Programming trading

with deep convolutional neural network

JINCHUN LU

6219710

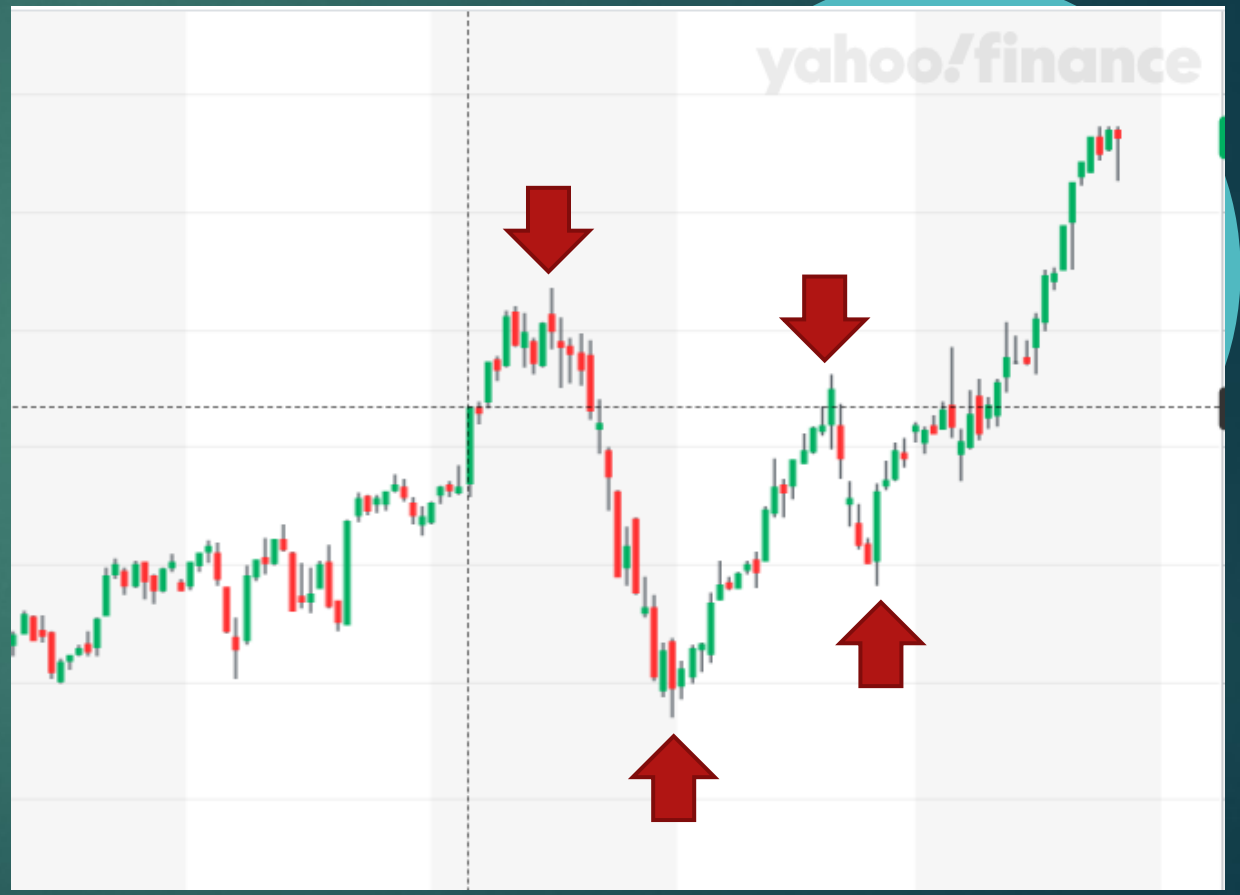
Stock market

- Stock Price changes every day
- Two strategies
  - Buy-and-hold
  - Buy and Sell
- Buy and sell

