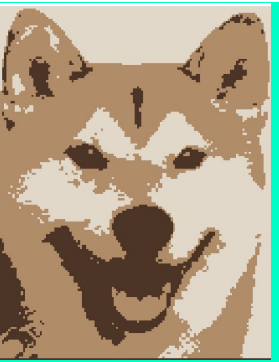
Real or Not? NLP with Disaster Tweets



SHIBA-INU

Xumin Cai
Kiril Kuzmin
Chenyu Wang
Chan Aek Panichvatana
Pragna Reddy Kancharla
Krishna Mani Teja Katragadda

4/28/2020

Contents

The Problem

- Given: a **text of a tweet** and some metadata (optional)

  Metadata:

  - Location
  - Keyword

- Find: if the tweet talks about real disaster
- TRAIN SIZE: 7613 TEST SIZE: 3263

Motivations

- Monitor the situation in a real time
  - Quick
  - Inexpensive
  - Safe
- Predict a follow-up process
- Real situation from contradicting tweets
- Categorization of situational and conversational data is important

Pre-Processing Data

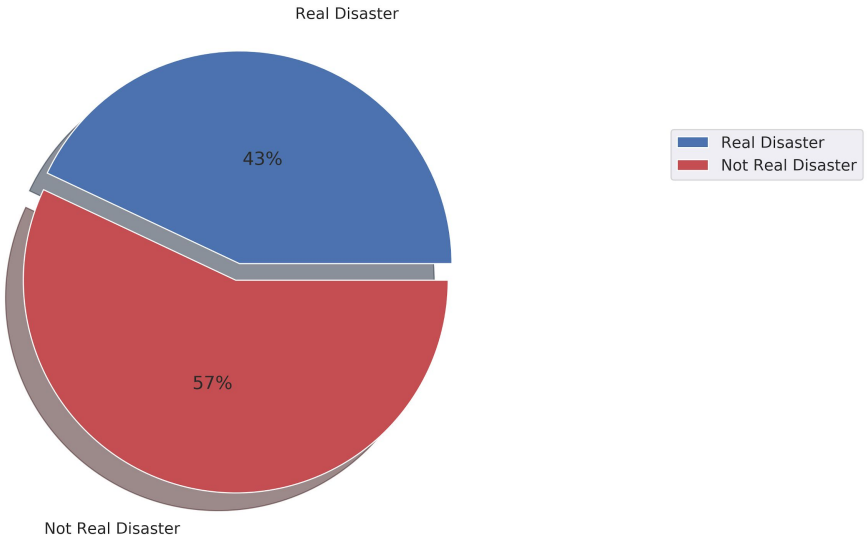
- Convert text to lowercase
- Remove stopwords
- Remove punctuation
- Remove digits
- Text vectorization (TF-IDF transformation)

	id	keyword	location	text	target
90	132	accident	NaN	???? it was an accident http://t.co/Oia5fxi4gM	0
91	133	accident	New Hanover County, NC	FYI CAD:FYI: ;ACCIDENT PROPERTY DAMAGE;WPD;160...	1
92	134	accident	NaN	8/6/2015@2:09 PM: TRAFFIC ACCIDENT NO INJURY a...	1

affect	aftershock	air	airplan	airplaneaccid	airport	...	wreck	wreckag	year	yet	your	youtub
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.00000	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.29624	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.00000	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.00000	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.00000	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.00000	0.0	0.0	0.0

