Stock price prediction using Generative Adversarial Networks

Group 1:

Chen Chen, HungChun Lin

Outline

- Introduction
- Data description
  - Technical Indicator
  - Fourier Transformation
- Data preprocessing
- Model theory
- Model description
- Summary
  - Challenge
  - Future work

Introduction

- Long Short-Term Memory (LSTM)
  - A powerful method that is capable of learning order dependence in sequence prediction problems
- Generative Adversarial Networks (GAN)
  - One of the hot topic in deep learning nowadays
- Utilized GAN in financial area
  - High-frequency
  - Trading, portfolio optimization
  - Fraud detection
  - Risk management
- Project goal
  - Compare the basic LSTM model with GAN, and improve the model to get more accurate prediction.
- Contribution
  - Compare the different models
  - Improve GAN by adjusting the loss function
  - Input different features

Data Description

Data category	Description	Data Source
Target Data	Apple Stock price (Close)	https://finance.yahoo.com
Features	Similar company stock price (Amazon, Microsoft, Google)	https://finance.yahoo.com
	Stock Index (NASDAQ, NYSE, FTSE100, Nikkei225, BSE SENSEX, HENG SENG, SSE)	https://finance.yahoo.com
	Economic Index (Crude Oil, Gold, VIX, USD index)	https://fred.stlouisfed.org https://finance.yahoo.com

Data Description

Data category	Description	Data Source
Calculated Features	Daily news of Apple company	http://seekingalpha.com/
	Technical indicator (7 and 21 days moving average, Exponential moving average, Momentum, Bollinger bands, MACD)	
	Fourier transform (3, 6 and 9 components)	

- **2517 observations (2010 July - 2020 June)**
- **36 features**

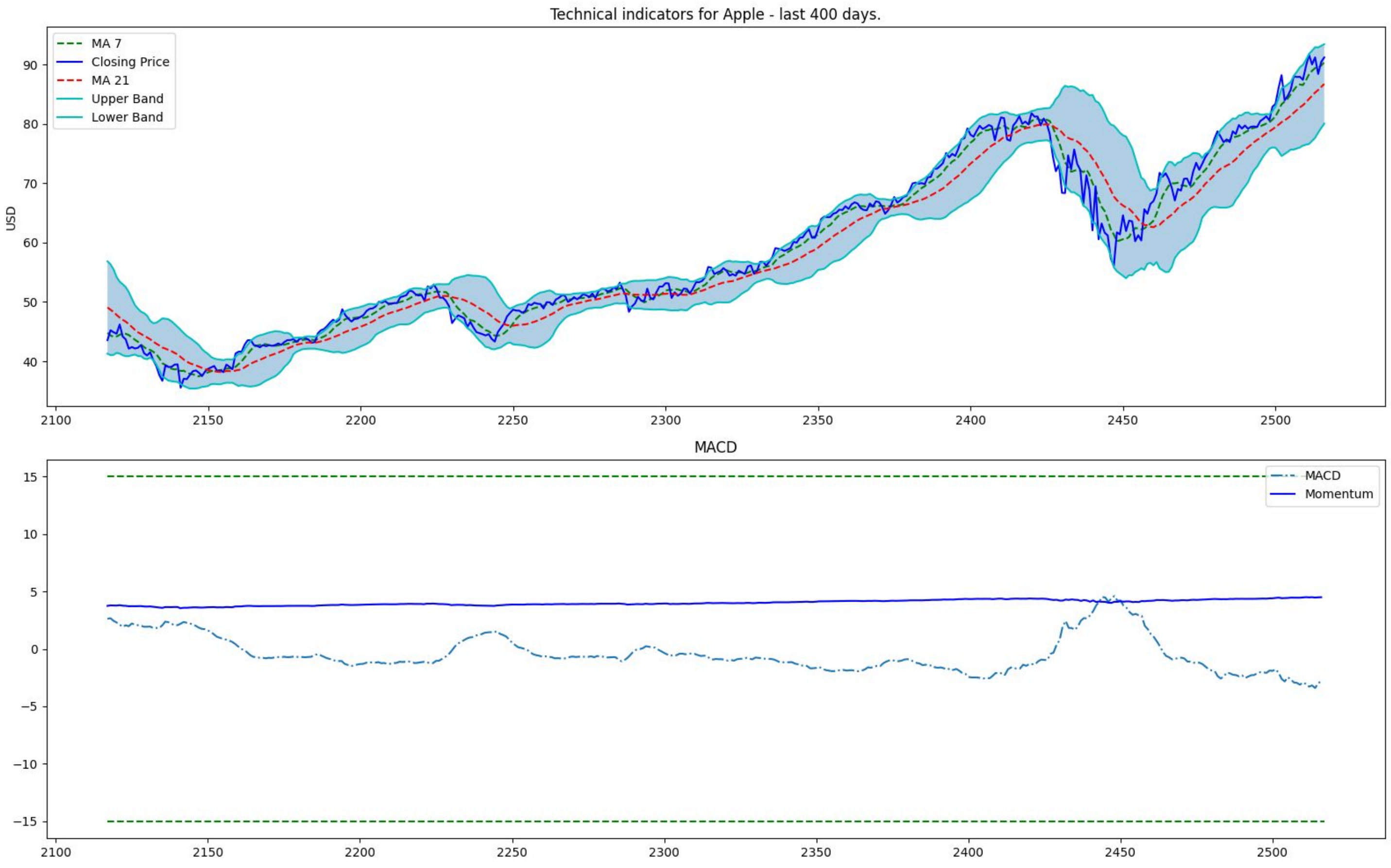
Calculated Features - News Sentiment Analysis

- **Finbert**
- **Giving score to the news [-1,1]**

Date	Sentimen Score	Article Title
2020/6/30	-0.918081701	Apple Arcade cancels games in strategy shift - Bloomberg
2020/6/30	-0.888130665	Shipment estimates cut for Apple's 5G iPhones - Digitimes
2020/6/29	-0.885745525	NYT pulls out of Apple News partnership
2020/6/29	-0.825985014	Apple leaving adapters out of iPhone 12 box - analyst
2020/6/27	0.847903252	Apple seen benefiting from chips play
2020/6/26	-0.084378615	DOJ's Apple probe focusing on App Store payment rules - Bloomberg
2020/6/25	-0.912210941	Apple re-closing 14 stores in Florida
2020/6/25	-0.709997535	Apple closes more stores amid spike in COVID-19 cases
2020/6/24	0.006058903	UBS reviews names to watch in 'consumerization of healthcare'

Calculated Features – Technical Indicator

- 7-day and 21 day Moving Average
- Moving Average Convergence Divergence (MACD)
- Exponential moving average (EMA)
- Momentum