  Buy at lower point and then sell at higher point

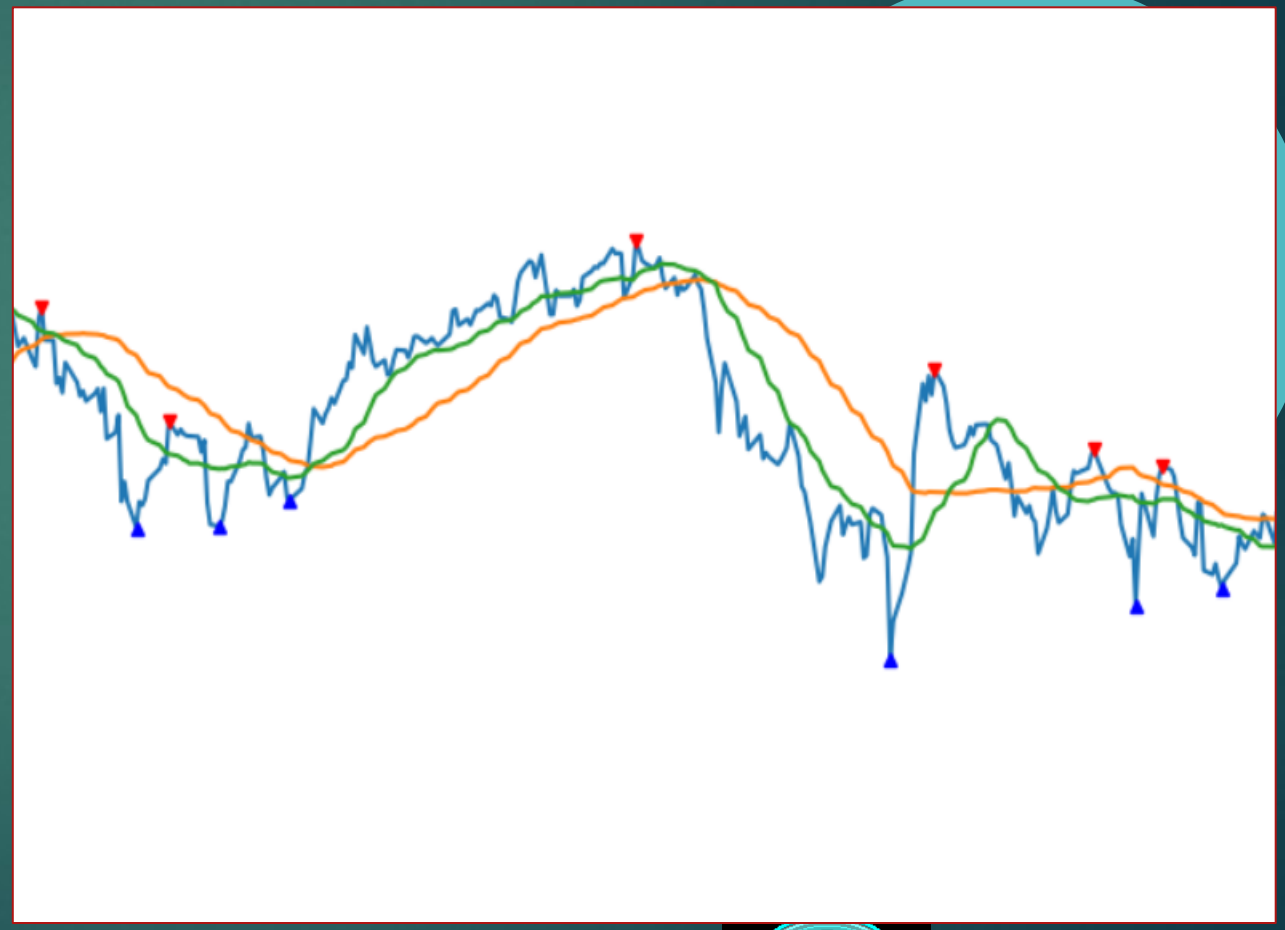
Methodology

- Idea
  - Find the day with a profitable chance that the stock may increase in the following days, and the day when its price may drop after.
- Objective
  - Try to find a model that can recognize profitable chances in the market
- Implementation
  - CNN