Data Re-Exploration

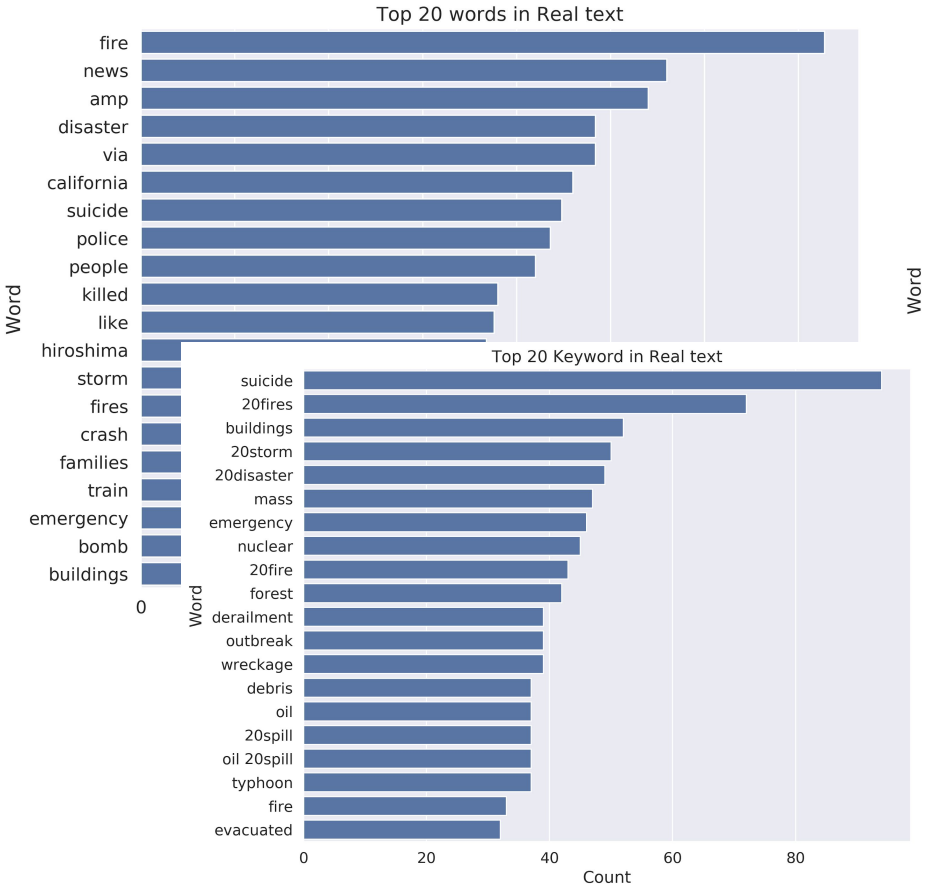
- Frequent words analysis
- Hyperlink Analysis
- Vocabulary Embedding Coverage Analysis

