Sequence to Sequence(Seq2Seq) Learning

Content

- NLP Tasks & Challenges
  - Text Classification
  - Machine Translation
  - Text Generation
- Introduction to Seq2Seq learning
  - Encoder -Decoder
- Pre Transformer Architectures & their Drawbacks
  - Brief Introduction to RNN & Its Drawbacks
  - Overcoming RNN Drawbacks through LSTM, Bi-LSTM
  - LSTM, Bi-LSTM Drawbacks & need for Transformer

NLP Tasks & Challenges

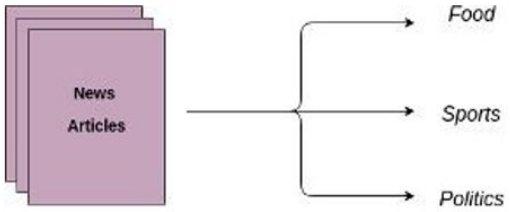
Tasks in NLP

Text Classification	Machine Translation	Text Generation
• Sentiment Analysis • Resume Analysis • Spam Analysis • Fraud Detection	• Speech to text • Text to Speech • Language Translation • Chatbots • Voicebots	• Text Summarization • Semantic Search • Autofill • Chatbots

Text Classification

Let's consider that we have the following headlines from news articles, and we want to classify them into three categories - **Sports, Politics, or Food**

1. *Paulo Dybala to leave Juventus in summer after contract not renewed.*
2. *Joe Biden to call China's Xi Jinping to discuss Russia, economic issues.*
3. *Know the health benefits of eating sprouts for breakfast.*



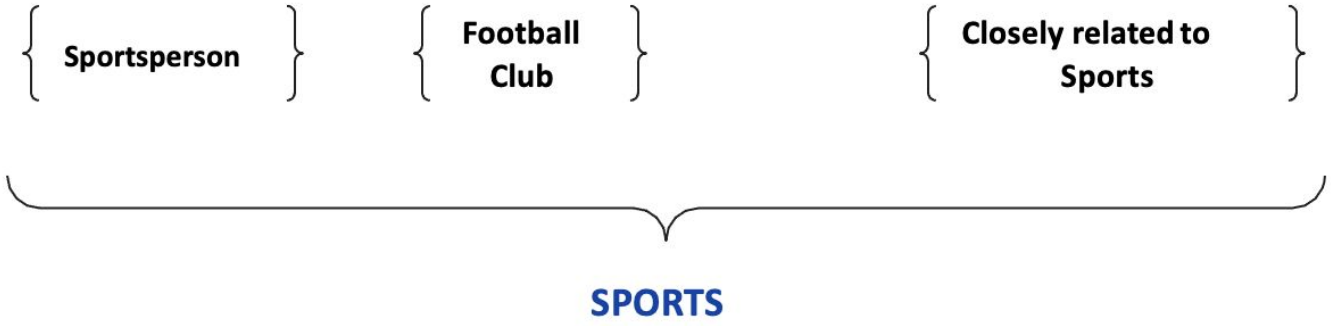
Text Classification

Let's take the first headline:

"Paulo Dybala to leave Juventus in summer after contract not renewed"

What are some of the key components that can help us in identifying the category of this article?

"Paulo Dybala to leave Juventus in summer after contract not renewed"



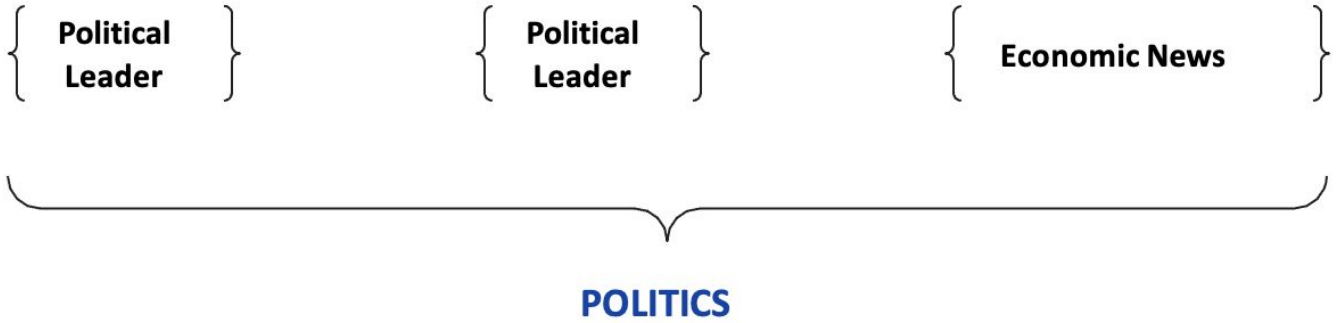
Text Classification

Similarly, let's look at the second headline:

"Joe Biden to call China's Xi Jinping to discuss Russia, economic issues"

What are some of the key components that can help us in identifying the category of this article?