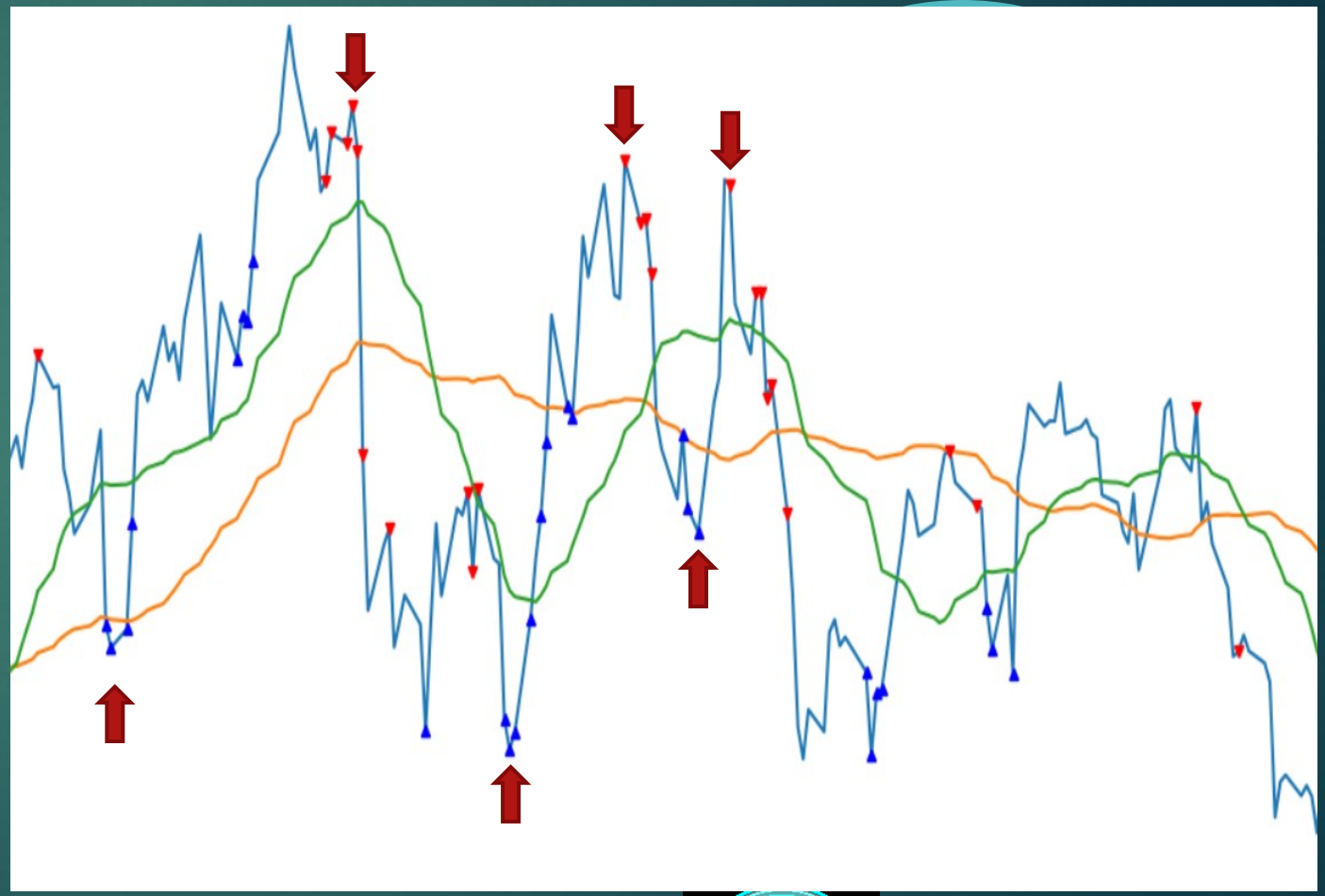
Data preparation

- Ten yeas of daily data of five stocks was used, 2456 data for each
- Data of three stocks are used as training data
- Data of two stocks are used as validating data



Label

- Three labels: Hold, Buy, Sell
- Set label as Buy if price increases more than 5% after five days
- Set label as Sell if price drops more than 5% after five days
- Set label as Hold if not Buy and Sell



Attributes

1. ***RSI*** measures the internal strength of a single stock.

$$RSI = 100 - \left[\frac{100}{1 + \left(\frac{U}{D}\right)}\right]$$

where U is the average of upward price changes and D is the average of downward price changes in certain days

2. ***PROC*** displays the difference between the current price and the price x-time periods ago

$$PROC = \left[\frac{C - C_x}{C_x}\right] 100$$

where C is the today close price of the stock and Cx is the close price of the stock before x days.