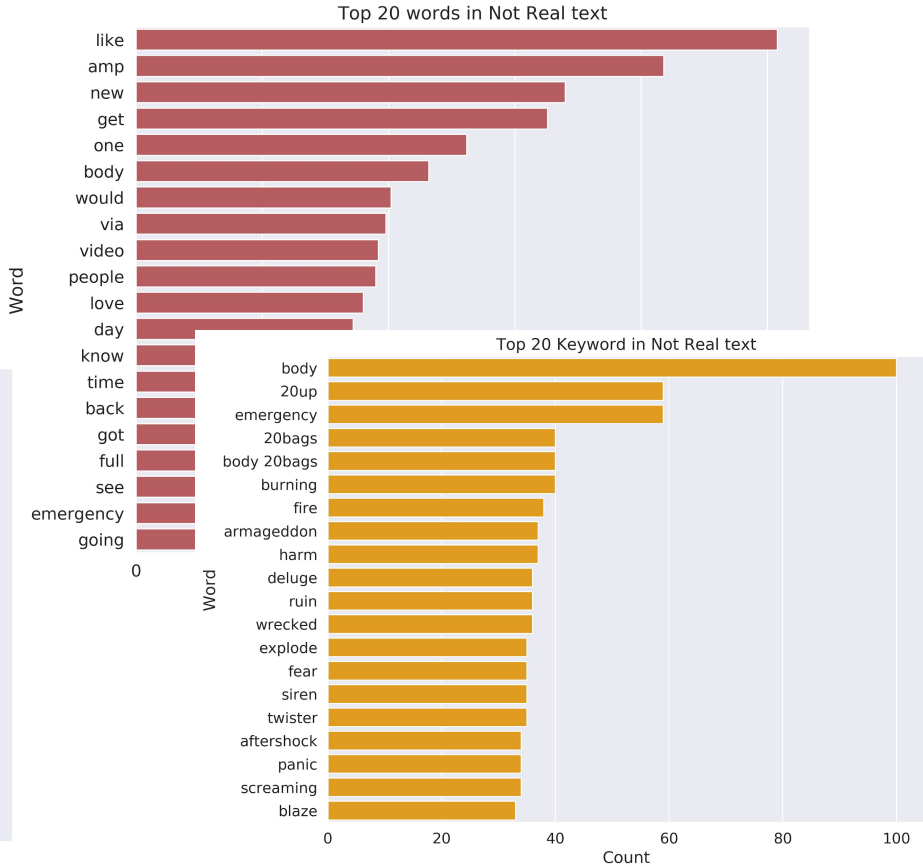


Label 1: 3271 of 7613 twits

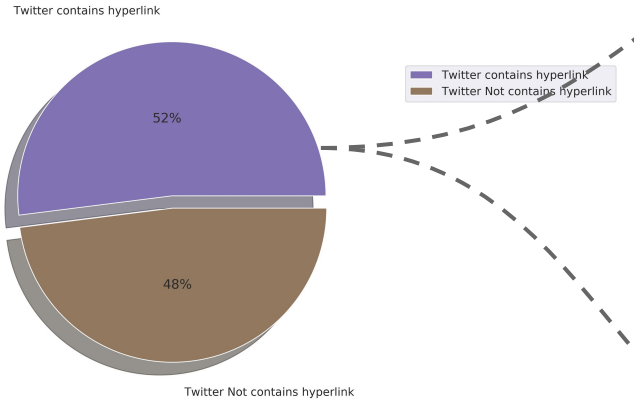
Label 0: 4342 of 7613 twits

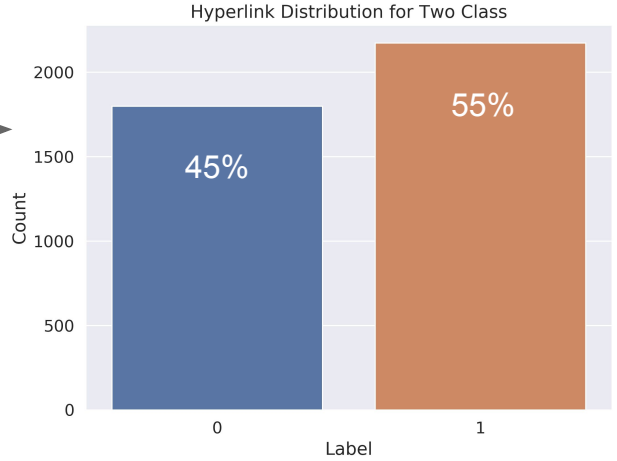
Top 20 frequent words



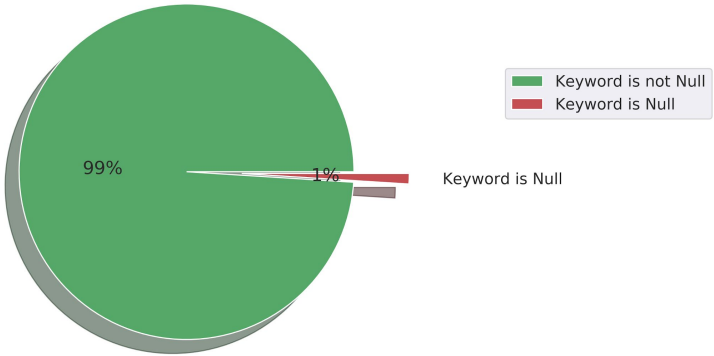


HyperLink Analysis





Keyword is not Null



3971 out of 7613 training data contains hyperlink
“ http://t.co/lHYXEOHY6C”

2172 out of 3971 is label 1
1799 out of 3971 is label 0

3951 out of 3971 contain keyword
20 out of 3971 don’t have keyword

Equivalent classes

★ 336 equivalent classes (1092 tweets out of 7613)
★ 79 “bad” equivalent classes (18 from the beginning)

{6816, 6834, 6837, 6840, 6841} -> [0, 0, 1, 1, 0]

Hollywood Movie About Trapped Miners Released in Chile:
'The 33' Hollywood movie about trapped miners starring
hyperlink

{3670, 3674, 3688, 3667, 3696} -> [0, 0, 1, 0, 0]

fatality

Fatality https://t.co/407V1y4HHg

Fatality https://t.co/GF5qjGoyCi

Fatality!

