

# **ABSTRACTIVE TEXT SUMMARISATION**

GROUP NO- 39

GUIDE : DR. RAJU HAZARI

TEAM MEMBERS :

BHUKYA VASANTH KUMAR – B180441CS

BILLA AMULYA – B180404CS

SEVAKULA JYOTHI – B180359CS

# TABLE OF CONTENTS

- ▶ Introduction
- ▶ Problem Statement
- ▶ Literature Survey
- ▶ Design
- ▶ Work done
- ▶ Future Work Plan
- ▶ References

# INTRODUCTION

1. Text Summarisation is the task of generating a short and concise summary that captures the salient ideas of the source text.
2. There are two different approaches Extractive Summarisation and Abstractive Summarisation.
3. Abstractive Summarisation generates new sentences, possibly rephrasing or using the words that were not in the original text. This ensures that the core information is conveyed through the shortest text possible. This reduces reading time and accelerates the process of searching the information.

# PROBLEM STATEMENT

To generate a short, precise summary of a longer text by retaining the key information of the text using Natural language processing (NLP) techniques.

## **Input:**

1. Data(Sentence/Paragraph)
2. Original summary of data (Used for calculating accuracy)

## **Output:**

A short summary is generated based on abstractive text summarisation using seq2seq model with LSTM's

# LITERATURE SURVEY

- Abstractive techniques need a more profound examination of the text i.e it needs deeper analysis. These techniques can produce new sentences, and improve the focus of the summary to maintain a decent compression rate[4].
- An RNN Encoder decoder based architecture, which is based on sequence to sequence model is applied to process the data in sequential manner.
  - The input of any state may depend on the output of the previous states [5,6], this scenario is most likely to occur in a sentence, where the meaning of a word is closely related to the previous words meaning.
- Study [17], explains the problem of vanishing gradients, which happens while training a long sequence with an RNN, is solved with gated RNNs. Allowing the gradients to backpropagate along a linear path using gates, each with a weight and bias, can address this problem. The weights and biases of the gates are updated during training.



# LITERATURE SURVEY

- Rnn encoder-decoder summarisation is utilised with LSTM units and attention to generate headlines from the text of news articles. The model is quite effective in predicting headlines from the same newspapers as it was trained on. [18]
- From [19], we can notice that LSTM and GRU are generally used for abstractive summarisation. since LSTM has a memory unit that provides extra control but the computation time of the GRU is less. Also, while it is easier to tune the parameters with LSTM, the GRU takes less time to train.
- in [4], we can notice that few methods applied LSTM to resolve the gradient vanishing problem that occurred when using an RNN, while other approaches applied a GRU.

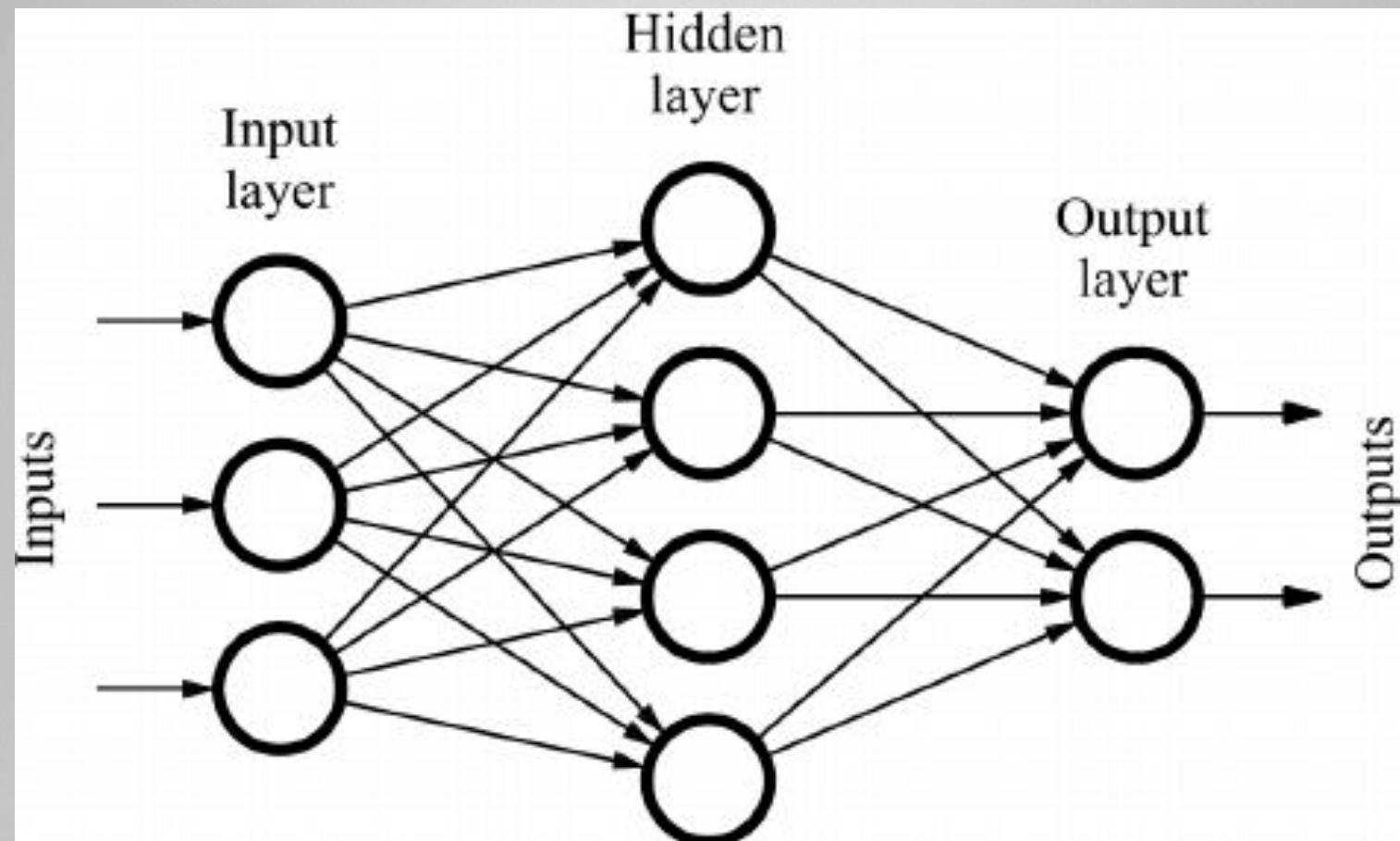
# What is a Neural Network?

- ▶ Neural networks used in deep learning consist of different layers connected to each other and work on the structure and functions of a human brain. It learns from huge volumes of data and uses complex algorithms to train a neural net.

# What is a feed forward Neural Network?

- ▶ In a feed forward network information flows in forward direction from the input nodes through the hidden layers if they are present and to the output nodes . There are no cycles or loops in the network





## **Problems with feed forward Neural Networks**

- ▶ Decisions are based on the current input
- ▶ No memory about the past
- ▶ No future scope
- ▶ Not Designed for sequences / time series data, hence the results with time series/sequential data are bad.
- ▶ Example of Sequential data:  
Sentences, Stock Prices, Video Stream  
etc.

# What are RNNs?

- ▶ Recurrent Neural Networks are types of neural networks designed for capturing information from sequences / time series data. It is a special kind of artificial neural network that permits continuing information related to past knowledge by utilizing a special kind of looped architecture.