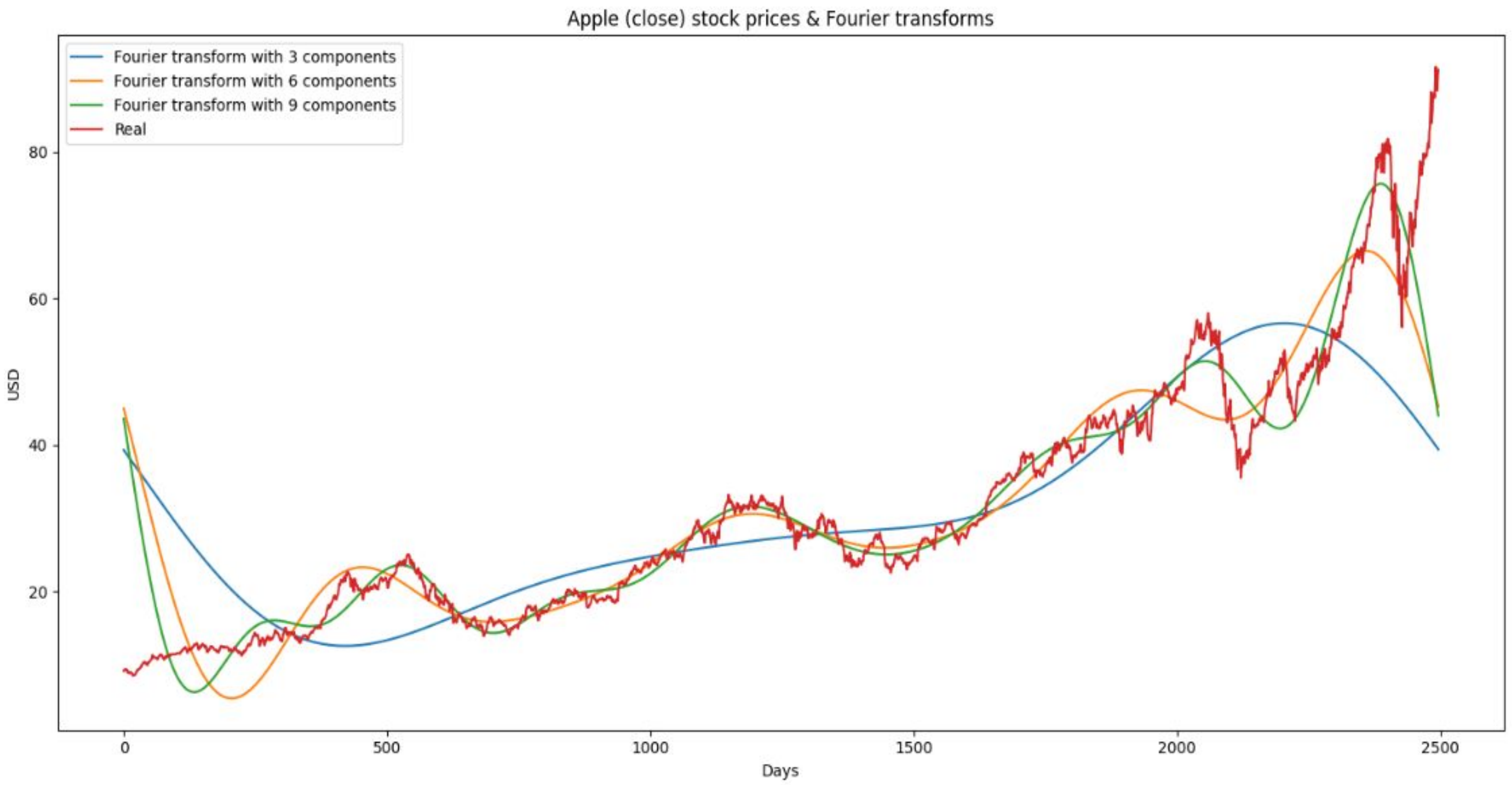
Reference: https://www.investopedia.com

Calculated Features – Fourier Transform

- Fourier variation extracted trend features of different frequency domains.

- Eliminate a lot of noise (random walks) and create approximations of the real stock movement.

- Help the LSTM network pick its prediction trends more accurately.



Data preprocessing

- NA value for index value

  $x_t = \text{mean}(x_{t-1}+x_{t+1})$

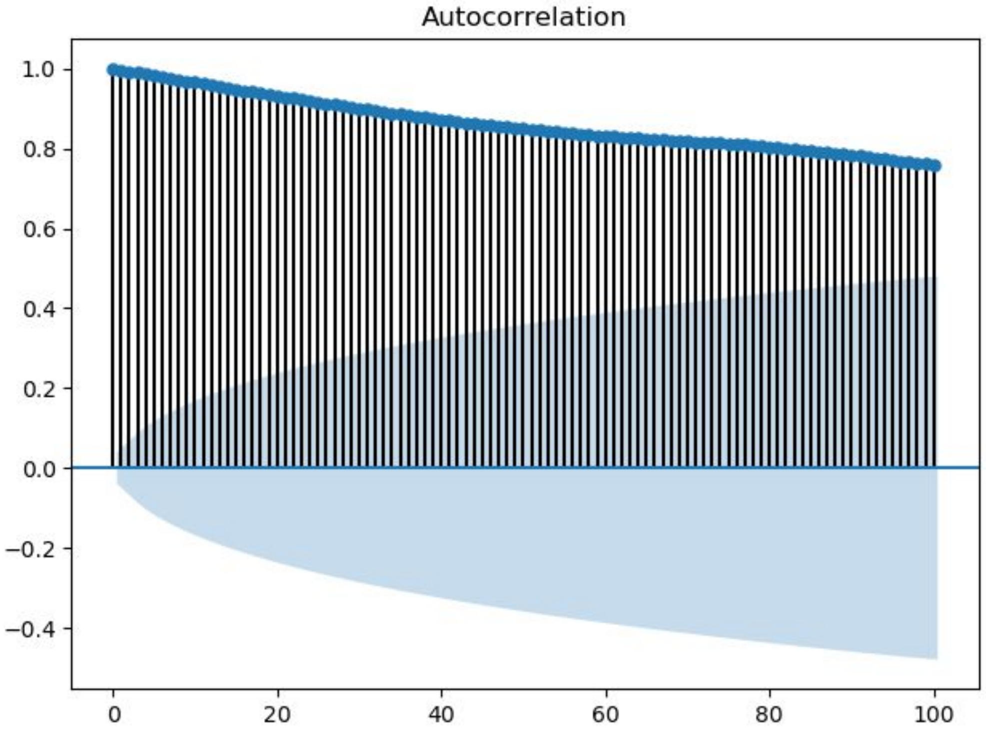
- Normalized the data to (-1,1)