"Joe Biden to call China's Xi Jinping to discuss Russia, economic issues"



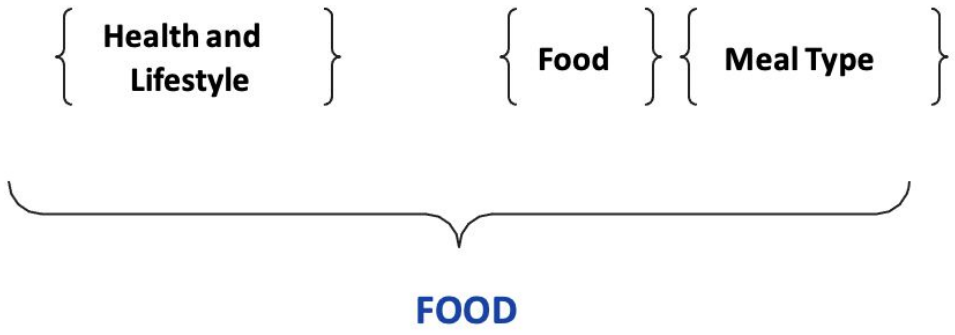
Text Classification

In the third headline:

"Know the health benefits of eating sprouts for breakfast"

What are some of the key components that can help us in identifying the category of this article?

"Know the health benefits of eating sprouts for breakfast"



Text Classification

- Similar to the previous examples, NLP facilitates the automatic analysis of text in order to classify it based on its context and that's where text classification comes into the picture.
- Text classification is the process of **categorizing text into one among a group of predefined classes.**
- **Two broad approaches to Text Classification are –**
  1. **Rule-based Systems**
  2. **Machine-based Systems**

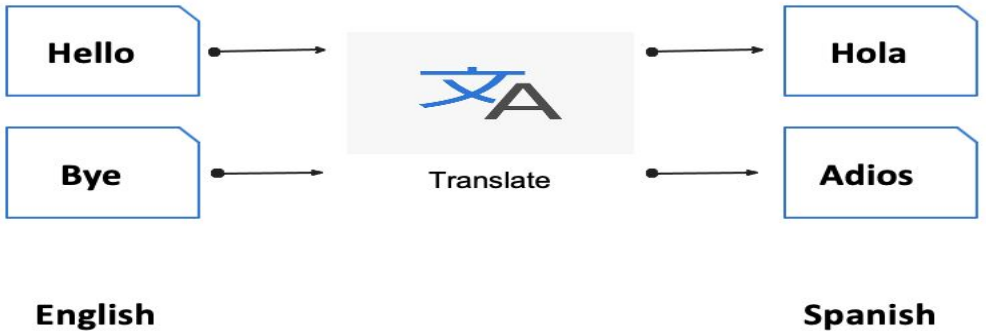
Text Classification

Rule-based System	Machine-based System
1. Text is classified into groups based on **predefined linguistic rules.** 2. These **rules can be a list of words** characterized by different classes. 3. For example, words like **pasta** and **doughnut** would be **classified as food** while words like **ball** and **pitch** would be into **sports**.	1. The machine classifies text into **categories based on past observations** from the dataset, and continuously learns patterns and words. 2. This model is then used to **predict classes for new text.** 3. For example, a dictionary {**pasta, pizza and chocolates, are, tasty, the**} is used for training an ML model and generating predictions to classify food items like **pasta** and **pizza** into the category **food**.

Machine Translation

- **There are over 6,500 languages spoken worldwide**, and it is impossible to know them all, making it a fairly difficult endeavour for humans to be able to translate written text from one language to another.

- Machine Translation refers to the idea of an **automated system** that **can translate text from one human language to another, while accounting for grammar syntax, semantics, and real-world information** . **Google Translate** is of course one frequently-used and freely available tool**,** where we give an **input in one language and we can get outputs in any other language**.



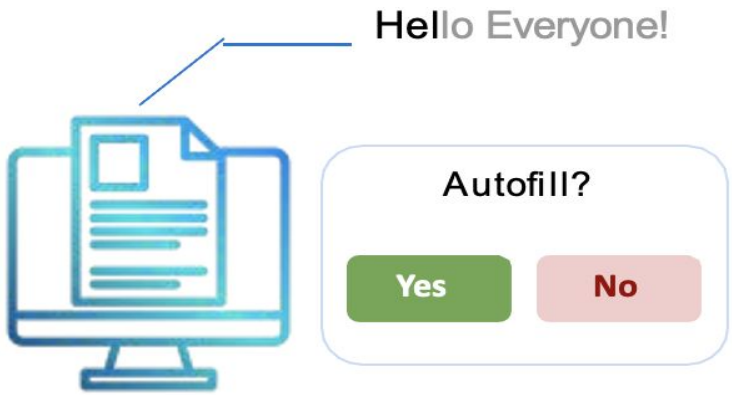
Machine Translation

- The different approaches that NLP has taken towards Machine Translation are:

Rule-based machine translation	Statistical machine translation	Neural machine translation
To achieve translation, a rule-based system requires experts to **define syntactic, semantic, and morphological rules** in the source and target languages.	Translations in statistical machine **translation are generated using statistical models** whose parameters are determined from the analysis of bilingual text.	Instead of simply translating each word, **Neural Machine Translation** (NMT) takes into account the context of a word using neural networks to provide more accurate translations.