# How does RNN work ?

- Recursive Formula

$$S_t = F_w (S_{t-1}, X_t)$$

$X_t$  - Input at time step  $t$

$S_t$  - State at time step  $t$

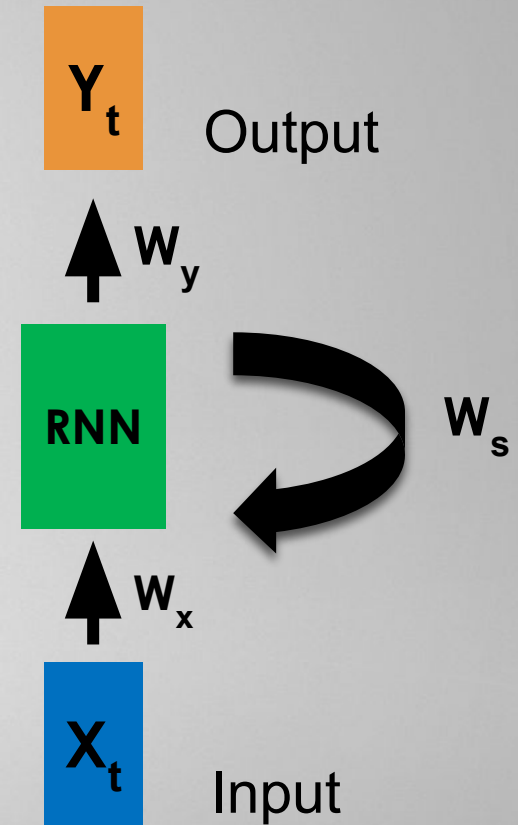
$F_w$  - Recursive function

# Simple RNN

$$S_t = F_w (S_{t-1}, X_t)$$

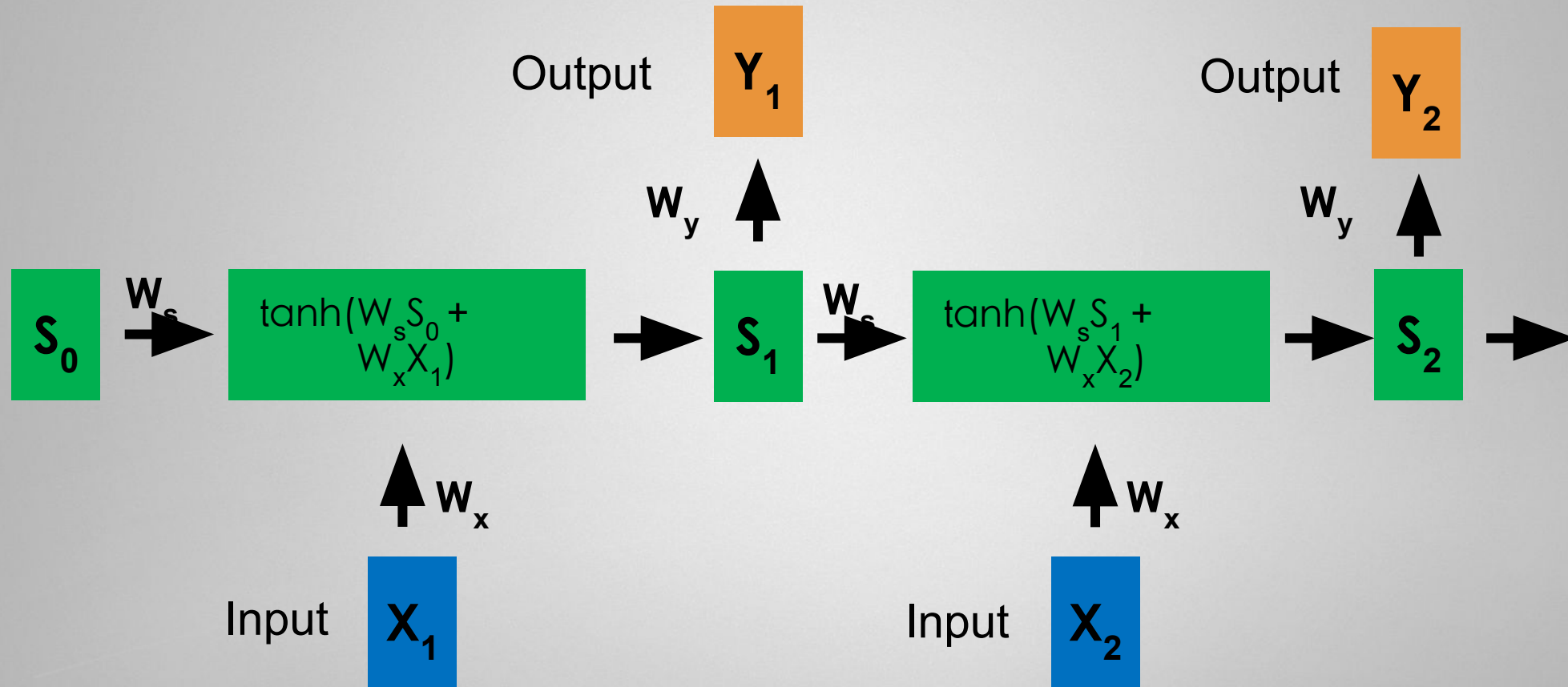
$$S_t = \tanh(W_s S_{t-1} + W_x X_t)$$

$$Y_t = W_y S_t$$

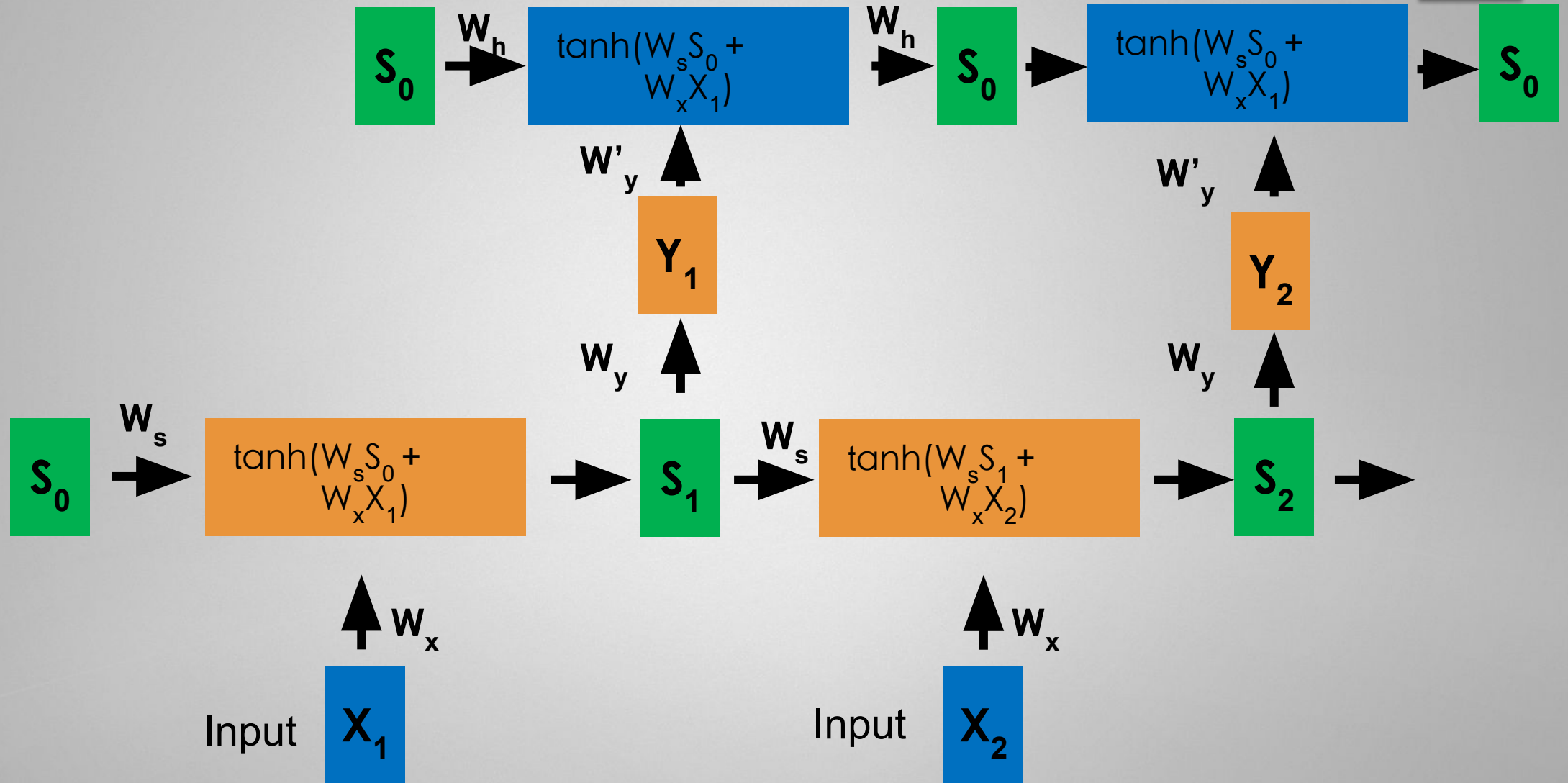




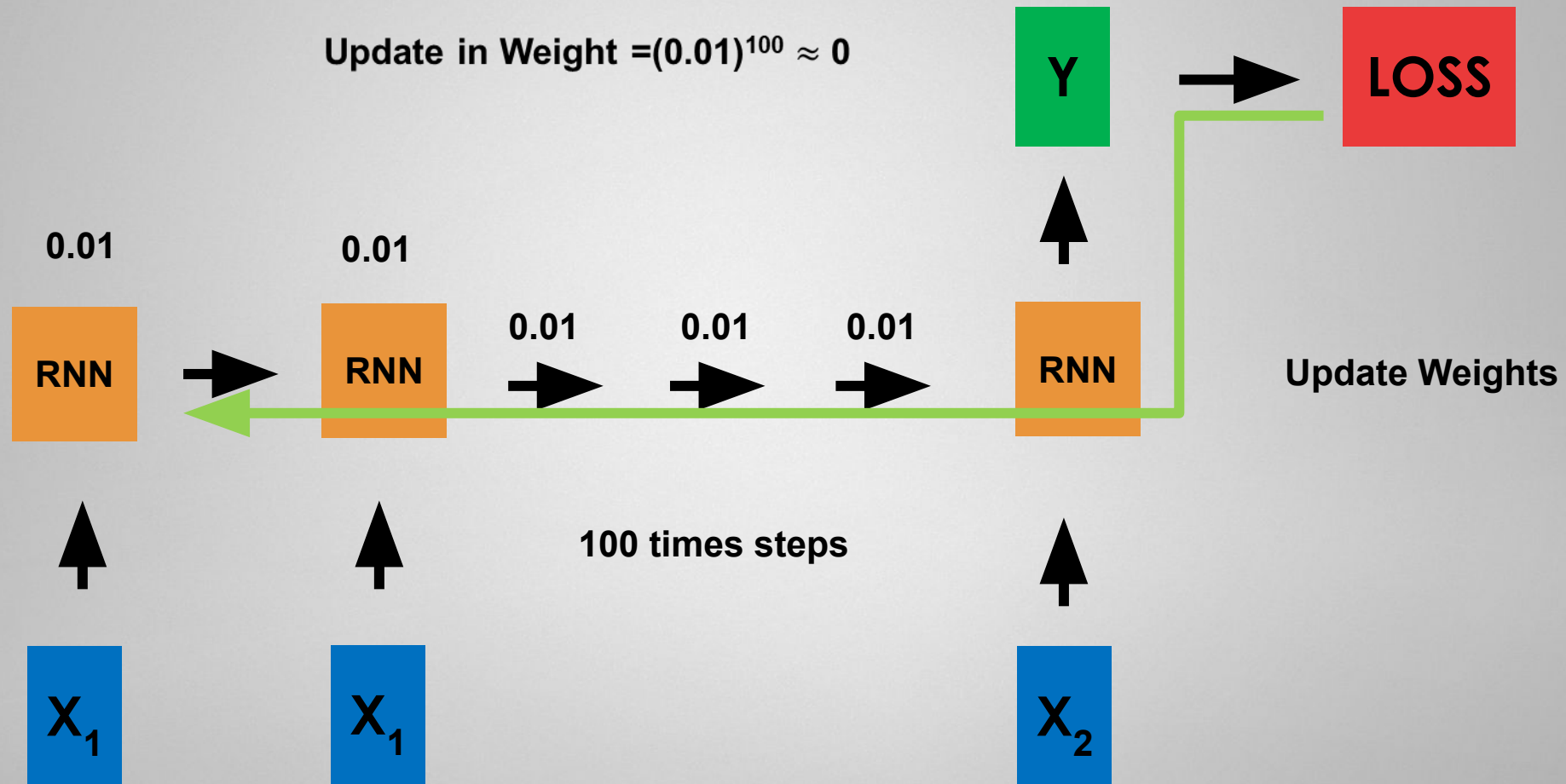
# Simple RNN (Unrolled)



## Multiple Hidden Layers



# Vanishing Gradient Problem



# Why LSTM?

- ▶ LSTMs are special kind of recurrent neural networks capable of learning long-term dependencies among the information for long periods of time is their default behaviour.