Fatality ????

{4232, 4235} -> [1, 0]

Caution: breathing may hazardous to your health.

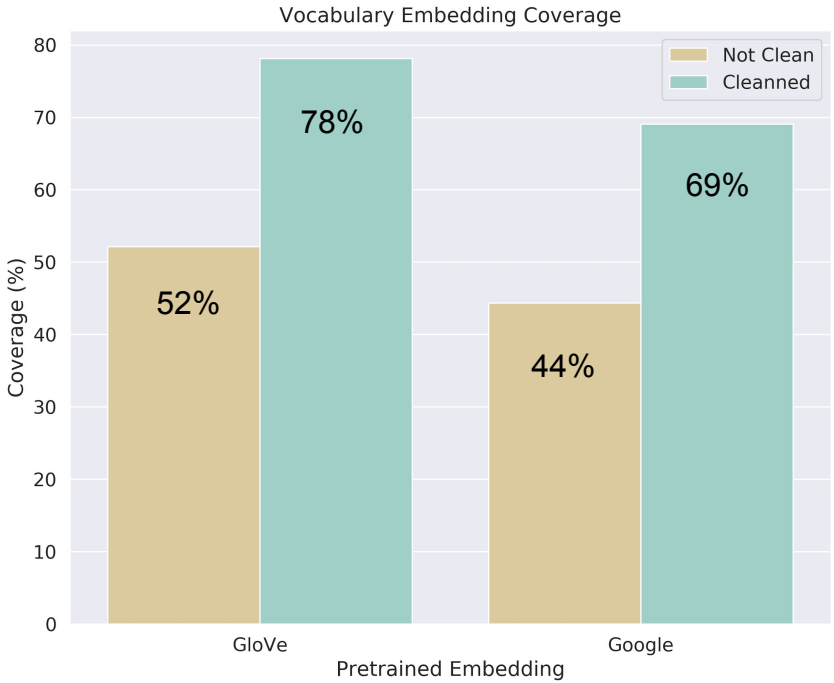
Vocabulary Embedding Coverage

- GloVe (300d)
- Google Negative(300d)

Some of the words not included in the embedding dictionary.

Eg. mh370 (GloVe,Google)
—> aircraft accident.

hiroshima, fukushima(Google)
-> atomic bomb



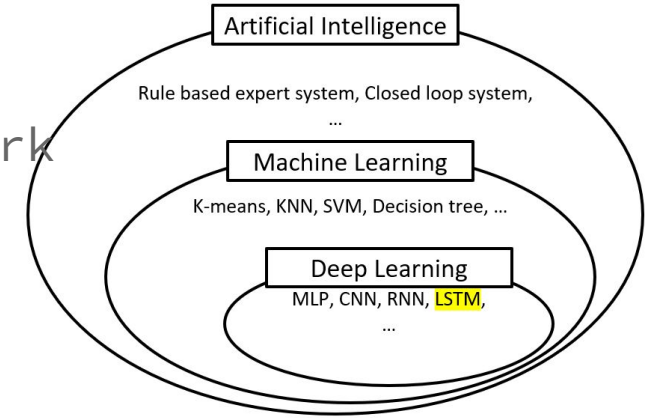
Baseline Model

- Vectorize the Text and Calculate the Frequency of each word
  - `from sklearn.feature_extraction.text import TfidfVectorizer`
- Collect Features from `TfidfVectorizer` (without semantics)
  (Term Frequency-Inverse Document Frequency)
  *tfidf(t,d,D) = tf(t,d) * idf(t, D)*
- Model : Logistic Regression and Support Vector Machine

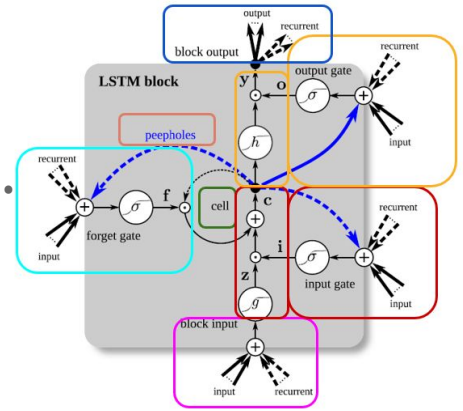
	Logistic Regression (no semantic)	SVM (no semantic)
Accuracy	0.7615	0.7815
F1	0.7561	0.7775
Precision	0.7745	0.7845
Recall	0.7611	0.7817