Text Generation

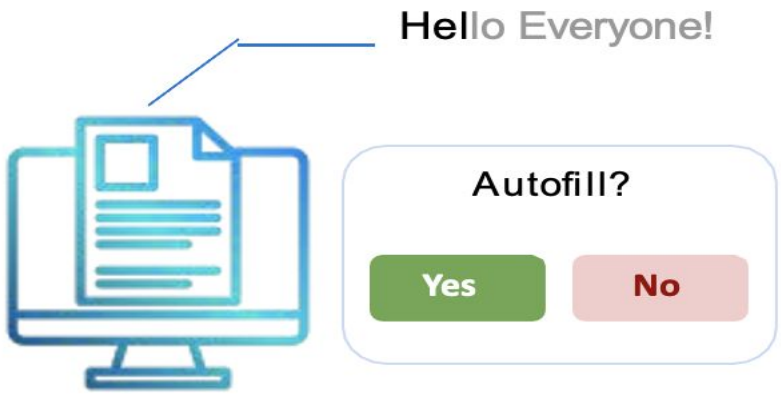
- Some common examples of text generation would be:
  - **the automatic completion of a sentence** in a word document
  - completion of sentences **while searching for something** on a web browser,
  - **automatic suggestions of next words** in an email



Text Generation

- Text Generation works on the prior sequence of words used in the text. A text generation model learns the likelihood of the occurrence of a word, and such models can be used at the character, sentence, or even paragraph levels.

- It is the core of NLP tasks like text summarization, chatbots, speech to text and several other applications.



Challenges in NLP

- **Contextual words, Phrases and Homonyms :**
  - Same words and phrases can have different meanings based on sentence context.
  - Example: "Let's go hang out by the pool" vs. "Let's go play some pool tomorrow."
- **Irony & Sarcasm :**
  - Humans use words with meanings opposite to their dictionary definitions.
  - Ambiguity involves sentences with multiple possible interpretations.
  - Example: "What lovely weather we have!" during bad weather.
- **Colloquialisms and Slang**
  - Informal expressions and culture-specific lingo pose challenges.
  - Example: "Hard to swallow" meaning difficult to believe.

Challenges in NLP

- **Domain-specific Language**
  - Different industries use specialized languages for activities.
  - Example: NLP models for healthcare vs. legal document processing.
- **Syntax vs. Semantics**
  - Correct syntax may lead to poor semantic value.
  - Example: "The elephant jumped over the wall to eat vada pav."
- **Syntax and Probability:**
  - Correct syntax, but improbable word sequence affecting semantic value.
  - Example: "NASA warns of five asteroids approaching the inner Solar system" vs. advice on coffee consumption.
- Basically, Navigating through the nuances of contextual meanings, linguistic subtleties, and industry-specific jargon poses significant challenges in Natural Language Processing. Overcoming these obstacles is essential for developing robust and accurate NLP models.

Introduction to Seq2Seq learning

What is Sequential Data ?

- Data where Order or Sequence of inputs matters
- For example,
  - Speech
  - Text
  - Music
  - Protein and DNA sequences
  - Stock prices and other time series

Text Processing and Seq2Seq Model

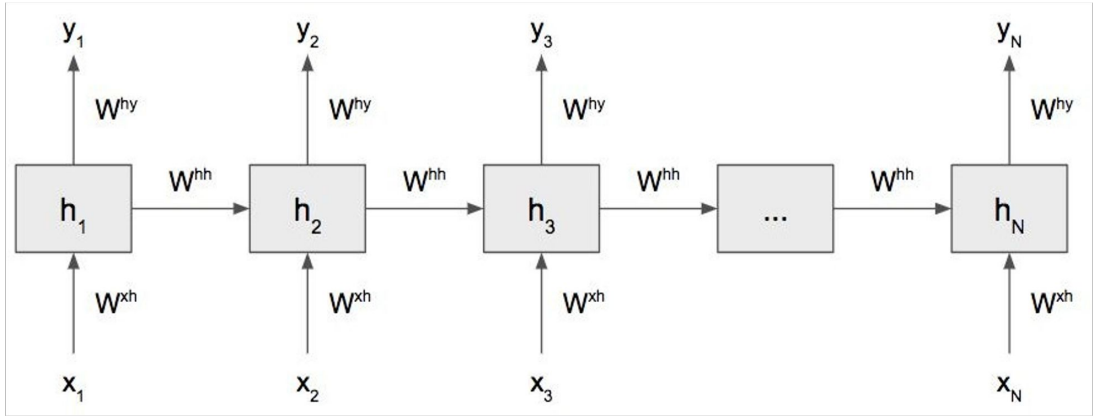
- Text data is always in the form of a sequence, and the prevailing idea was that order of the words in the sequence plays a large role in understanding the meaning of that text.

- A typical Seq2Seq processing model (for language translation task) consist of encoder and decoder and also known as encoder- decoded model.