- ▶ Main Advantages:
  - It solves the vanishing gradient problem by adding extra interactions
  - Works way better than RNNs

# LSTM

$$f_t = \sigma(W_f S_{t-1} + W_f X_t) \quad - \text{Forget Gate}$$

$$i_t = \sigma(W_i S_{t-1} + W_i X_t) \quad - \text{Input Gate}$$

$$o_t = \sigma(W_o S_{t-1} + W_o X_t) \quad - \text{Output Gate}$$

$$\tilde{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$$

$$c_t = (i_t * \tilde{C}_t) + (f_t * c_{t-1}) \quad - \text{Cell State}$$

$$h_t = o_t * \tanh(c_t) \quad - \text{New State}$$



# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

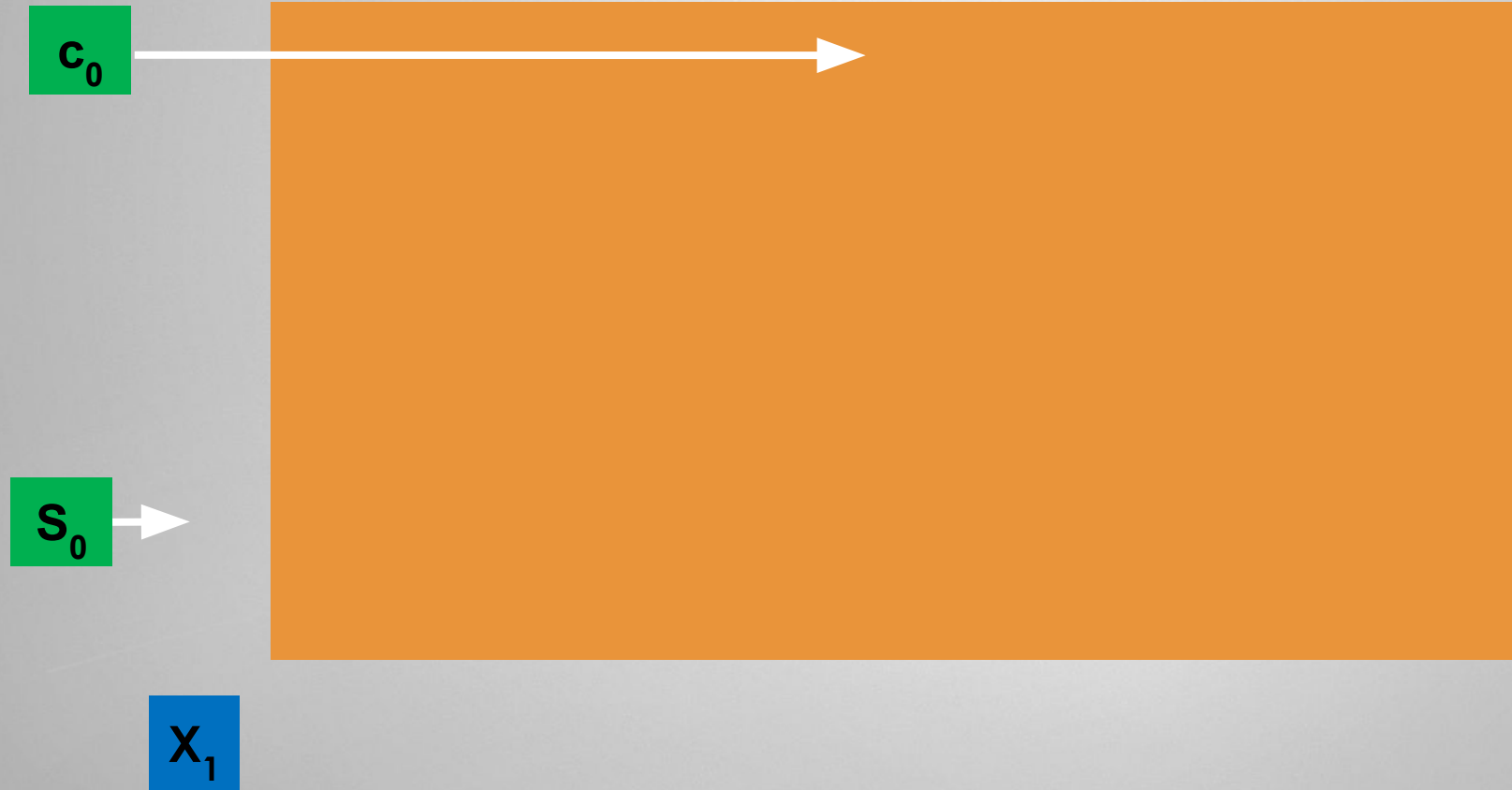
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State



# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

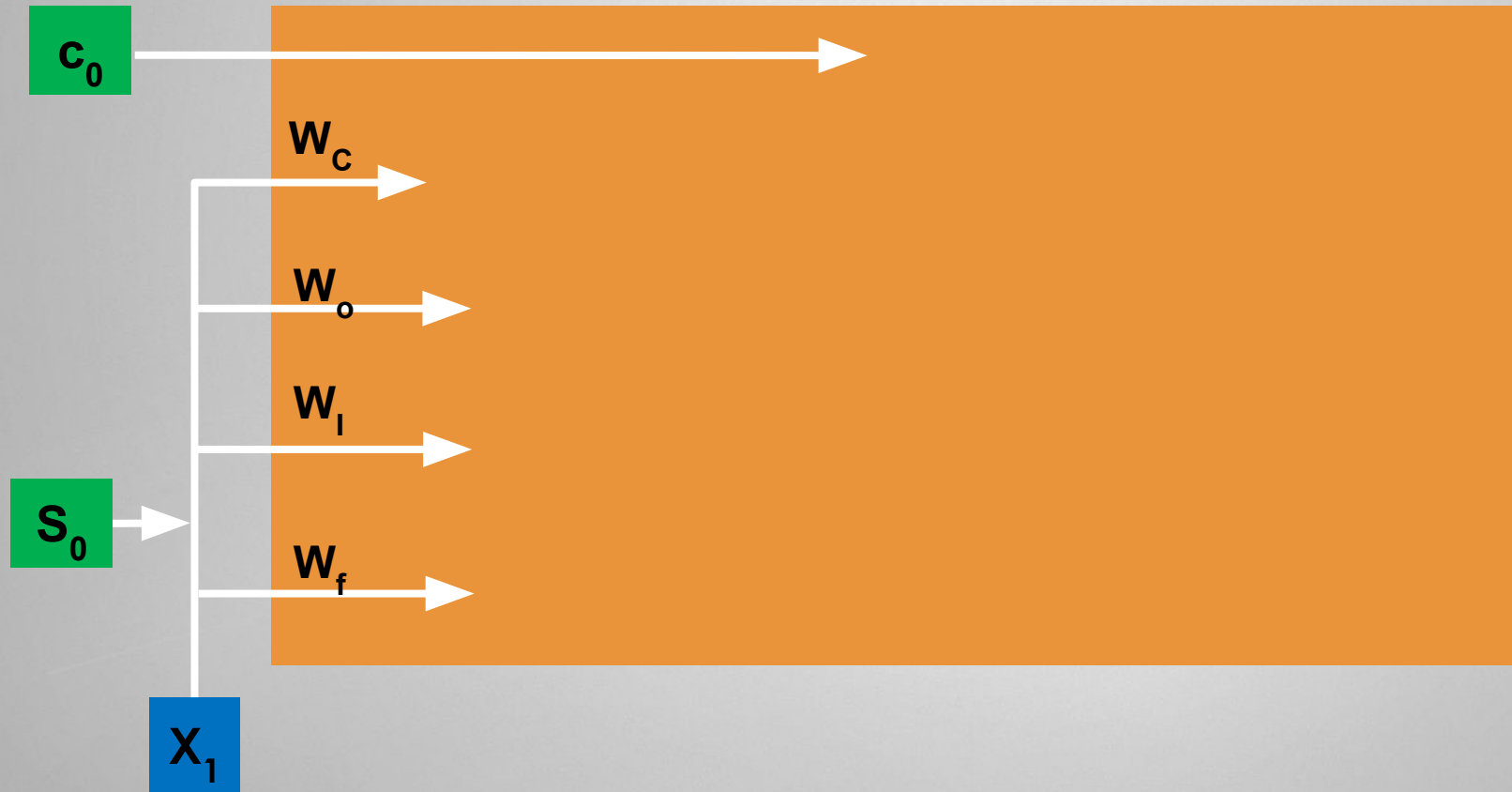
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State



# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

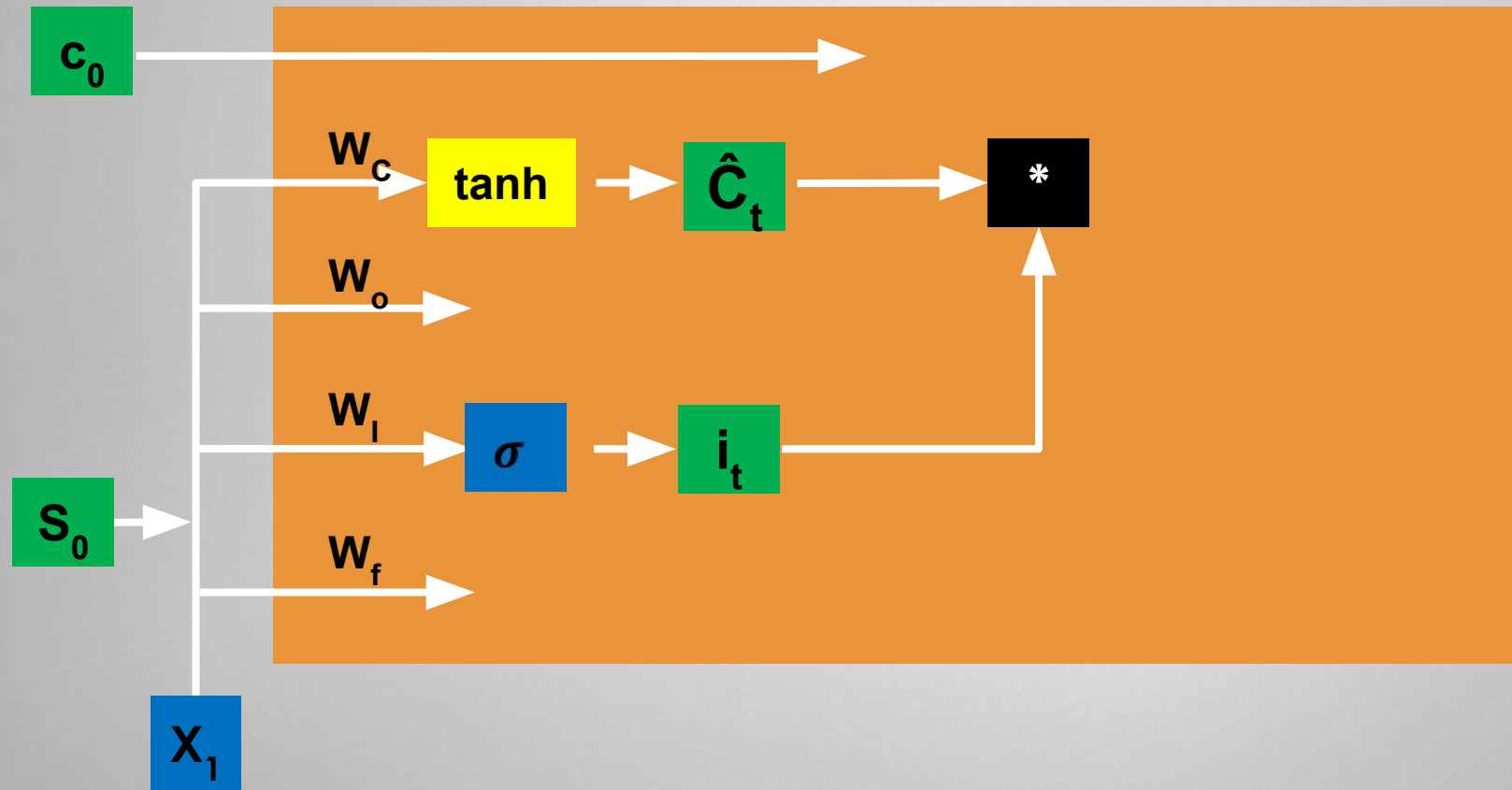
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State



# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

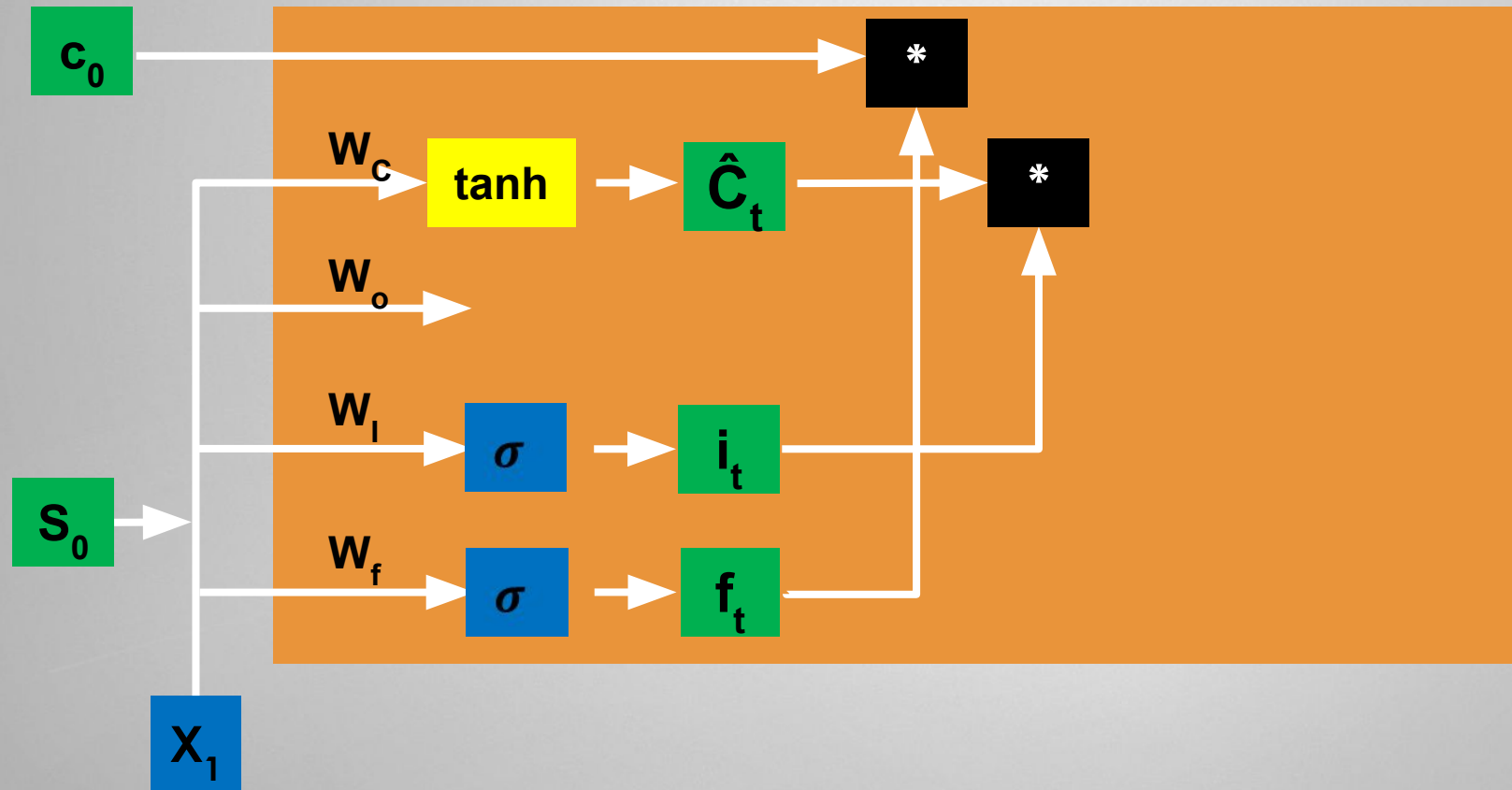
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State



# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

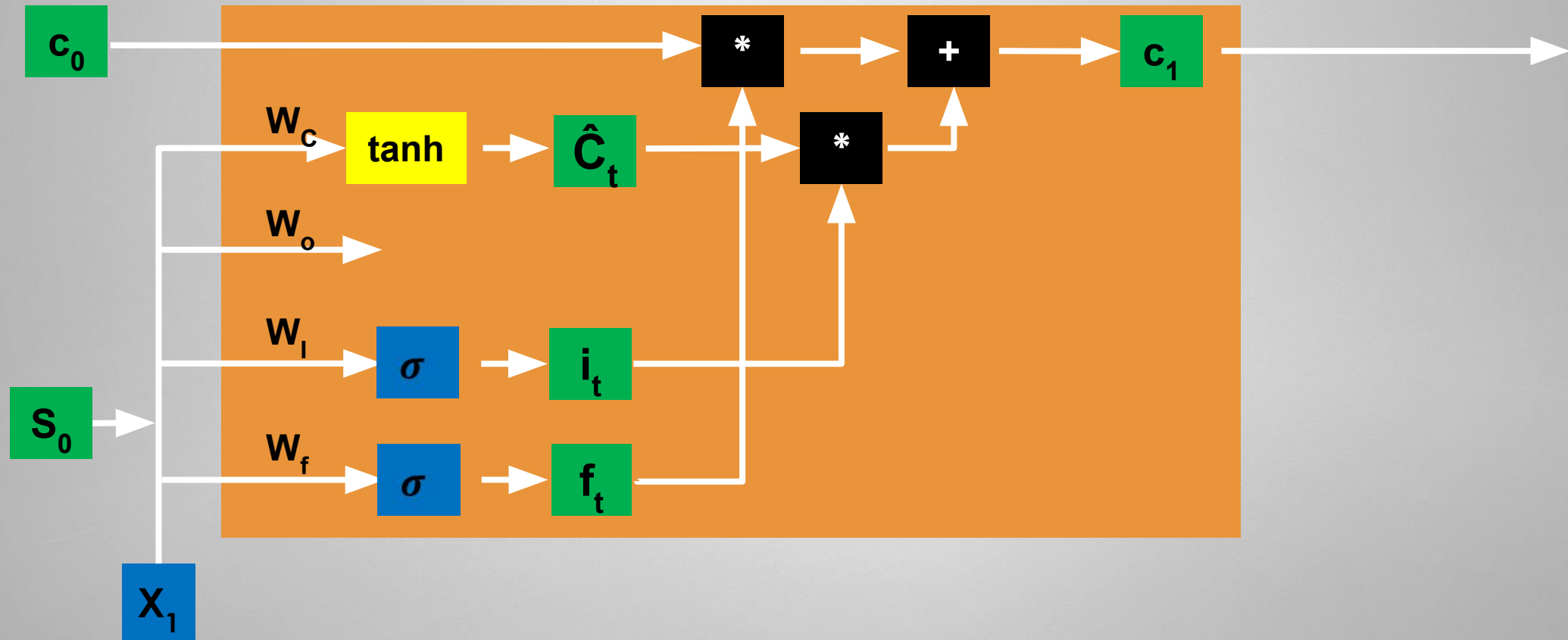
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State





