Natural language processing Term Project1
Fine-Grained Sentiment Analysis on Financial Microblogs

一、Team member and Division work：

湯忠憲 資料科學碩一　R06946003 Sentiment Analysis MSE
劉宏國 資工碩一　R06922006 Sentiment Analysis MSE
陳奎伯 資工碩一　P06922001 Sentiment Analysis F1 3-class

二、Methodology：

Sentiment Analysis MSE：

The goal of Fine-Grained Sentiment Analysis is to output a sentiment score, instead of polarity, given a final microblogs. For this task, sentiment dictionary plays a crucial role in both dictionary-based and machine learning approaches. Our approach combines the sentiment dictionary and machine learning algorithm to predict a sentiment score of financial microblogs. The details are then illustrated:

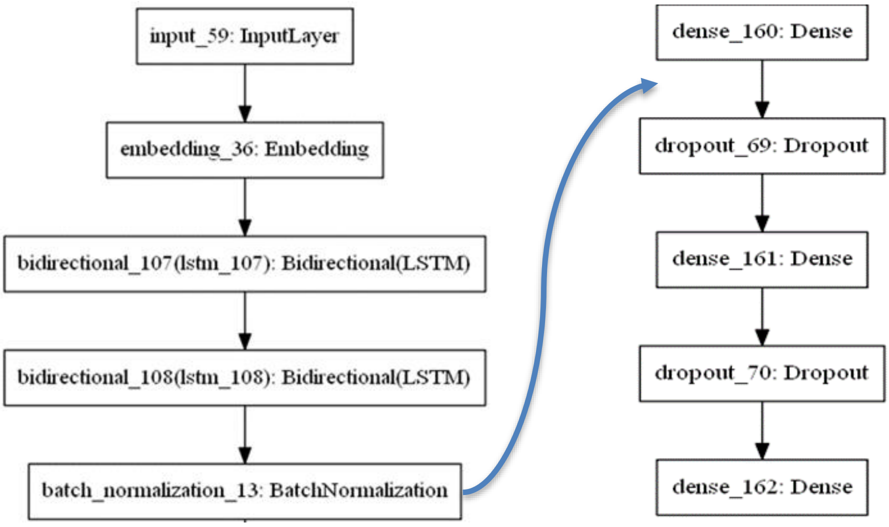
1. A given microblog sentence, tweet here, might have lots of noise, so we use stop word dictionary to filter out redundant words.
2. Then we leveraged several state-of-the-art sentiment dictionaries:
   a. NTUSD-Fin [1] [source]
   b. AFINN [2] [source]
   c. VADER (Valence Aware Dictionary and sEntiment Reasoner) [3] [source]
   d. SentiWordNet [4] [source]
3. Each of the above will output sentiment scores given a word. We average the output score for each word in a sentence with respect to different dictionaries. Therefore, each sentence would be represented 6-dim sentiment-score features (VADER and SentiWordNet provide negative and positive score.
4. We then feed those feature vectors to some machine learning algorithms. Linear regression was used for first trial. It can learn linear combination weights to fit the ground truth sentiment score. After that, we try XGboost [5] for non-linear features leaning.
5. Finally, mean square error is used for performance measurement.

Sentiment Analysis F1 3-class：

No	Class Name	Sentiment Score
1	Bullish	>0
2	Bearish	<0
3	Neutral	=0

[1] http://lrec-conf.org/workshops/lrec2018/W27/pdf/1_W27.pdf
[2] https://finnaarupnielsen.wordpress.com/2011/03/16/afinn-a-new-word-list-for-sentiment-analysis/
[3] https://goo.gl/ZXD2XS
[4] http://nmis.isti.cnr.it/sebastiani/Publications/LREC10.pdf
[5] https://github.com/dmlc/xgboost