- Seq2Seq Model Architecture
  - RNN
  - GRU
  - LSTM/BiLSTM

RNN Architecture

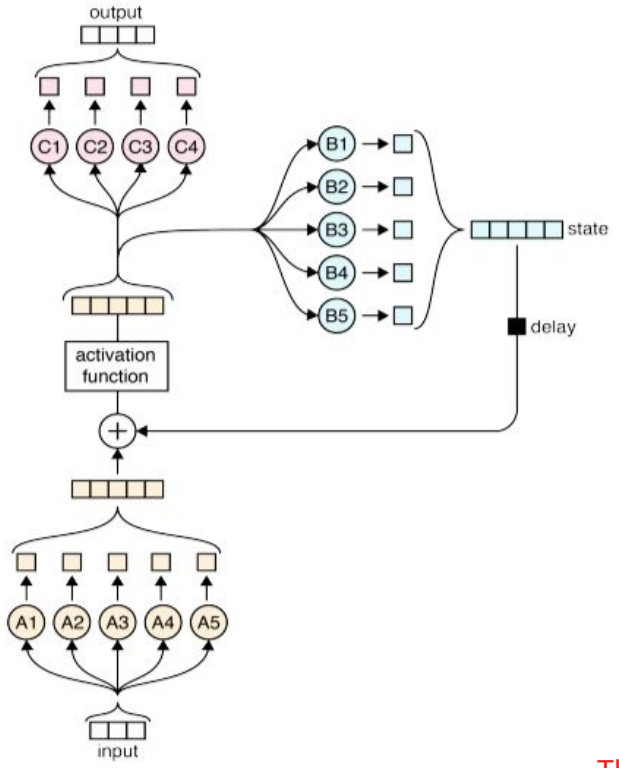
- One of the initial Seq2Se2 Architecture was RNN
- The Recurrent Neural Network (RNN) is a type of Neural Network designed to handle sequential data by modelling the dependencies between elements in a sequence.



- RNNs use loops to allow information to be passed from one step of the sequence to the next.
- RNN architecture is shown,
  - Here **x** represents the words in the text,
  - and **h** represents the nodes of the layer of the RNN

RNN cell implementation : the virtual vector space

We have seen RNN with only one neuron both in folded and unfolded form. However, in realistic implementation, each RNN cell consists of multiple hidden neurons



1. Input consists of 3 element (x1, x2, x3) encoding
2. Input passes thru a set of A neurons (as many A neurons required as the size of internal state vector)
3. Input hidden state is a 5 element vector in this example which means it is a 15 dimensional space
4. Previous hidden state (ht-1) is added elementwise to the trigger vector
5. The result of this addition is passed thru an activation layer of 5 neurons (shown as a rectangle)
6. The output of the activation function is passed thru B layer and C layer of neurons
7. C layer has as many neurons as required for output
8. B layer has as many neurons as the size of new hidden state generated
9. New hidden state h(t) is stored in delay unit to be used with next input
10. Size of input, output and hidden state can be different
11. Input trigger size should be same as hidden state size

How do RNNs work?

Input Sequence	Hidden State Computation	Output Computation	Back Propagation	Recurrent Structure
An input sequence is fed into the network one time step at a time. The input at each time step can be any type of data, such as text, speech, or even time series data.	**The hidden state at each time step is computed based on the current input and the previous hidden state.** This computation is done using a set of weights and biases that are learned during training. The hidden state acts as a memory of past inputs and is used to make predictions for the current time step.	**The output of the RNN at each time step is computed based on the current hidden state.** This allows the network to produce a sequence of outputs for a sequence of inputs.	**During training, the error is back propagated through time to update the weights and biases in the network.** This allows the network to learn patterns in the input sequences and improve its performance over time.	**The hidden state of the network at time step t is influenced by the hidden state at time step t-1.** This recurrent structure allows the network to have a **"memory"** of past information.

RNN Limitations

RNN Cell:

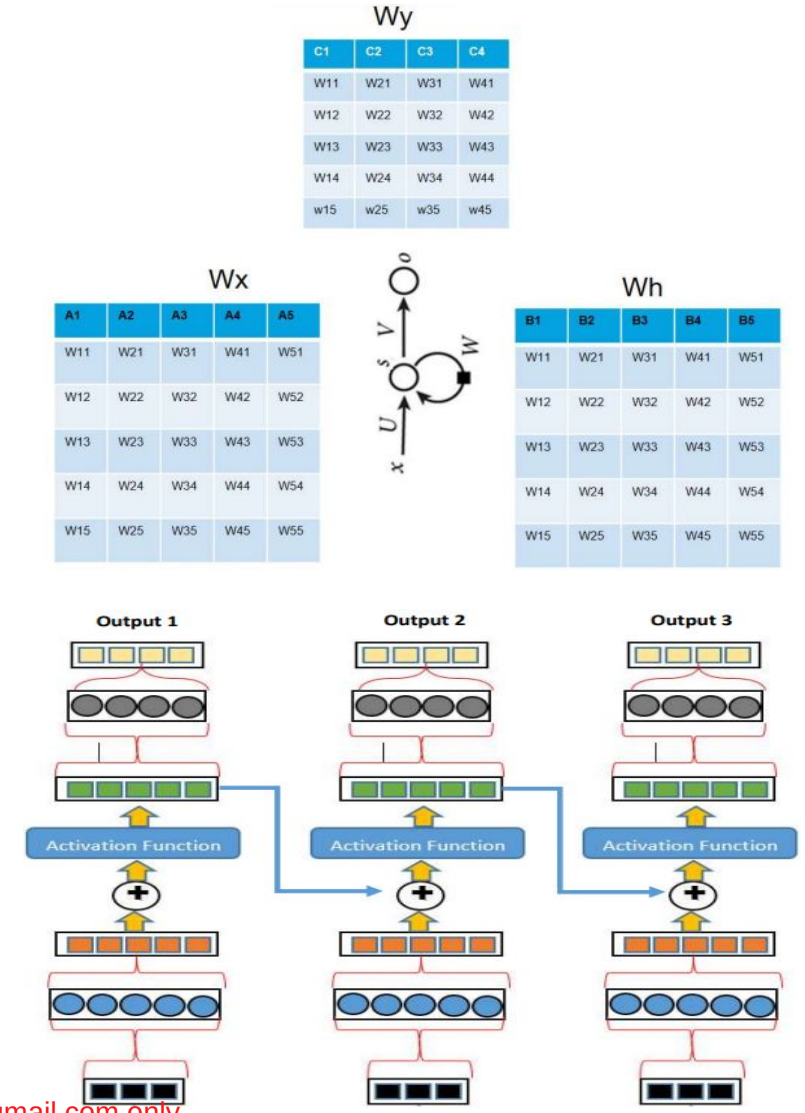
1. When RNN is unrolled into n instances (n = 3) in the picture. Every instance shares the same set of weight matrix

2. During the training process, using the back propagation method, the RNN algorithm has to learn the weights such that the weights are good for all the instances / steps

3. When the number of instances increase, this becomes difficult to achieve due to a phenomenon called vanishing gradient

4. As a result the weights are not equally well learnt for all instances. The RNN fails to understand long term relations between words

5. It cannot map the statistical relation between words separated over a long chain in the input



RNN Drawbacks

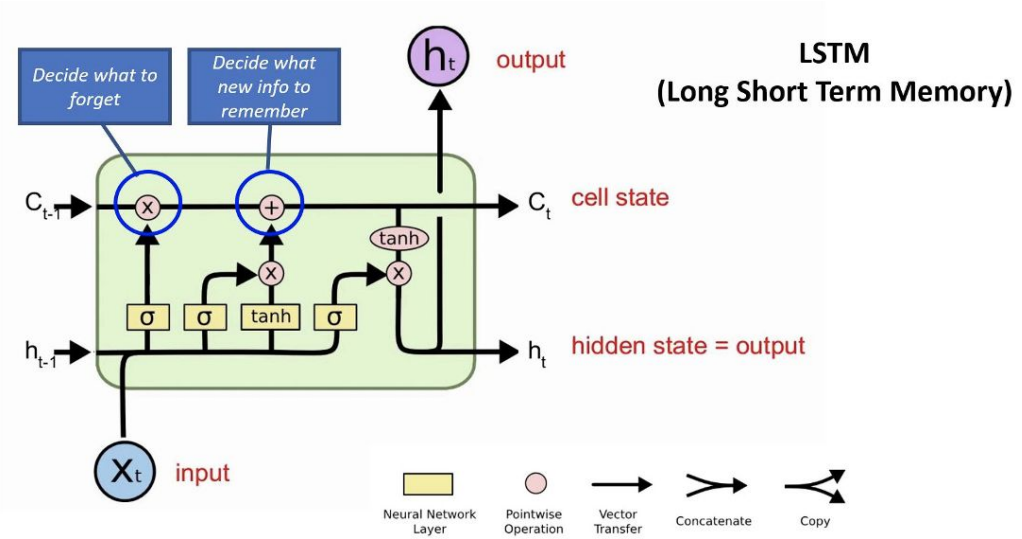
- Vanishing or exploding gradient or overfitting
- Short memory /Difficulty in accessing information from long time ago
- Slow computation for long sequence
- Inadequate Memory

Long Short-term Memory Networks (LSTMs)

- The LSTM architecture was developed as an attempt to address limitations with RNNs, by allowing the network to **“forget”** unimportant information and **“store”** important information for later, through the usage of a **gating mechanism inside each cell.**
- These innovations were important because they allowed LSTMs to **sidestep the Overfitting problem** by forgetting unimportant information, and **also sidestep the Vanishing Gradient problem** by using gates to remember important information in their cell state for long periods of time.
- LSTMs were hence able to make higher-quality predictions on text wrt RNN and were the de facto NLP model architecture before the rise of Transformers.