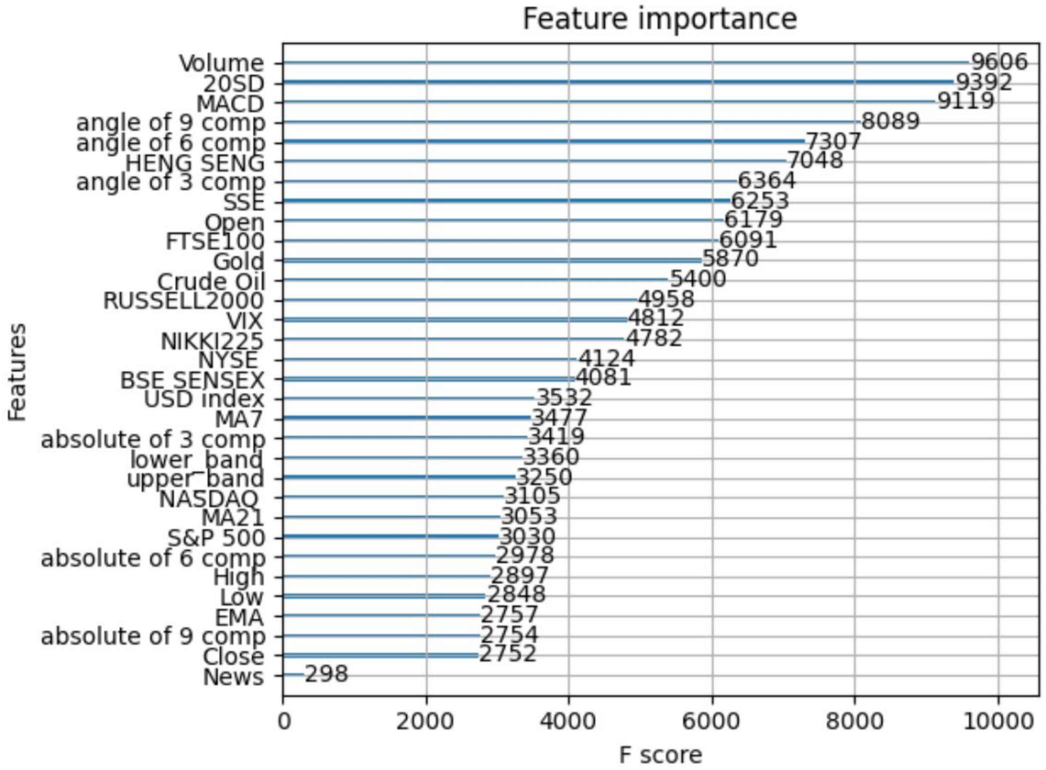
- Train test split (7:3)

  Train: 1732

  Test:743

Statistical Check



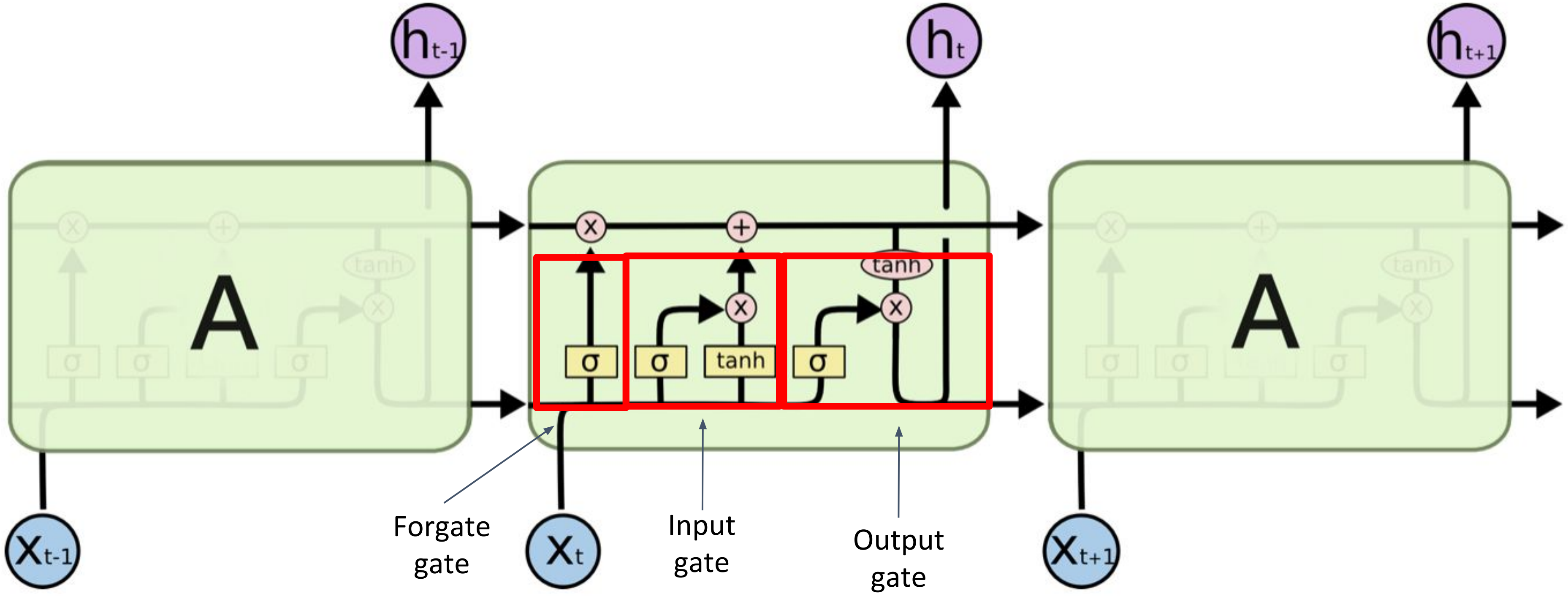


- Autocorrelation for target value
- previous stock price has influence when making predict, use previous close price as an feature in the model.

- XGBoost
- all selected features proved somewhat important so we won't exclude anything when training the model

Model theory

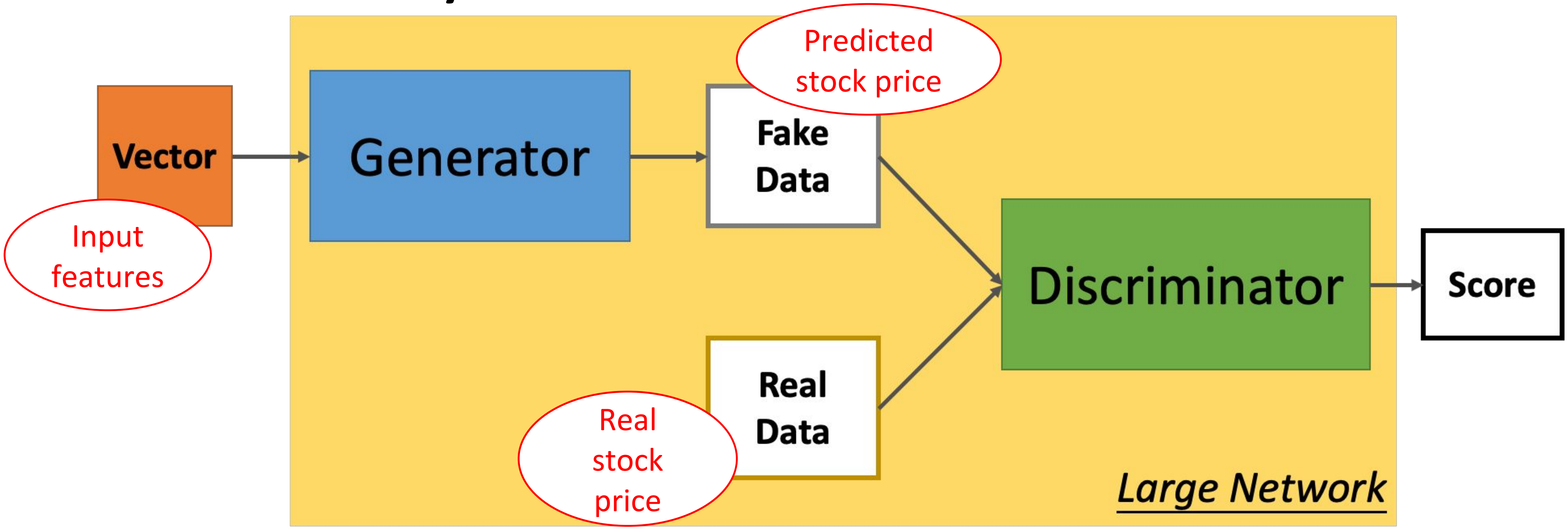
LSTM theory



- The basic components of LSTM are an input gate, an output gate and a forget gate
  - Forget gate: Decide what information we're going to throw away from the cell state.
  - Input gate: Decide what new information we're going to store in the cell state/ which values will be updated/ will be added
  - Output gate: Decide what we're going to output