對於此分類問題，我們嘗試用基本的 RNN (Bi-LSTM) with an Embedding layer 來實驗模型架構圖如下：


input_59: InputLayer
embedding_36: Embedding
bidirectional_107(lstm_107): Bidirectional(LSTM)
bidirectional_108(lstm_108): Bidirectional(LSTM)
batch_normalization_13: BatchNormalization
dense_160: Dense
dropout_69: Dropout
dense_161: Dense
dropout_70: Dropout
dense_162: Dense

Processing (主要實驗項目)

處理資料主要分為兩個部分：第一部分為 Word Level：將原始文字中先用 space 進行斷詞之後，去除內含「http」、「$」、「@」等的整個詞，因為觀察這些詞彙對於判別情緒並無相關聯，然後再檢查這些詞是否有在英文常見 stopword 列表中，例如：「for」、「do」等並將之移除；第二部分為 Char Level：將第一部分處理後的詞集轉為字串，檢查字串中的每個字元是否有在 stopword 列表中出現，例如「$」、「@」等，將這些字元移除。最後剩下的字元全部轉換為小寫，並重新以 space 進行斷詞。

Gensim Word2Vec

將每個詞轉換至 300 維度的向量空間，且每個詞與它前後 5 個詞有相關聯，並建立起所有出現過的詞集合對應的索引典，索引典共含 2464 個詞。另外由於每個句子長度不一樣，故先計算出最大句長 22，將所有句子的 embedding 拉（補 0）至等長。

Labeling

依照每個 Tweet 對應的 Sentiment Score，將其分類至 3 種 class 的其中之一。

RNN Training

主要參數：lstm1-units(128)、lstm2-units(64)、lstm-dropout(0.2)、dense-dropout(0.5)、batch-size(64)、epoch(10)、optimizer(adam)、loss(categorical_crossentropy)。預測出結果是三個分類的個別機率 (sum=1)，取機率最大的分類視為預測值。

三、Evaluation：

Sentiment Analysis MSE：

We use the Mean squared error to evaluate our model [6].

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

$\hat{Y}$：a vector of n predictions.

Y：the ground truth vector

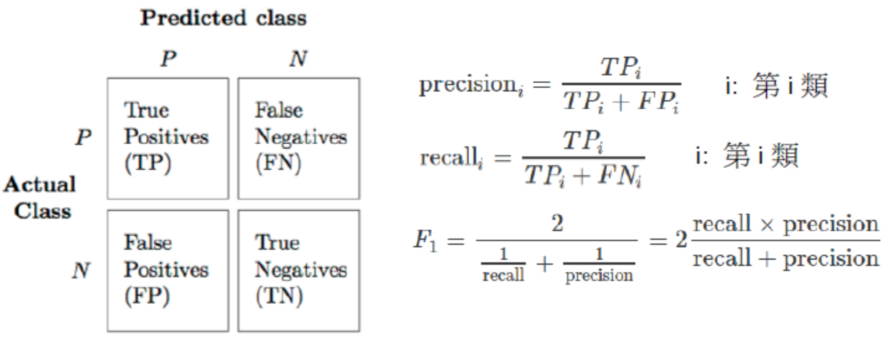
[6] https://en.wikipedia.org/wiki/Mean_squared_error

Results：

1. By using only NTUSD-Fin dictionary:
   testing MSE: 0.2615
2. By using Linear regression：
   testing MSE: 0.1045
3. By using XGboost：
   Depth: 7; learning rate: 0.1; n_estimator 200;
   testing MSE: **0.0832**

Sentiment Analysis F1 3-class：

由於是多分類問題，不能採用二元 F-Measure，而是採用 Micro-F1 以及 Macro-F1。



		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

$$\text{precision}_i = \frac{TP_i}{TP_i + FP_i}$$

i: 第 i 類

$$\text{recall}_i = \frac{TP_i}{TP_i + FN_i}$$

i: 第 i 類

$$F_1 = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2\frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