# LSTM

$f_t = \sigma(W_f S_{t-1} + W_f X_t)$  - Forget Gate

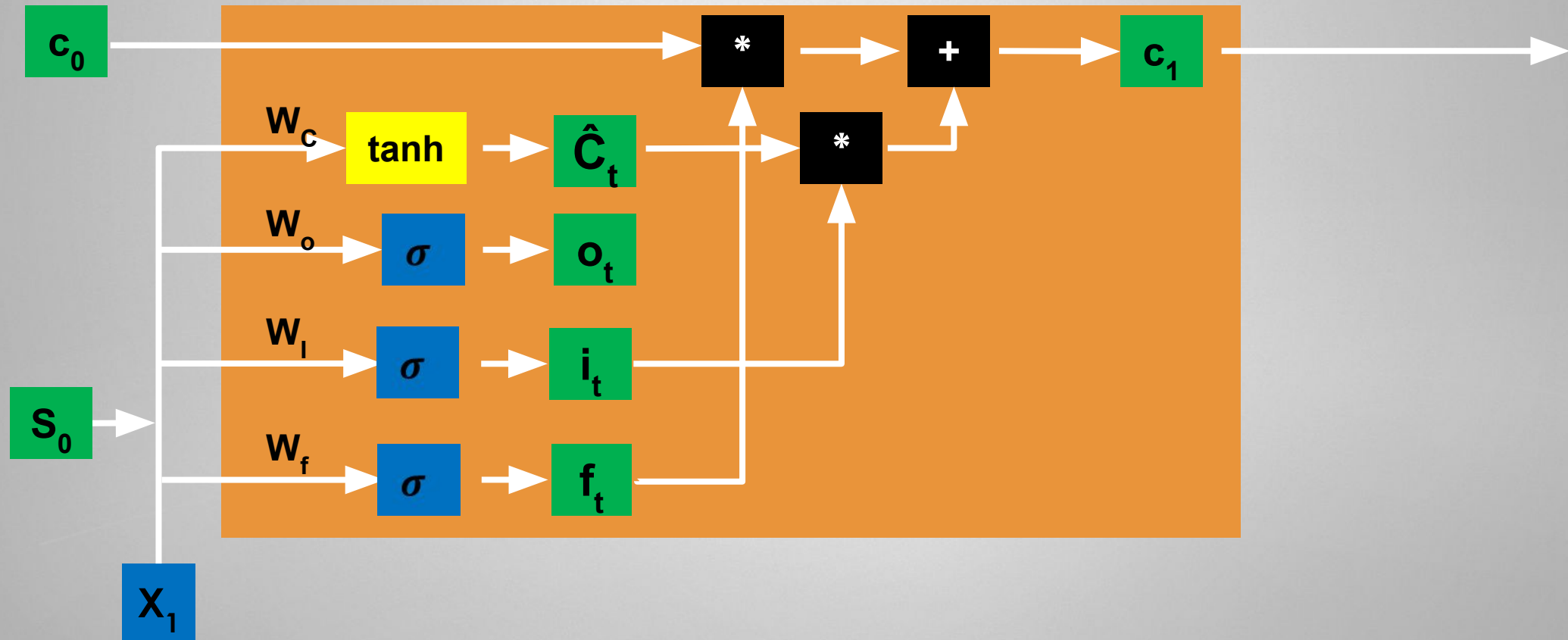
$i_t = \sigma(W_i S_{t-1} + W_i X_t)$  - Input Gate

$o_t = \sigma(W_o S_{t-1} + W_o X_t)$  - Output Gate

$\hat{C}_t = \tanh(W_c S_{t-1} + W_c X_t)$

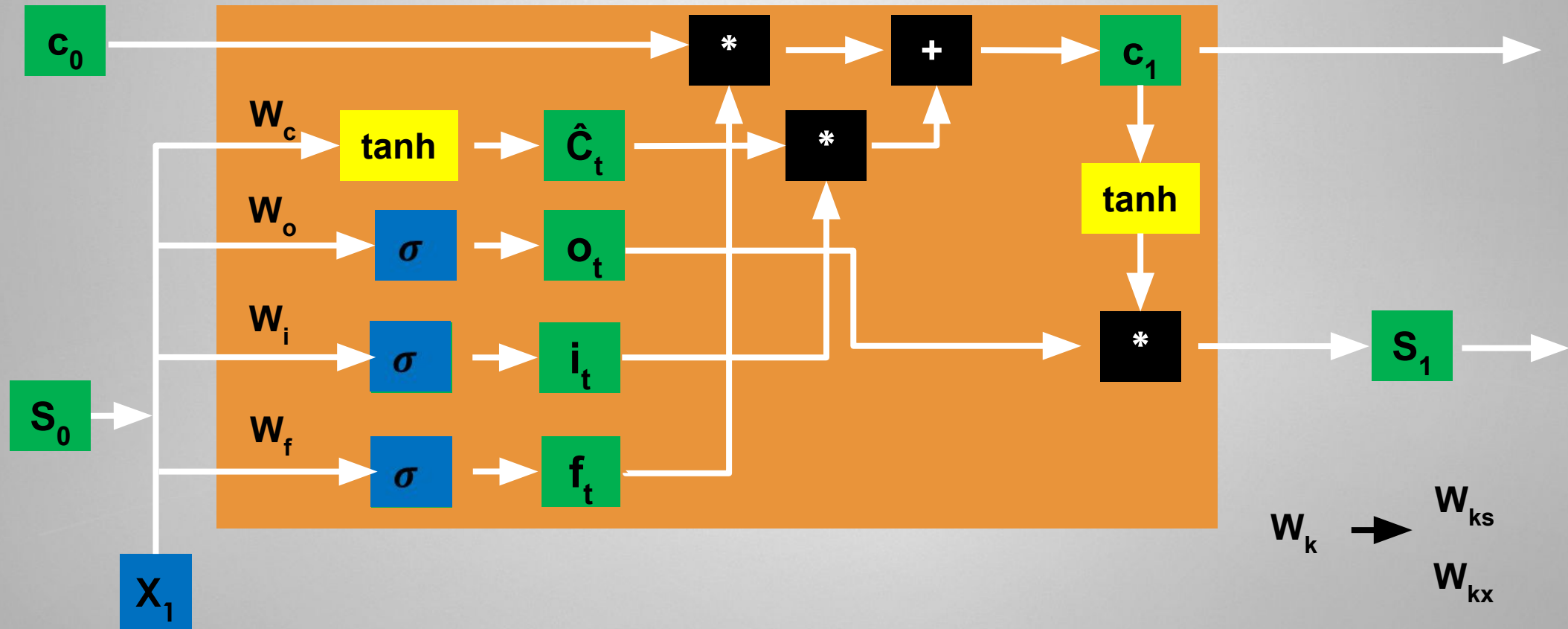
$c_t = (i_t * \hat{C}_t) + (f_t * c_{t-1})$  - cell state