Reference: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

GAN theory



- GAN basically made up of two competing neural network models
- The Generator generates fake data and tries to fool the Discriminator
- The Discriminator tries to distinguish between the real data and fake data
- This process will be repeated several times and the Generator and Discriminator will get better through this process

GAN Math

x : Input for generator
y : Real price from original data
$G(x^i)$: Generated price (fake price)

- *Learning D*

  Maximize *the objective function:*

  the smaller the better

  $$\hat{V} = \boxed{\frac{1}{m}\sum_{i=1}^{m} \log D\left(y^i\right)} + \sum_{i=1}^{m}(1 - \boxed{\log D(G\left(x^i\right)}))$$

  the bigger the better

- *Learning G*

  Minimize *the objective function:*

  $$\hat{V} = \frac{1}{m}\sum_{i=1}^{m}(1 - \boxed{\log D(G\left(x^i\right)}))$$

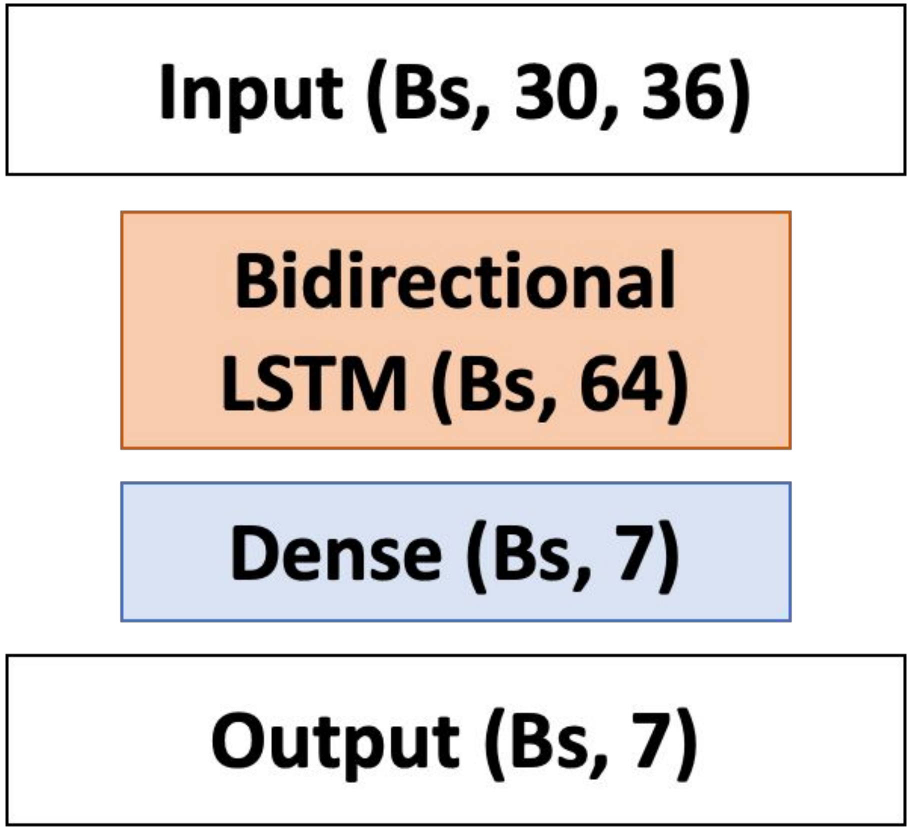
  the bigger the better

Model Implementation

- Baseline Model (Bidirectional LSTM)
- GAN Model (Using Bidirectional LSTM as a generator, and using CNN as a discriminator)