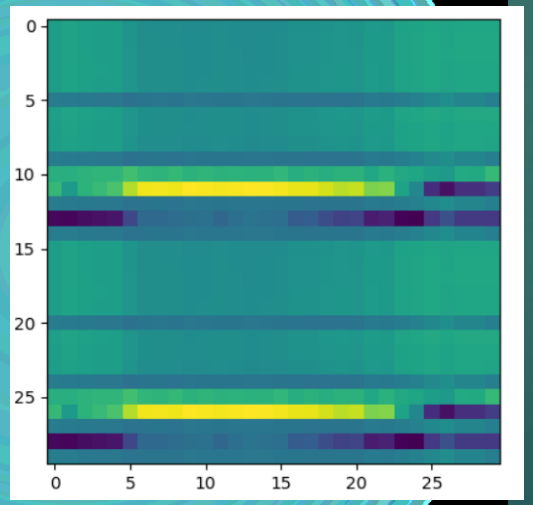
3. ***MACD*** is a trend-following momemtum indicator

$$\text{MACD} = \text{12-Period EMA} - \text{26-Period EMA}$$

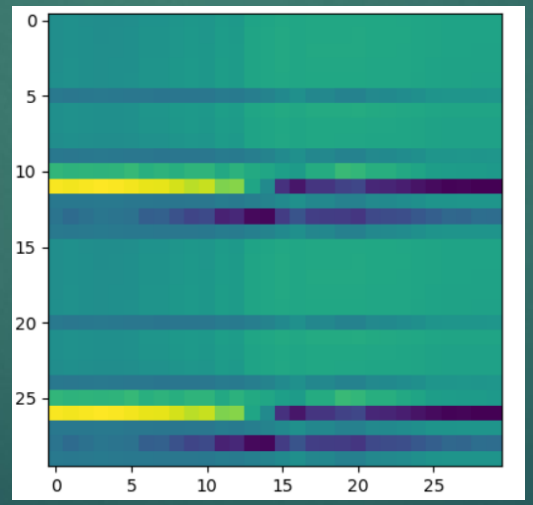
Construct Image for CNN

- We construct a 30*30 image for each day, 30 attributes of 29 days before the day and the day consist of the image.
- Example:

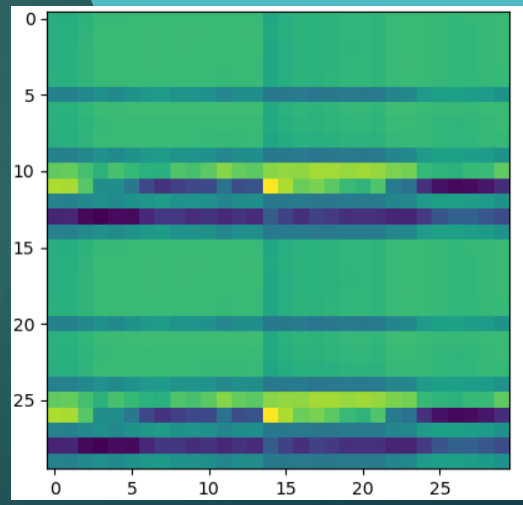
Hold



Buy



Sell



Training

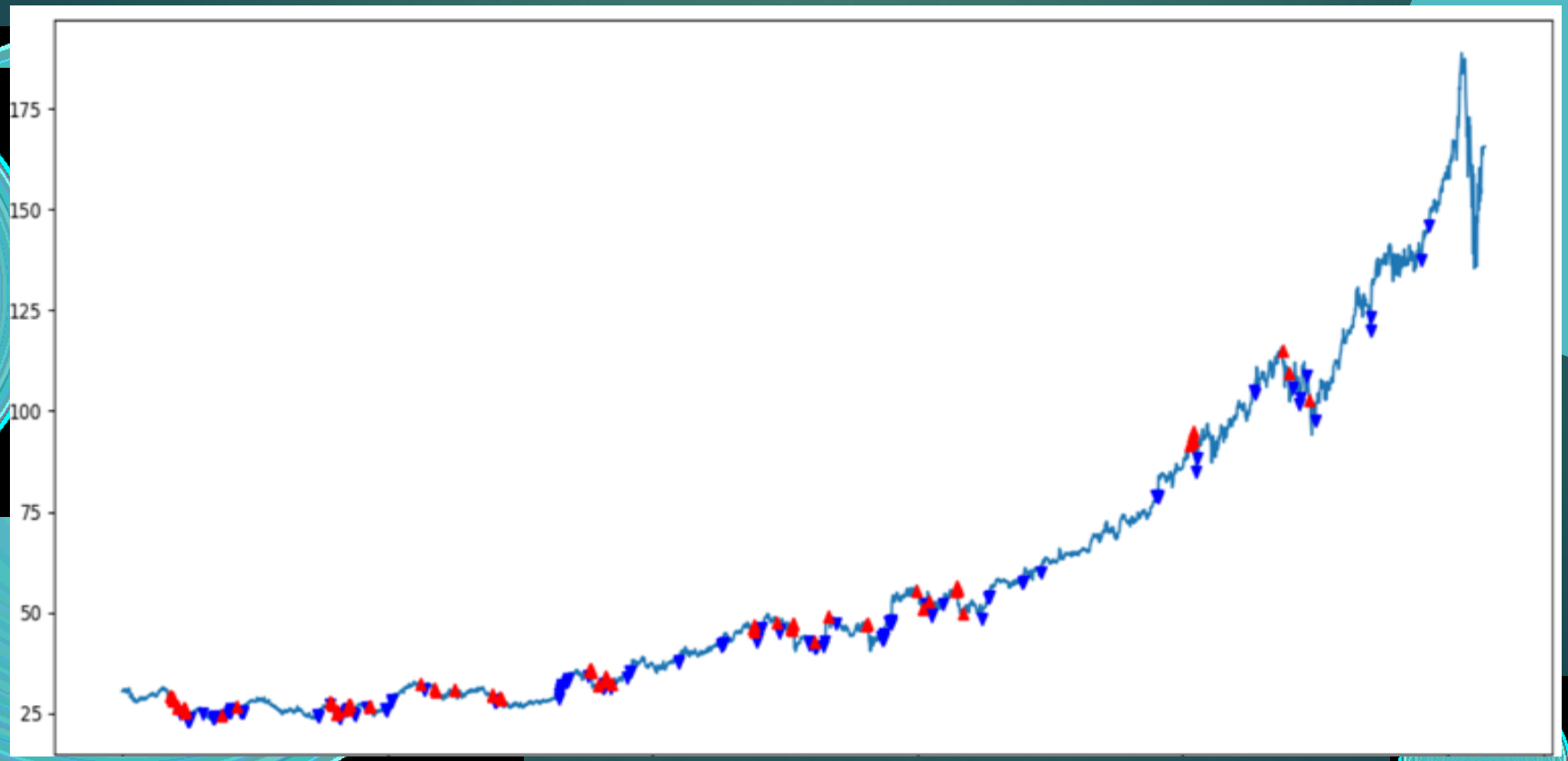
- Tensorflow CNN model

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),activation='linear',input_shape=(utils.IMG_SIZE,utils.IMG_SIZE,1),padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D((2, 2),padding='same'))
model.add(Conv2D(64, (3, 3), activation='linear',padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Conv2D(128, (3, 3), activation='linear',padding='same'))
model.add(LeakyReLU(alpha=0.1))
model.add(MaxPooling2D(pool_size=(2, 2),padding='same'))
model.add(Flatten())
model.add(Dense(128, activation='linear'))
model.add(LeakyReLU(alpha=0.1))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
```

Predicting

- Generate predicted label



Performance Evaluation

- Confusion Matrix
- Financial Evaluation

Confusion matrix of microsoft

Actual\Predicted	Hold	Buy	Sell
Hold	2217	25	15
Buy	48	92	0
Sell	40	0	60

Actual\Predicted	Hold	Buy	Sell
Hold	0.984	0.01	0.006
Buy	0.35	0.65	0
Sell	0.4	0	0.6

Financial evaluation of microsoft

Emulate trading based on the signal that CNN model predicts

Profit from CNN: 366%

Profit from buy-and-hold: 194%

Confusion matrix

Actual\Predicted	Hold	Buy	Sell
Hold	1196	184	95
Buy	292	136	25
Sell	352	81	41

Actual\Predicted	Hold	Buy	Sell
Hold	0.81	0.12	0.06
Buy	0.65	0.3	0.15
Sell	0.74	0.17	0.08

Financial evaluation

Emulate trading based on the signal that CNN model predicts

Profit from CNN: 237%

Profit from buy-and-hold: -51%

Conclusion

- Both two stocks got positive results.
- Both two stocks beat buy-and-hold strategy.

	Profit	Buy-and-hold	Difference
Microsoft	366%	194%	172%
Au	237%	-51%	288%

Future work

- Improve accuracy
- Construct different images to compare their performances

Thank you