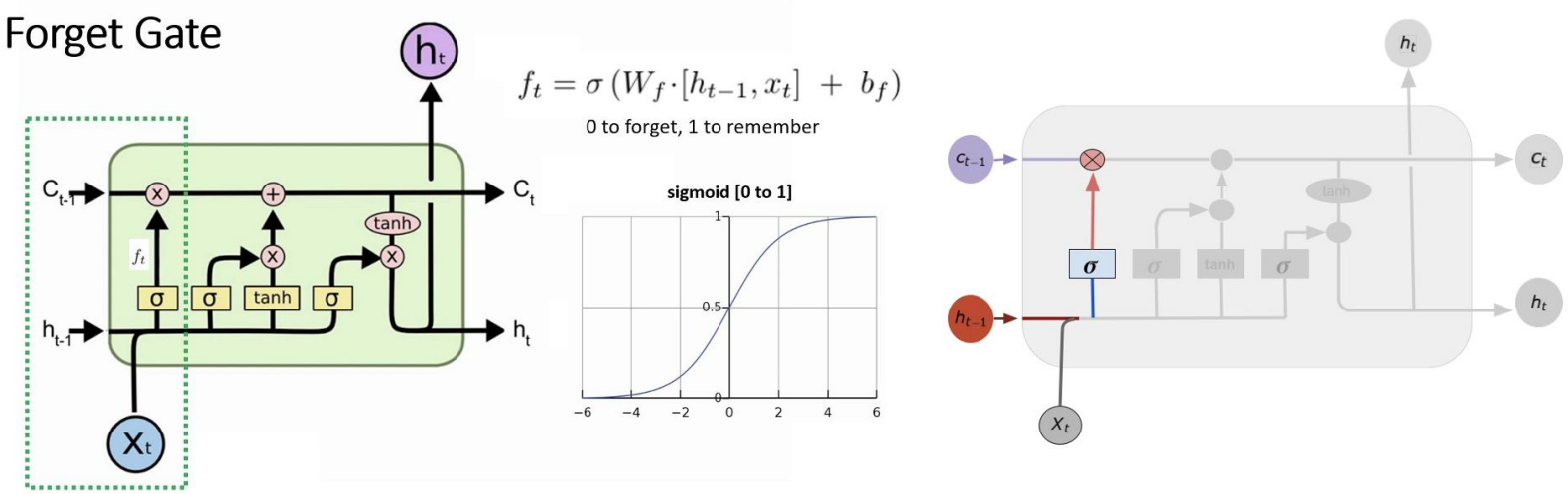
Long Short-term Memory Networks (LSTMs)

- The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.
- The cell state is updated twice with a few computations that result in stabilized gradients. This is the Long-term Memory of the LSTM.
- However, the LSTM model also has a separate hidden state that acts like a Short-term Memory.
- This dual split role of Long-term & Short-term memory inside the same network was the key improvement of LSTMs to the RNN architecture.



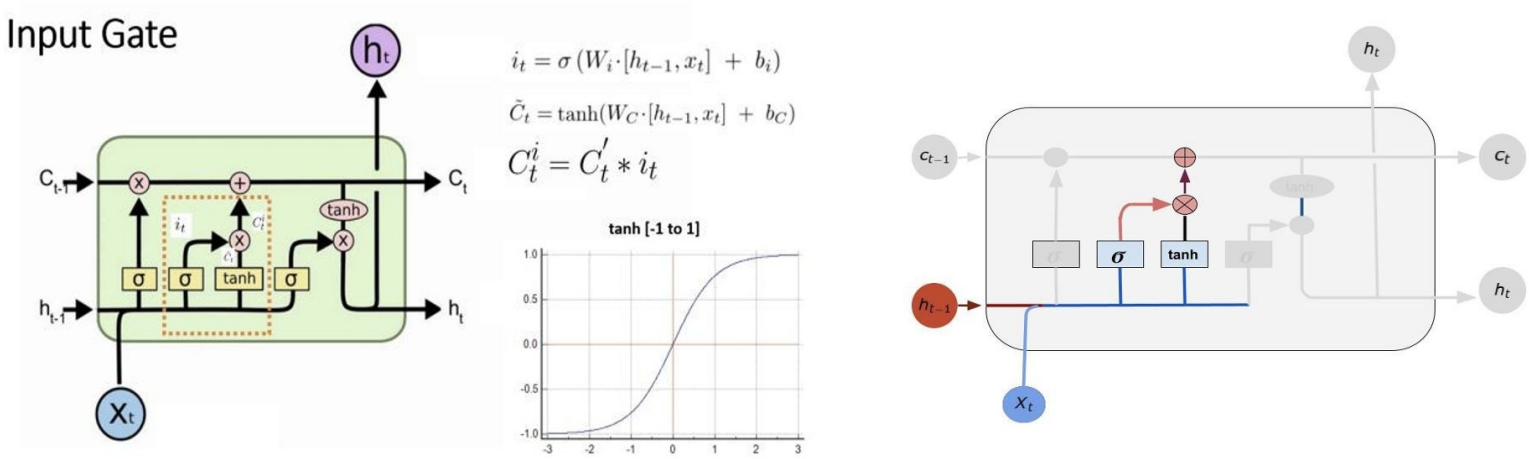
LSTMs - The Forget Gate

- The first kind of gate present in the LSTM is the Forget Gate.
- The idea is to decide what information we're going to throw away from the cell state.
- This decision is made by a Sigmoid activation function called the "Forget Gate" layer.