Figure 1. Confusion Matrix and F-measure

Table 1. Computation of Macro-F1 and Micro-F1

Macro-F1	$\text{precision}_{ma} = \frac{\text{precision}_1 + \text{precision}_2 + \text{precision}_3}{3}$ $\text{recall}_{ma} = \frac{\text{recall}_1 + \text{recall}_2 + \text{recall}_3}{3}$ $F_{1,ma} = 2\frac{\text{recall}_{ma} \times \text{precision}_{ma}}{\text{recall}_{ma} + \text{precision}_{ma}}$
Micro-F1	$\text{precision}_{mi} = \frac{TP_1 + TP_2 + TP_3}{TP_1 + FP_1 + TP_2 + FP_2 + TP_3 + FP_3}$ $\text{recall}_{mi} = \frac{TP_1 + TP_2 + TP_3}{TP_1 + FN_1 + TP_2 + FN_2 + TP_3 + FN_3}$ $F_{1,mi} = 2\frac{\text{recall}_{mi} \times \text{precision}_{mi}}{\text{recall}_{mi} + \text{precision}_{mi}}$

本次 1396 筆訓練資料經過分類後，樣本數並不平均，Bullish 類占了 6 成以上。所以採用 Micro-F1 來評估較佳，因為 Macro-F1 未考慮各類別樣本大小。

此模型預測 100 次結果的平均 Micro-F1 為 0.7071，Macro-F1 為 0.2626。

四、Discussion：

Sentiment Analysis MSE：

Table 2. Feature importance

Feature	NTUSD-Fin	Senti pos	Senti neg	VADER pos	AFINN	VADER neg
Info gain	**0.3256**	0.2545	0.1810	0.093	0.077	0.067

To get an insight into works of those dictionaries, we compute the feature importance of each sentiment dictionary when training XGboost. The above table indicates that NTUSD-Fin, an innovative and powerful dictionary, did a great job in this task. It's not beyond our expectation since the dictionary was designed for financial domain and made by outstanding researcher. For more information, a little experiment was conducted.

We gave each dictionary some words and observed their sentiment score. The results are shown below:

Table 3. Experiment results

Word	Senti pos	Senti neg	NTUSD-Fin	AFINN	VADER neg	VADER pos
buy	0	0	0.596	0	0	0
downgrade	0	0	-1.071	0	0	0
competition	0	0	-1.095	0	0	0

From the above results, we can find out that NTUSD-Fin provides polarity scores for those words, which are treated as nature word in other dictionary.

Sentiment Analysis F1 3-class：

Table 4. 不同 Preprocessing 方式對應的平均預測 Micro-F1

Preprocessing	**Micro-F1**
Word Level + Char Level	0.7071
Only Word Level	0.6536
Only Char Level	0.6643
None	0.6214

觀察實驗數據，發現移除 Char Level Preprocessing，對 performance 影響較大，原因為這些 stopword 包含的字元，會使得 Word2Vec 模型在建立索引典時多出約 1000 多個字彙量，然而這些字彙其實只是因為舊字彙包含 stopword 而被視為新字彙，造成 Word2Vec 訓練的誤差。

五、Conclusion：

對情緒分數的預測，我們結合了網路上現成的情緒字典，其中包括 SentiWordNet、NTUSD-FIN、AFINN 等字典。而就以結果來說，把每個字典所算得的分數加總算平均，再透過 XGBoost 預測情緒分數，會比單一字典直接預測情緒分數的 mse 還要來得好。另外我們也有比較單一字典是否能預測特定的 word，像是 buy 或是 downgrade 等等，只有 **NTUSD-FIN** 能夠算出正面或是負面分數，其他的則沒辦法得到正確分數。

關於 3-class 的模型預測，preprocessing 的步驟很重要，大約可將 Micro-F1 提升 8 個百分點，若對於分類樣本數不平均的問題有做額外的處理，例如對 Bearish 及 Neutral 類做 Over-sampling，或對 Bullish 類做 Under-sampling，甚至是可以蒐集更多 Tweet 資料使得各類之間的比例較為平均，將會有更好的效能。