$h_t = o_t * \tanh(c_t)$  - New State

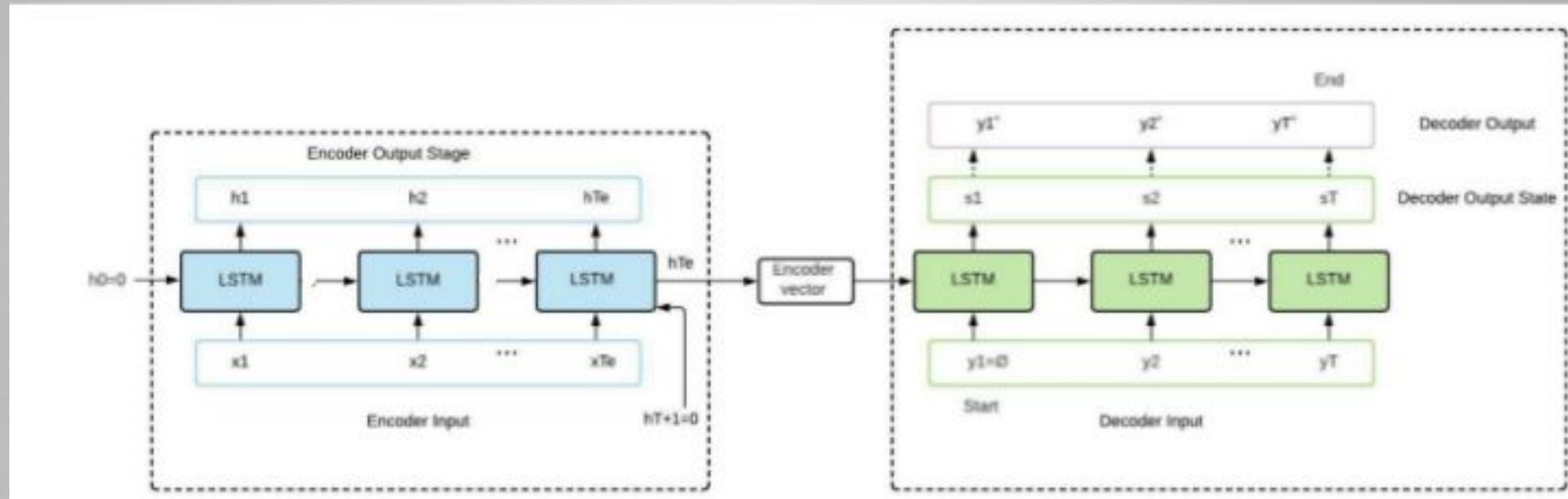


# LSTM

$$\begin{aligned}
 f_t &= \sigma(W_f S_{t-1} + W_f X_t) && \text{- Forget Gate} \\
 i_t &= \sigma(W_i S_{t-1} + W_i X_t) && \text{- Input Gate} \\
 o_t &= \sigma(W_o S_{t-1} + W_o X_t) && \text{- Output Gate} \\
 \hat{C}_t &= \tanh(W_c S_{t-1} + W_c X_t) \\
 c_t &= (i_t * \hat{C}_t) + (f_t * c_{t-1}) && \text{- cell state} \\
 h_t &= o_t * \tanh(c_t) && \text{- New State}
 \end{aligned}$$

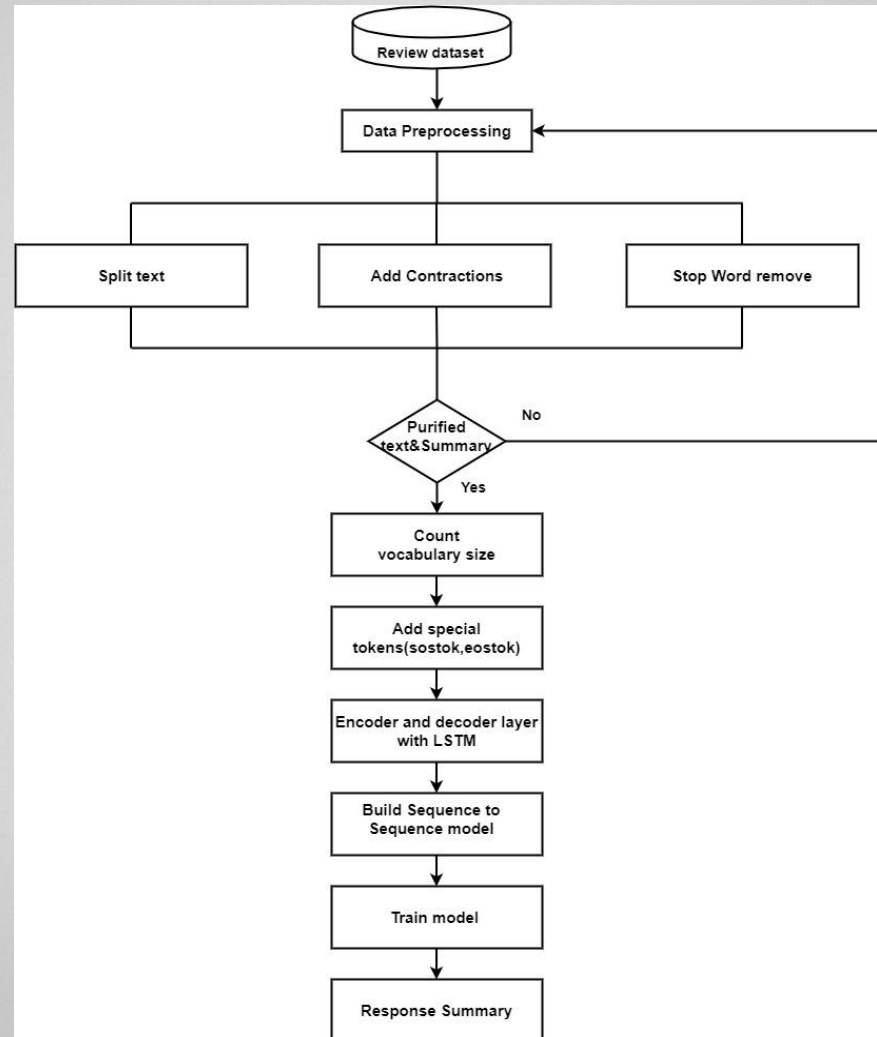


# DESIGN



Seq2Seq Model

# Flow graph of the Implementation



# Sample Input and Output

## Input:

REVIEW:

chips great delicious healthier  
regular deli chips first purchased  
super market found delicious found  
amazon cheaper ordered variety  
flavour flavours liking con addicting  
warned

Original Summary: delicious

## Output:

PREDICTED SUMMARY: great chips

```
#Preparing a Tokenizer for reviews on training data
X_tokenizer = Tokenizer(num_words=tot_cnt-cnt) #provides top most common words
X_tokenizer.fit_on_texts(list(X_train))
print(X_tokenizer.word_index)

#Converting text sequences into integer sequences
#print(X_train)
X_train_seq = X_tokenizer.texts_to_sequences(X_train)
print(X_train_seq)

X_test_seq = X_tokenizer.texts_to_sequences(X_test)

#Padding zero upto maximum length
X_train = pad_sequences(X_train_seq, maxlen = max_text_len, padding = 'post')
X_test = pad_sequences(X_test_seq, maxlen = max_text_len, padding = 'post')

#Size of vocabulary (+1 for padding token)
X_voc = X_tokenizer.num_words + 1

{'food': 1, 'great': 2, 'good': 3, 'dog': 4, 'product': 5, 'candy': 6, 'love': 7, 'twizzlers': 8, '
[[13, 2, 24, 24, 10], [2, 5, 25, 26], [14, 2, 15, 6, 16, 27, 17, 6], [13, 28, 3, 13, 18, 1, 24, 9],
```



# Output:

Review: love gluten free granola plain cranberry flavors great breakfast yogurt snack right bag definitely recommend product clearly says bag everything contain makes easier reading ingredi  
Original summary: delicious love them  
Predicted summary: great gluten free snack

Review: first time gave two little dogs hard treats real nicely next day noticed really excited opened package front keep mind treats really hard know large breeds smaller pups definitely t  
Original summary: even dog has  
Predicted summary: my dogs love these

Review: chips great delicious healthier regular deli store chips first purchased supermarket found delicious found amazon cheaper ordered variety flavor flavors liking con addicting warned  
Original summary: delicious  
Predicted summary: great chips

Review: gave product one zero option like generic tomato soup bits fish lobster soup shell salty fishy nasty inedible cans dented complete waste money beyond disappointing made  
Original summary: disgusting nasty  
Predicted summary: not bad

Review: taste okay away hidden fact truly decaffeinated coffee place find bottom box want first thing morning  
Original summary: just okay decaf  
Predicted summary: not bad

Review: got decaf keurig assortment pack enjoyed flavors weak post review  
Original summary: no flavor very weak  
Predicted summary: weak

# Bleu Score Results:

```
#cumulative BLEU scores
from nltk.translate.bleu_score import sentence_bleu
for i in range(0,1000):
    reference = seq2summary(y_train[i])
    candidate = decode_sequence(X_train[i].reshape(1, max_text_len))

print('Cumulative 1-gram: %f' % sentence_bleu(reference, candidate, weights=(1, 0, 0, 0)))
print('Cumulative 2-gram: %f' % sentence_bleu(reference, candidate, weights=(0.5, 0.5, 0, 0)))
print('Cumulative 3-gram: %f' % sentence_bleu(reference, candidate, weights=(0.33, 0.33, 0.33, 0)))
print('Cumulative 4-gram: %f' % sentence_bleu(reference, candidate, weights=(0.25, 0.25, 0.25, 0.25)))

Cumulative 1-gram: 0.35432
Cumulative 2-gram: 0.314802
Cumulative 3-gram: 0.285321
Cumulative 4-gram: 0.247890
```

```
#cumulative BLEU scores
from nltk.translate.bleu_score import sentence_bleu
for i in range(0,1000):
    reference = seq2summary(y_test[i])
    candidate = decode_sequence(X_test[i].reshape(1, max_text_len))

print('Cumulative 1-gram: %f' % sentence_bleu(reference, candidate, weights=(1, 0, 0, 0)))
print('Cumulative 2-gram: %f' % sentence_bleu(reference, candidate, weights=(0.5, 0.5, 0, 0)))
print('Cumulative 3-gram: %f' % sentence_bleu(reference, candidate, weights=(0.33, 0.33, 0.33, 0)))
print('Cumulative 4-gram: %f' % sentence_bleu(reference, candidate, weights=(0.25, 0.25, 0.25, 0.25)))

Cumulative 1-gram: 0.428571
Cumulative 2-gram: 0.389121
Cumulative 3-gram: 0.278128
Cumulative 4-gram: 0.262589
```