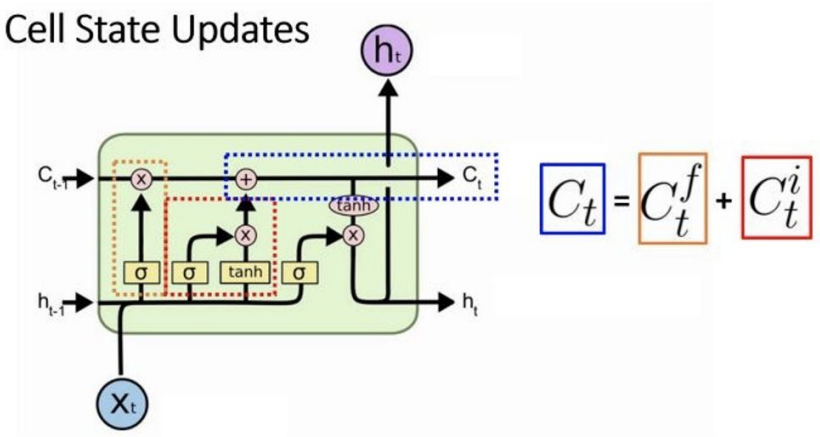
LSTMs - The Input Gate

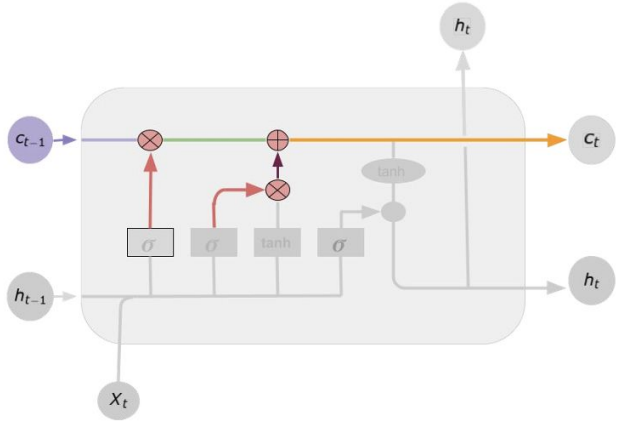
- The next step is to decide what new information to store in the cell state. This has two parts.
- First, a Sigmoid layer called the “Input Gate” layer decides which values we’ll update.
- Next, a Tanh layer creates a vector of new candidate values that could be added to the state.



Combining the Forget & Input Gate Outputs

- Next, we'll combine these two (Forget & Input) to create an update to the cell state.
- This updates the old cell state to the new cell state. This is the output of the Long-term memory.