MODELS

Long Short term Memory - LSTM



- LSTM is part of Recurrent Neural Network under Deep Learning.
- LSTM architecture has gates and cell.
- LSTM is one of top Kaggle's prizes.
  - LSTM 23%
  - CNN 21%
  - Deep Learning 8%

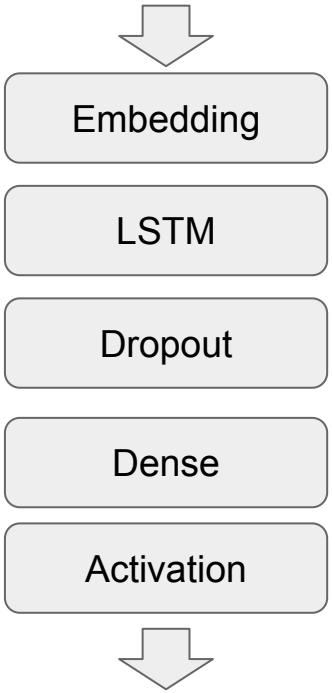


LSTM Components

1. Block input
2. Cell
3. Forget gate
4. Input gate
5. Output gate
6. Block output
7. Peephole

Reference: Graves and Schmidhuber (2005)

LSTM - Model Design




```
#LSTM
model_2 = Sequential()
model_2.add(Embedding(max_features,
                      embed_size,
                      weights=[embedding_matrix_glove],
                      input_length=maxlen,
                      trainable=False))
model_2.add(LSTM(32, kernel_regularizer=l2(0.01),
                 recurrent_regularizer=l2(0.01),
                 bias_regularizer=l2(0.01)))
model_2.add(Dropout(0.5))
model_2.add(Dense(units=300, activation='sigmoid'))
model_2.add(Dense(units=100, activation='sigmoid'))
model_2.add(Dense(units=1, activation='sigmoid'))
model_2.compile(loss='binary_crossentropy',
                optimizer='adam',
                metrics=['accuracy'])
```