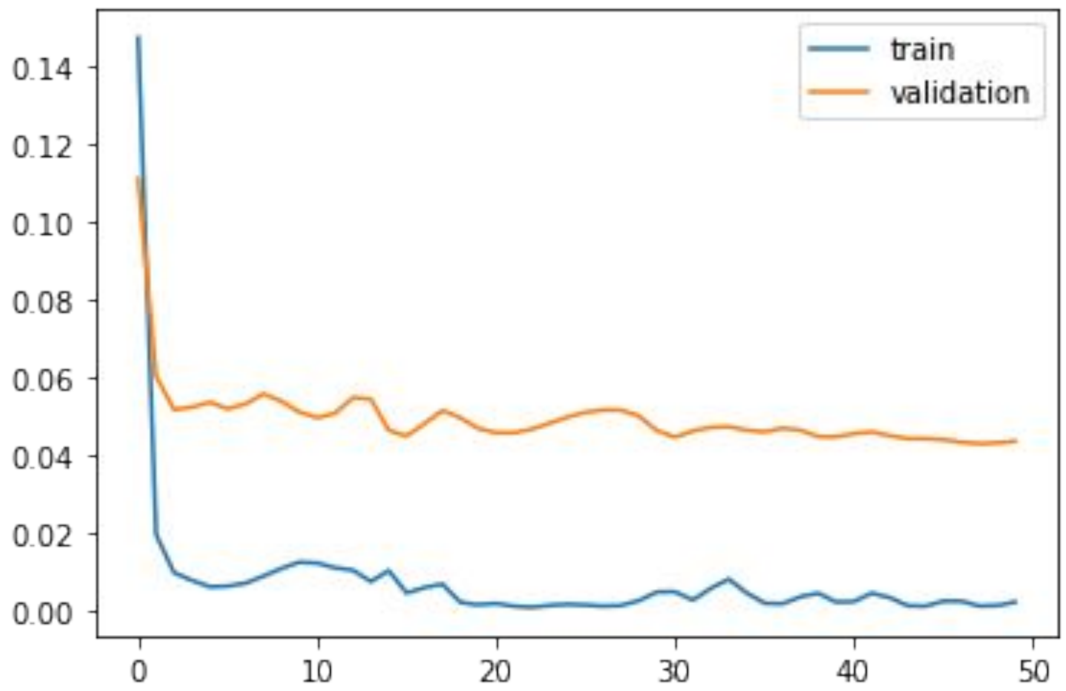
LSTM

Input_steps = 30
Output_steps = 7
Features = 36



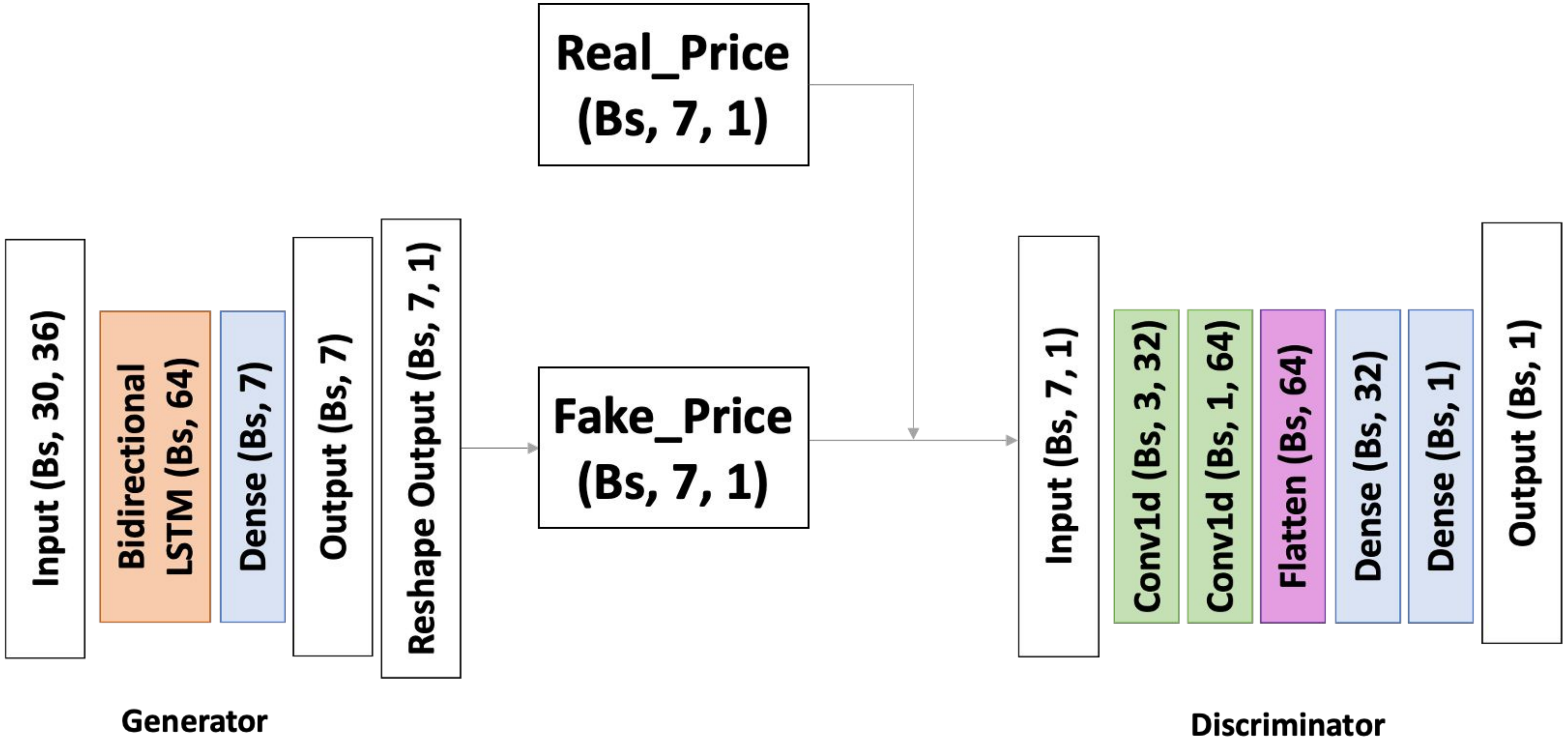
Bidirectional LSTM

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_2 (Bidirection (None, 128)               51712
_________________________________________________________________
dense_4 (Dense)              (None, 7)                 903
=================================================================
Total params: 52,615
Trainable params: 52,615
Non-trainable params: 0
```



RMSE: 2.75

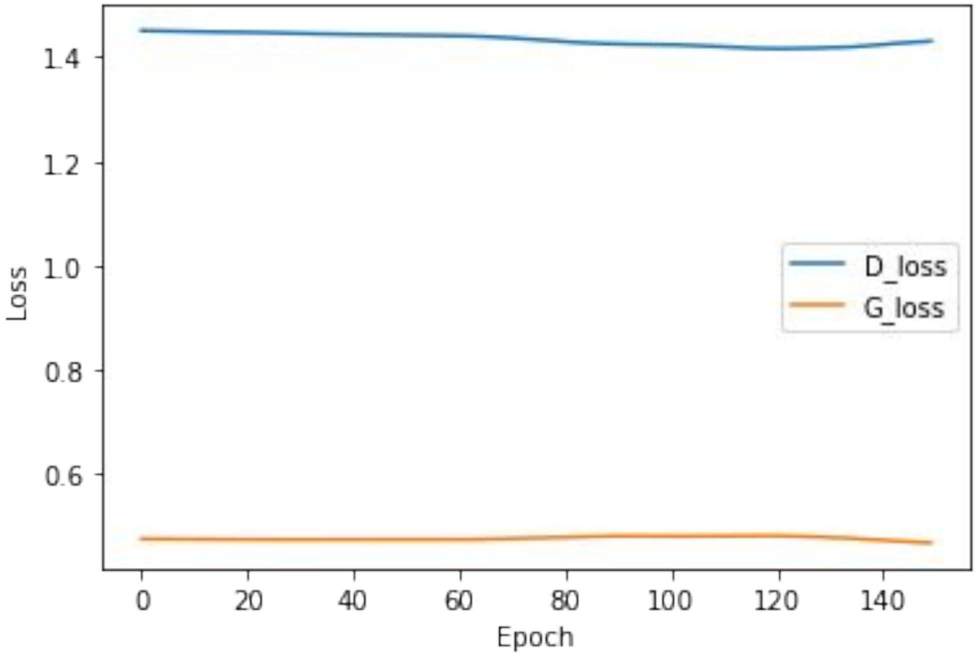
GAN Model


Real_Price (Bs, 7, 1)
Input (Bs, 30, 36)
Bidirectional LSTM (Bs, 64)
Dense (Bs, 7)
Output (Bs, 7)
Reshape Output (Bs, 7, 1)
Fake_Price (Bs, 7, 1)
Input (Bs, 7, 1)
Conv1d (Bs, 3, 32)
Conv1d (Bs, 1, 64)
Flatten (Bs, 64)
Dense (Bs, 32)
Dense (Bs, 1)
Output (Bs, 1)
Generator
Discriminator

GAN

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 64)	25088
dense (Dense)	(None, 7)	455

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 3, 32)	128
conv1d_1 (Conv1D)	(None, 1, 64)	6208
flatten (Flatten)	(None, 64)	0
dense_1 (Dense)	(None, 64)	4096
dense_2 (Dense)	(None, 32)	2048
dense_3 (Dense)	(None, 1)	33



RMSE: 4.91

Hyperparameter tuning

- Method: Bayesian optimization
    - learning rate
    - epoch
    - batch size