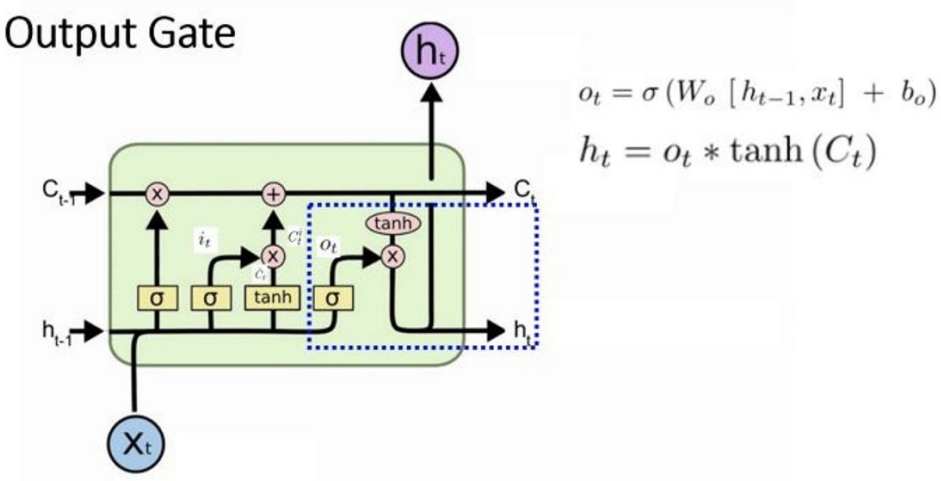


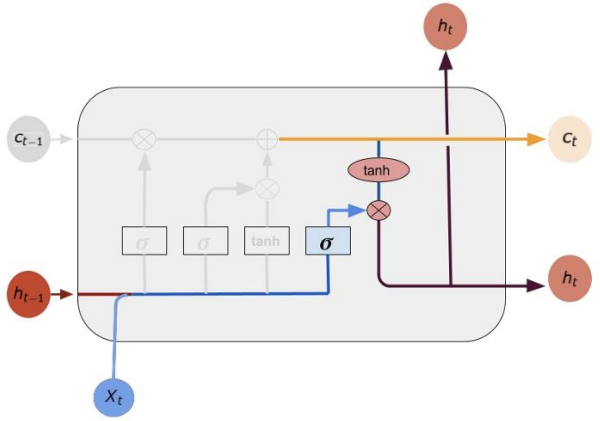


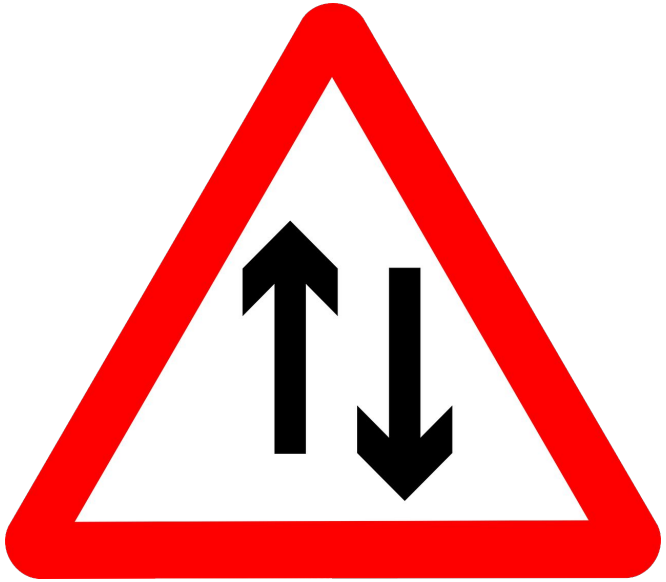
$$c_t = f_t * c_{t-1} + i_t \tilde{c}_t$$

LSTMs - The Output Gate

- The final output is based on our cell state, **but a filtered version of it run through one more activation.** This is the output of the Short-term memory portion of the LSTM.







Bidirectional LSTM

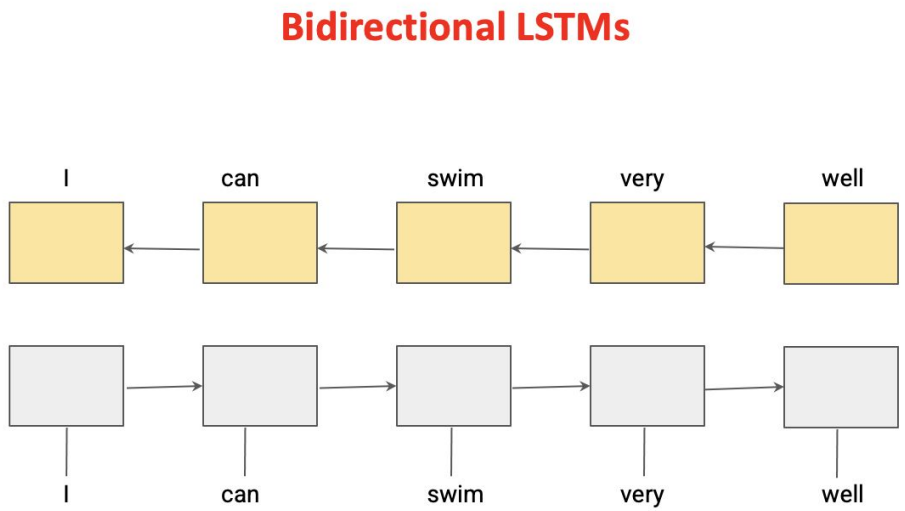
Bidirectional LSTMs

- Two LSTMs are used for better learning

  Data for one LSTM gets fed in forward direction

  while for other LSTM, data is fed in reverse direction

- Output of both LSTM gets merged



Implementing in Keras - Bidirectional LSTM

```
model.add(Bidirectional(LSTM(256, return_sequences=True, merge_mode='concat'))
```

merge_mode can be '**concat**', '**sum**' or '**mul**'

LSTM Drawbacks & Need for Transformer

- LSTMs when combined with an input Neural Word Embedding mechanism such as Word2Vec, were able to achieve the best performance across various NLP tasks upto 2016-17 - with comfortably better performance than the earlier statistical text encoding ideas such as TF-IDF. LSTMs and their variants were used by every major industry player, for applications ranging from Alexa's Voice Assistant, to Google Search, Google Translate, and even the Autosuggest in Smartphone Keyboards.

- However, LSTMs also had some key limitations - especially relating to their Sequential Text Processing mechanism - processing one word at a time could not computationally scale well to large documents and corpuses of text, and there was a need for a more parallelizable computing architecture that could accept an entire corpus at once and send each word / token of text through its own pathway independent of the rest. There was also a realization that despite the smart hack of creating a Gated Mechanism, LSTMs still couldn't resolve the fundamental RNN problem of remembering the very long-range dependencies that are common in flowing paragraphs of text.

- Both these limitations required a fundamentally different architecture for their resolution, and the industry was able to achieve this in one go with the invention of the Transformer in 2017, which we will learn next session.