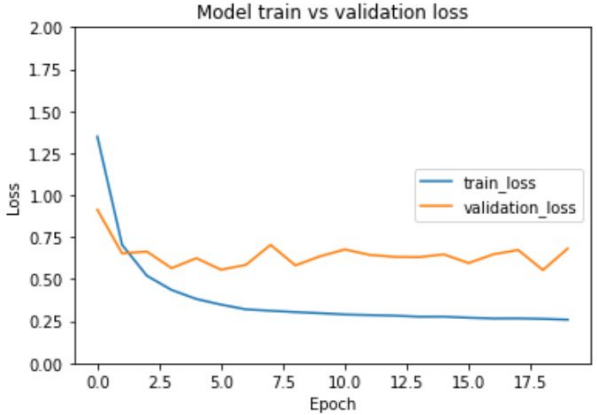
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 26, 300)           3987600
_________________________________________________________________
lstm_1 (LSTM)                (None, 32)                42624
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 300)               9900
_________________________________________________________________
dense_2 (Dense)              (None, 100)               30100
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 101
=================================================================
Total params: 4,070,325
Trainable params: 82,725
Non-trainable params: 3,987,600
_________________________________________________________________
None
```

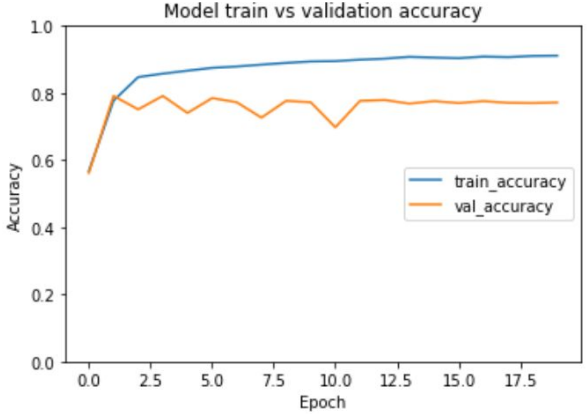
Reference: https://www.kaggle.com/martofthenorth/fork-lstm-of-shiba-inu-keraslstm-2/edit/run/32760562

LSTM-Model Performance

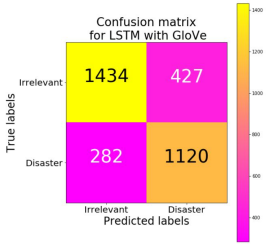
Shallow LSTM :77%

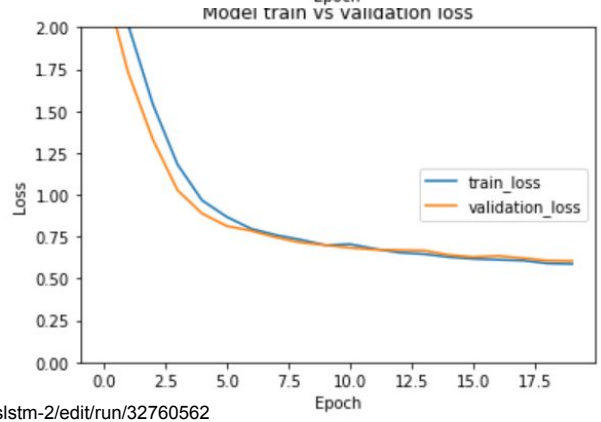
(Note: Bi-LSTM and GRU show similar result.)

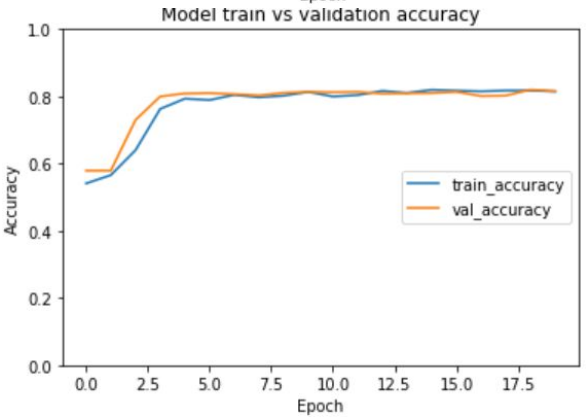




GloVe LSTM :78%



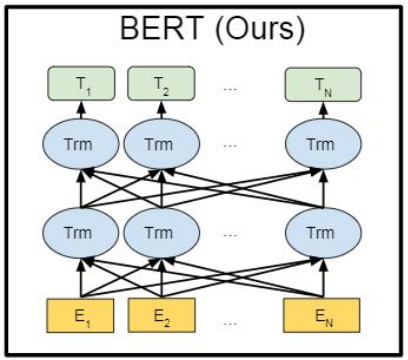


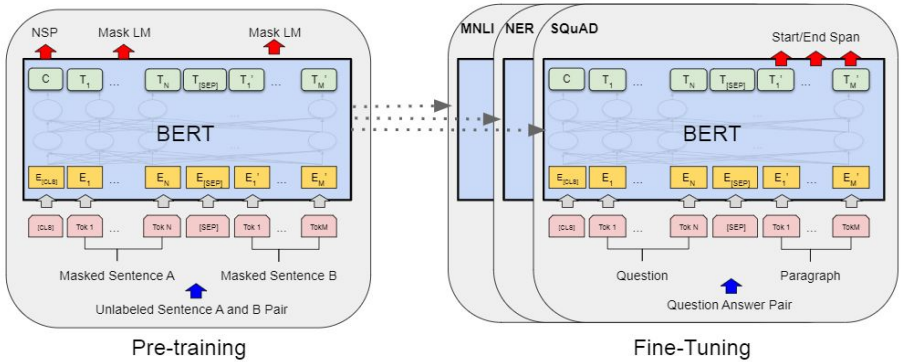


References:
https://www.kaggle.com/martofthenorth/fork-lstm-of-shiba-inu-keraslstm-2/edit/run/32760562