```
pbounds={'lr': (0.0001, 0.0008),
         'epoch': (100, 300),
         'bs': (64, 512)},
```

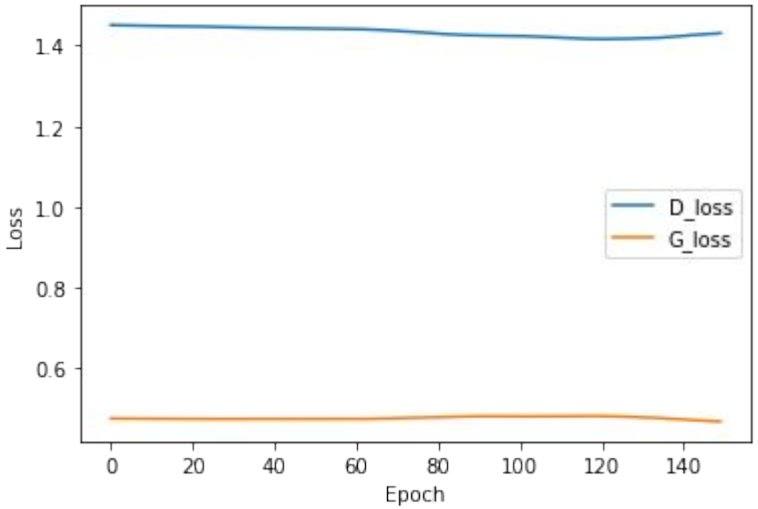
- strides
- kernel_size
- relu_alpha

Benchmark

	RMSE
Basic LSTM	2.75
GAN	4.91
WGAN	In progress
WGAN-GP	In progress

Challenge

- The loss in GAN training process looks not good (The G_loss should be larger than D_loss), it may indicate that our Discriminator is too weak
- Hyperparameter tuning through Bayesian optimization seems not improving our result
- The RMSE of GAN is still larger than basic LSTM



Future work

- Keep working on WGAN and WGAN-GP
- Adjust CNN model structure to improve the Discriminator
- Work on the Bayesian Optimization

Thank you

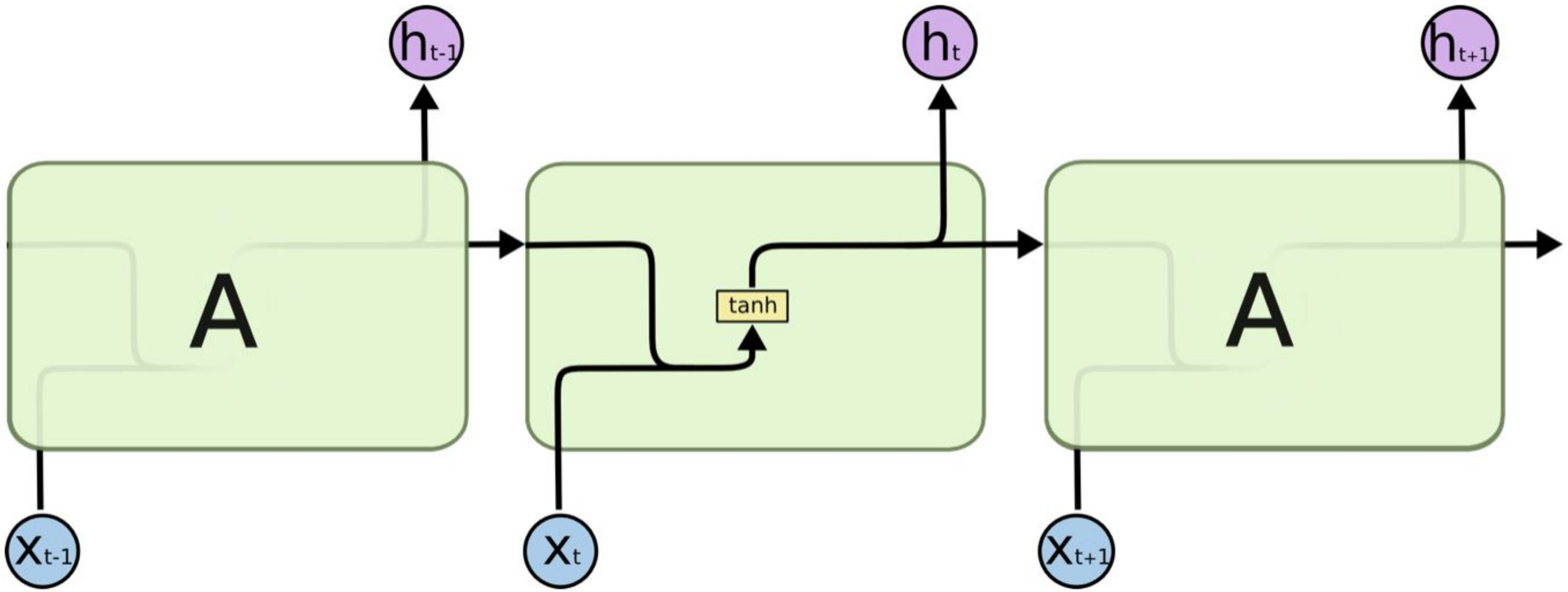
Appen

EMA equation

The Formula for EMA Is

$$EMA_{\text{Today}} = \left(\text{Value}_{\text{Today}} * \left(\frac{\text{Smoothing}}{1 + \text{Days}}\right)\right) + EMA_{\text{Yesterday}} * \left(1 - \left(\frac{\text{Smoothing}}{1 + \text{Days}}\right)\right)$$

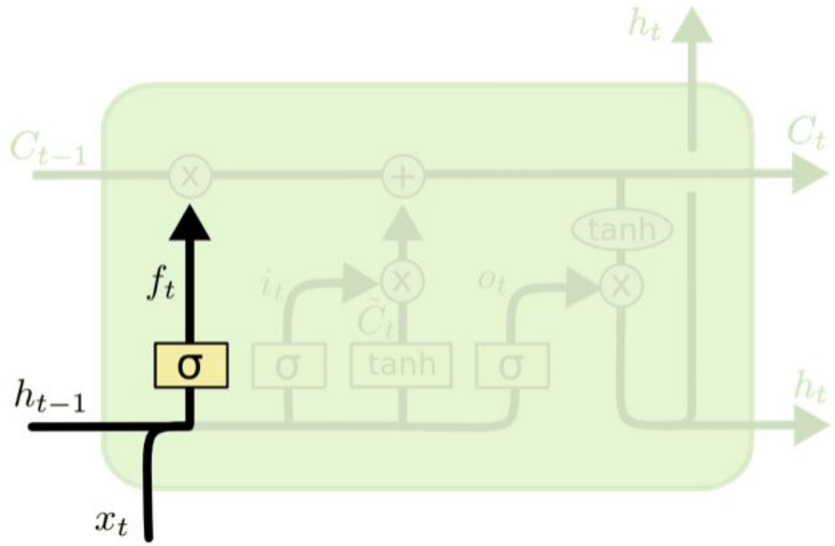
RNN



The repeating module in a standard RNN contains a single layer.

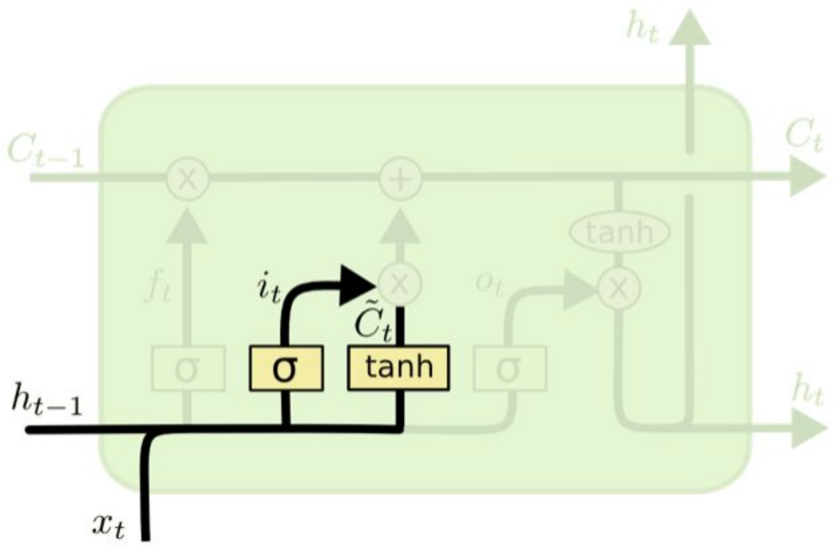
LSTM step-by-step

