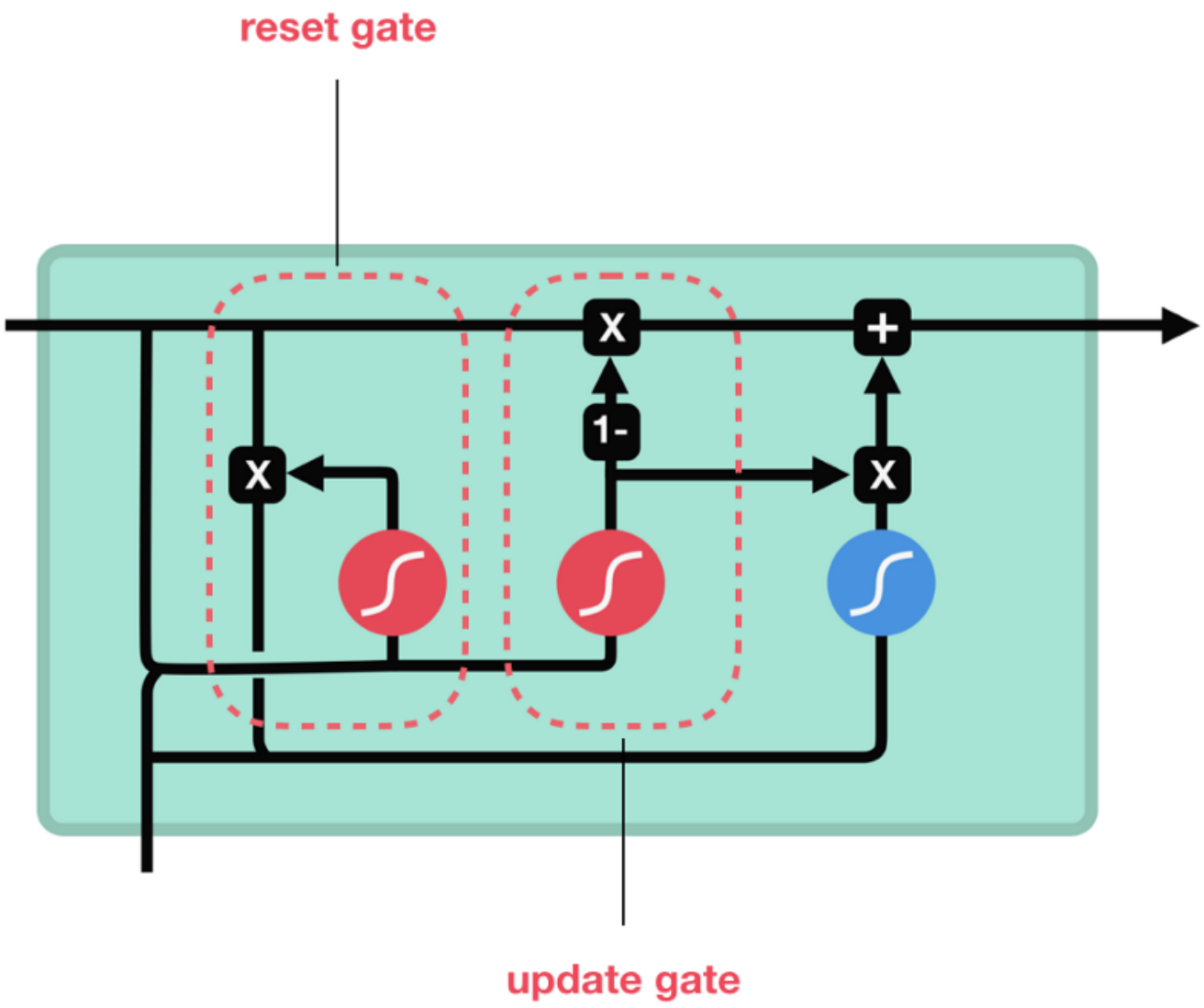
The input should be at least 3D, and the dimension of index one will be considered to be the temporal dimension.

Consider a batch of 32 video samples, where each sample is a 128x128 RGB image with `channels_last` data format, across 10 timesteps. The batch input shape is `(32, 10, 128, 128, 3)`.

You can then use `TimeDistributed` to apply a `Conv2D` layer to each of the 10 timesteps, independently:

```
>>> inputs = tf.keras.Input(shape=(10, 128, 128, 3))
>>> conv_2d_layer = tf.keras.layers.Conv2D(64, (3, 3))
>>> outputs = tf.keras.layers.TimeDistributed(conv_2d_layer)(inputs)
>>> outputs.shape
TensorShape([None, 10, 126, 126, 64])
```

Gated Recurrent Unit



LSTM – A couple of considerations

It helps to prevent vanishing gradient, however it did not solve it completely….

Long sequences problems, Try hundreds not thousands

More effective in forecasting and seq2seq than classification

It made history with chatbots, translators, speech-to-text, etc…

It might need lot of embeding ….

Resource Expensive if compared to Resnets…

LSTM – Long sequences Strategy

Truncate

Embed

Subsample

Auto Encoders

What is up for now? Attention Mechanism ...

Centro Brasileiro de Pesquisas Físicas

Redes Neurais profundas e aplicações Deep Learning

Clécio Roque De Bom – debom@cbpf.br

CBPF

clearnightsrthebest.com