BLEU, or the Bilingual Evaluation Understudy, is a score for comparing a candidate translation of text to one or more reference translations. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0. NLTK provides the `sentence_bleu()` function for evaluating a candidate sentence against one or more reference sentences. An individual N-gram score is the evaluation of just matching grams of a specific order, such as single words (1-gram) or word pairs (2-gram or bigram). Cumulative scores refer to the calculation of individual n-gram scores at all orders from 1 to n and weighting them by calculating the weighted geometric mean. By default the `sentence_bleu()` calculate the cumulative 4-gram BLEU score called BLEU-4. The weights for the BLEU-4 are  $\frac{1}{4}$ (25%) or 0.25 for each 1-gram, 2-gram, 3-gram and 4-gram.

# WORK DONE

1. Studied and examined various papers and completed the literature work for this topic.
2. Inspected some popular text summarising web applications to understand the features they provide [10]
3. Briefly discussed the proper parameters, inputs and outputs of the model from our understanding.
4. Explored various methods to generate abstract summary for a given text.
5. Selected amazon-fine-food-reviews from kaggle website which is almost 294 megabyte with 560,000 reviews approx.
6. Performed Data Preprocessing which includes the cleaning the data, encoding the data by expanding all the contractions, and removing all the unwanted characters and obtained purified text and summary. Later the data is splitted into train and test data to learn effective mapping from inputs to outputs.



7. Added special tokens at the start of summary and end of summary to differentiate the given summary and text for better decoding.

8. Calculated the percentage of rare words and total coverage of rare words in the text.

9. Also calculated the maximum length of reviews and summary for making all the reviews length equal.

10. Padding zeros to the maximum length for both training and testing data has been done.

11. Implementation of the Model

12. Results from the implementation is discussed.

```
#Rarewords and their coverage in review
thresh = 4 #If a word whose count is less than threshold i.e 4, then it's considered as rare word

cnt = 0 #denotes no. of rare words whose count falls below threshold
tot_cnt = 0 #denotes size of unique words in the text
freq = 0
tot_freq = 0

for key,value in X_tokenizer.word_counts.items():
    tot_cnt=tot_cnt+1
    tot_freq=tot_freq+value
    if(value<thresh):
        cnt=cnt+1
        freq=freq+value

print("% of rare words in vocabulary:",(cnt/tot_cnt)*100)
print("Total Coverage of rare words:",(freq/tot_freq)*100)

% of rare words in vocabulary: 65.60676192392836
Total Coverage of rare words: 4.94040608149725
```

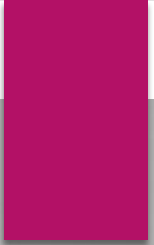
# FUTURE PLAN

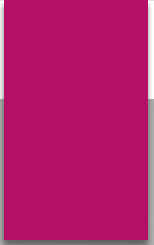
1. Making minor changes in the model to increase the accuracy of bleu score.
2. Designing the new model in-order to increase the accuracy .
3. Implementing the algorithm of the new model.
4. Finding the end results using bleu score and various other metrics.
5. Comparing the previous and improved design results to observe the difference.

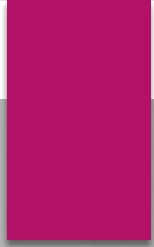
# REFERENCES

- [1] Dragomir R Radev, Eduard Hovy, and Kathleen McKeown. 2002. "Introduction to the special issue on summarization," *Computational linguistics* 28, 4 (2002).
- [2] <https://www.mordorintelligence.com/industry-reports/natural-languageprocessing-market>.
- [3] Mehdi Allahyari, Seyedamin Pouriyeh and Mehdi Assef "Text Summarization Techniques: A Brief Survey".
- [4] Saranyamol C S, Sindhu L, "A Survey on Automatic Text Summarization", *International Journal of Computer Science and Information Technologies*, 2014, Vol. 5 Issue 6.
- [5] C. L. Giles, G. M. Kuhn, and R. J. Williams, "Dynamic recurrent neural networks: theory and applications," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 153–156, 1994.
- [6] A. J. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 298–305, 1994.
- [7] Sutskever et al "Sequence to Sequence Learning with Neural Networks," *Conference on Neural Information Processing Systems (NIPS, 2014)*.
- [8] Abu Kaisar Mohammad Masum, Sheikh Abujar, "Abstractive method of text summarization with sequence to sequence RNNs," *IEEE* (2019)..



- 
- [10] <https://quillbot.com/summarize>
- [11] <https://github.com/google-research/google-research/tree/master/rouge>
- [12] <https://github.com/neural-dialogue-metrics/BLEU>
- [13] Hovy, E. and C.-Y. Lin. 1999. "Automated text summarization in SUMMARIST". In I. Mani and M. T. Maybury, editors, *Advances in Automatic Text Summarization*. MIT Press, Cambridge, pages 81–94.
- [14] <https://medium.com/@dusejashivam>
- [15] N. Raphal, H. Duwarah, and P. Daniel, "Survey on abstractive text summarization," in *Proceedings of the 2018 International Conference on Communication and Signal Processing (ICCSP)*, pp. 513–517, Chennai, 2018.
- [16] T. Shi, Y. Keneshloo, N. Ramakrishnan, and C. K. Reddy, *Neural Abstractive Text Summarization with Sequence-ToSequence Models: A Survey*, <http://arxiv.org/abs/1812.02303>, 2020.
- [17] *Deep Learning Based Abstractive Text Summarization: Approaches, Datasets, Evaluation Measures, and Challenges* by Dima Suleiman and Arafat Awajan
- [18] *Generating News Headlines with Recurrent Neural Networks* by Konstantin Lopyrev

- 
- [19] Abstractive text summarization using LSTM-CNN based deep learning by S Song, H Huang
- [20] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proceedings of the International Conference on Learning Representations, Canada, 2014
- [21] Z. Cao, F. Wei, W. Li, and S. Li, "Faithful to the original: fact aware neural abstractive summarization," in Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), New Orleans, LA, USA, February 2018.
- [22] <http://www2.psych.utoronto.ca/users/reingold/courses/ai/cache/neural4.html>
- [23] Widrow G, Hoff ME. Adaptive switching circuits. Institute of radio engineers, Western Electronic show and Convention, Convention record. 1960, part 4:96–104
- [24] McCulloch WS, Pitts WH. A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 1943; 5:115–133. KK The first mathematical model of logical functioning of brain cortex (formal neuron) is exposed in the famous work of McCulloch and Pitts
- [25] Introduction to Artificial neural networks by European Journal of Gastroenterology Hepatology (EUR J GASTROEN HEPAT), Jan 2008
- [26] <https://www.ibm.com/cloud/learn/neural-networks>



- [27] [https://www.educba.com/feedforward-neural-networks.](https://www.educba.com/feedforward-neural-networks)
- [28] <https://www.analyticsinsight.net/what-is-nlp-and-why-is-it-important/>
- [29] <https://towardsdatascience.com/why-nlp-is-important-and-itll-be-thefuture-our-future-59d7b1600dda>
- [30] <https://www.sas.com/ensg/insights/analytics/what-is-natural-languageprocessing-nlp.html>
- [31] <https://intellipaat.com/community/21886/advantages-and-disadvantagesof-neural-networks>
- [32] <https://www.asquero.com/article/advantages-and-disadvantages-ofartificial-neural-networks/>
- [33] <https://medium.com/analytics-vidhya/data-cleaning-and-preprocessinga4b751f4066f>
- [34] <https://searchbusinessanalytics.techtarget.com/definition/datavisualization>