Forget gate: The first step in our LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer." It looks at ht−1 and xt, and outputs a number between 0 and 1 for each number in the cell state Ct−1. A 1 represents "completely keep this" while a 0 represents "completely get rid of this."



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

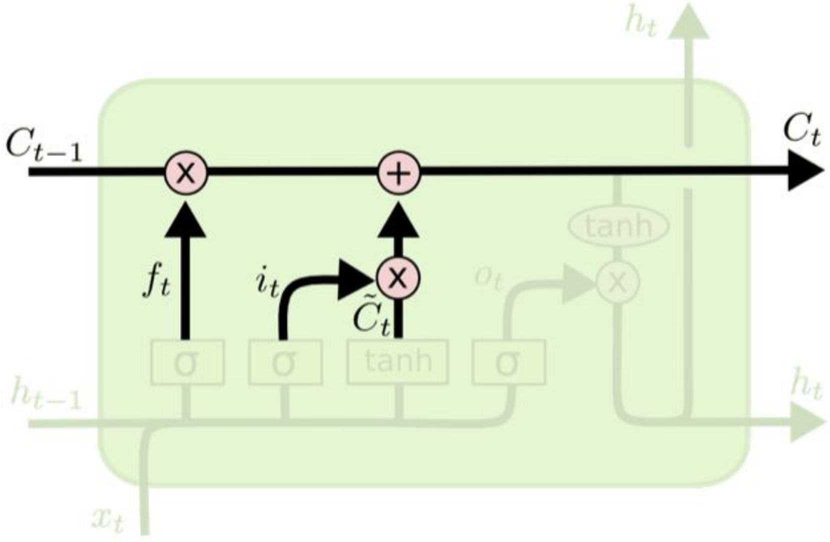
- The next step is to decide what new information we're going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values we'll update. Next, a tanh layer creates a vector of new candidate values, C̃ t, that could be added to the state. In the next step, we'll combine these two to create an update to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

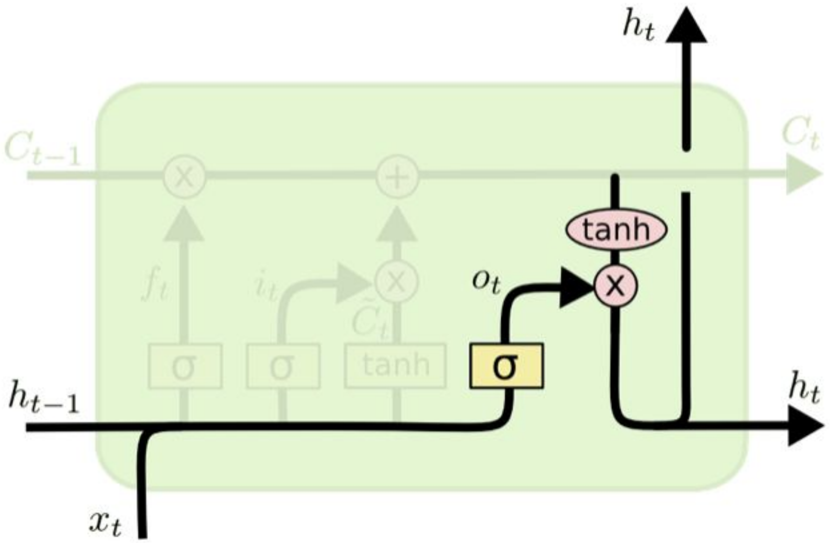
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

- It’s now time to update the old cell state, C_{t-1}, into the new cell state C_t. The previous steps already decided what to do, we just need to actually do it. We multiply the old state by f_t, forgetting the things we decided to forget earlier. Then we add $i_t * \tilde{C}_t$. This is the new candidate values, scaled by how much we decided to update each state value.



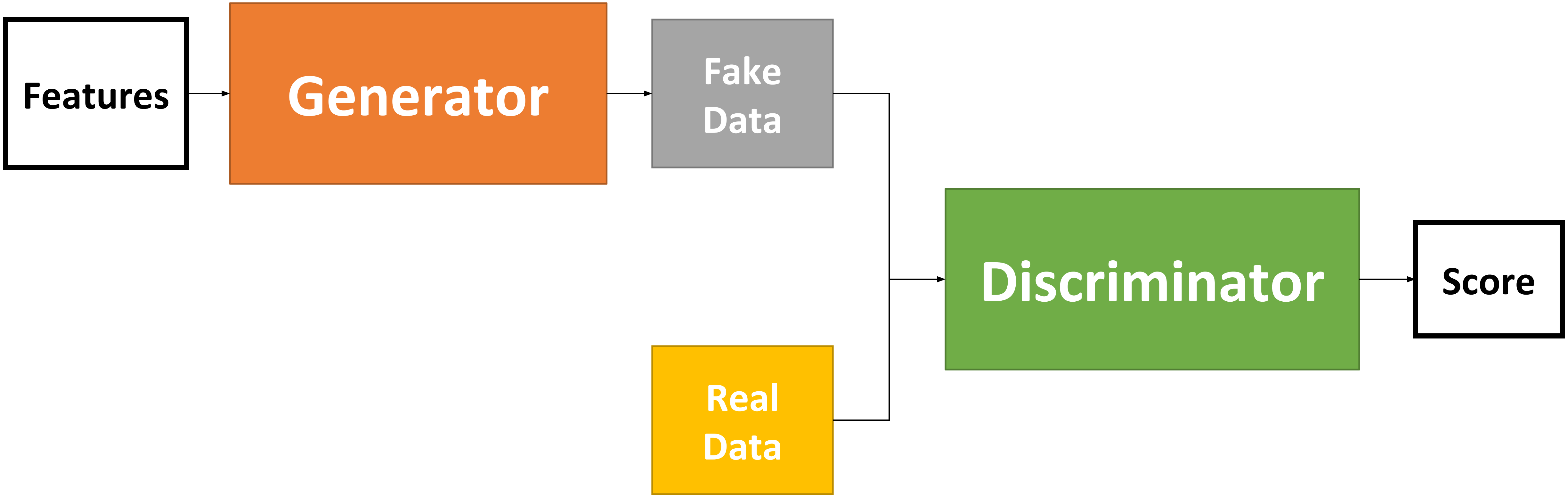
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

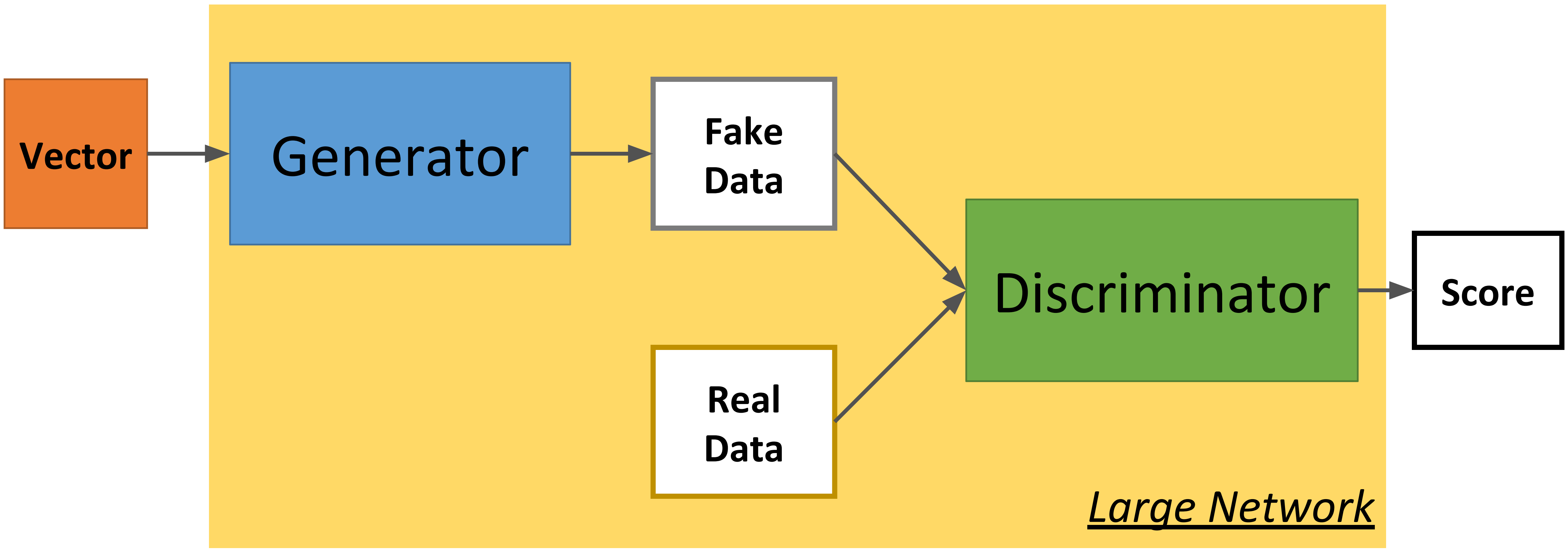
- Finally, we need to decide what we're going to output. This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.



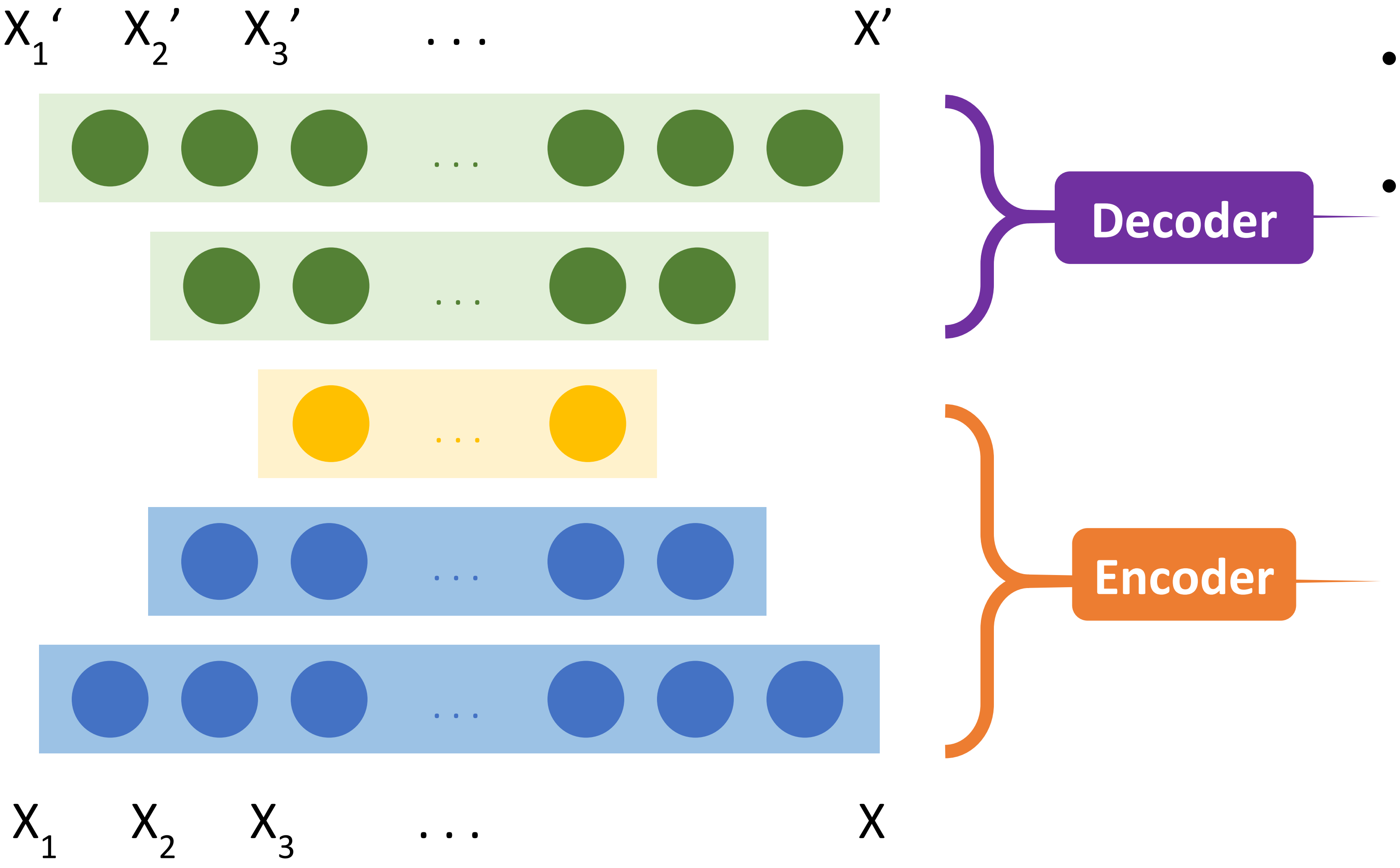
$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$


Features
Generator
Fake Data
Real Data
Discriminator
Score


Vector
Generator
Fake Data
Real Data
Discriminator
Score
Large Network

Calculated Features – Autoencoders



- For feature extraction
- The input number and output number will be the same

Data Description

Data category	Description
Stock price (Technology stocks)	10 years of Apple, Amazon, Microsoft, Google Yahoo finance: https://finance.yahoo.com
Stock index	NASDAQ, NYSE, FTSE100, Nikkei225, BSE SENSEX, HENG SENG, SSE
Economic index	Crude Oil, Gold, VIX, USD index FRED: https://fred.stlouisfed.org
Daily news	Daily news of Apple company Seeking alpha: http://seekingalpha.com/
Technical indicator	7 and 21 days moving average, Exponential moving average, Momentum, Bollinger bands, MACD
Fourier transform	3, 6 and 9 components
Autoencoder	Hidden features