https://www.kaggle.com/martofthenorth/clean1-glove1-lstm1-l1l2-fm-e20-mart?scriptVersionId=32828964

BERT - BASICS

- BERT (Bidirectional Encoder Representations from Transformers)
  - “Masked language model” task
  - “Next sentence Prediction task”



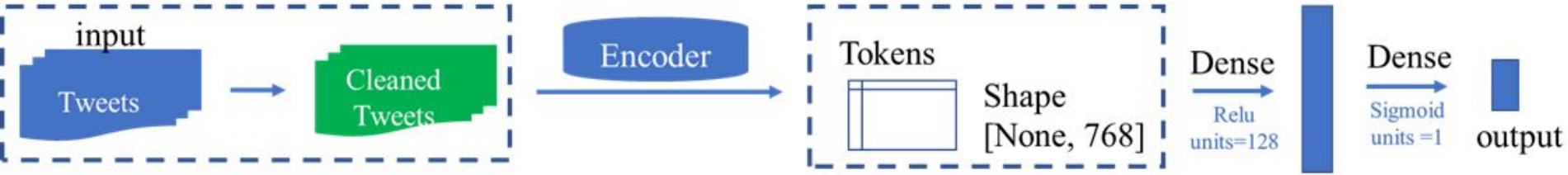


- Encoder: Google pre-trained Model (bert_en_uncased_L-12_H-768_A-12)
  - L = 12 hidden layers (i.e., Transformer blocks),
  - hidden size : H = 768, and A = 12 attention heads.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

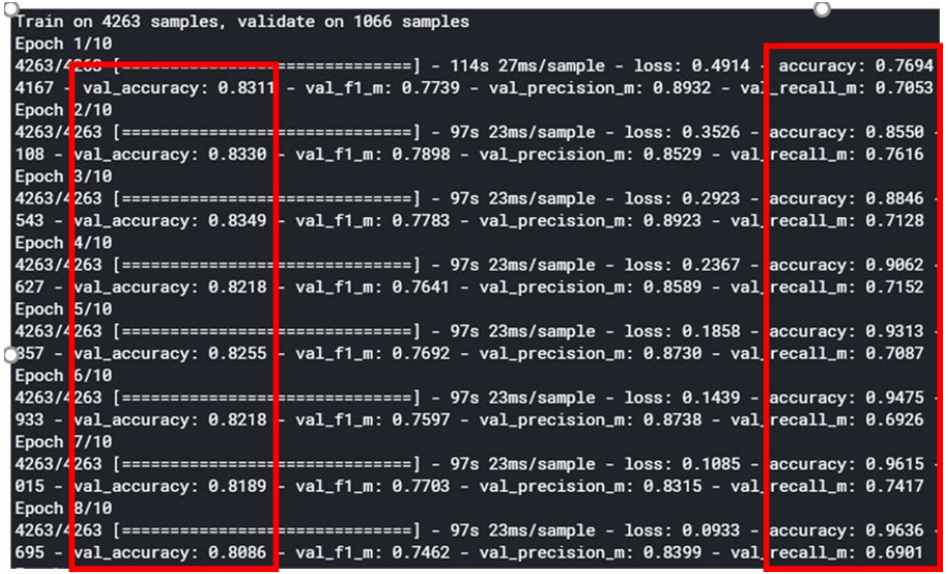
BERT - Applications

- Clean Methods
  - clean while keep the original semantics
  - clean while keep the sentence structure
- How we use BERT
  - Original Input: Tweet (Cleaned Version)
  - Input: Result from pre-train BERT is a [*, 768] vector
  - Add 2 Dense Layers
    - Unit = 128, activation Relu/Sigmoid
    - Unit = 1, activation Sigmoid (Final Result)



BERT - Tuning

- Parameters Tuning:
  - Learning Rate: 6e-6
  - Number of Dense Layer： 2
  - Epoch: 3
  - Batch Size: 16



- Result on Testing (w/o clean)
  - with clean:
    loss: 0.3158 acc: 0.8774 f1: 0.8474
    precision_m: 0.8759 recall: 0.8276

  - Without clean:
    loss: 0.4020 acc: **0.8314** f1: 0.7920
    precision_m: 0.8689 recall: 0.7387

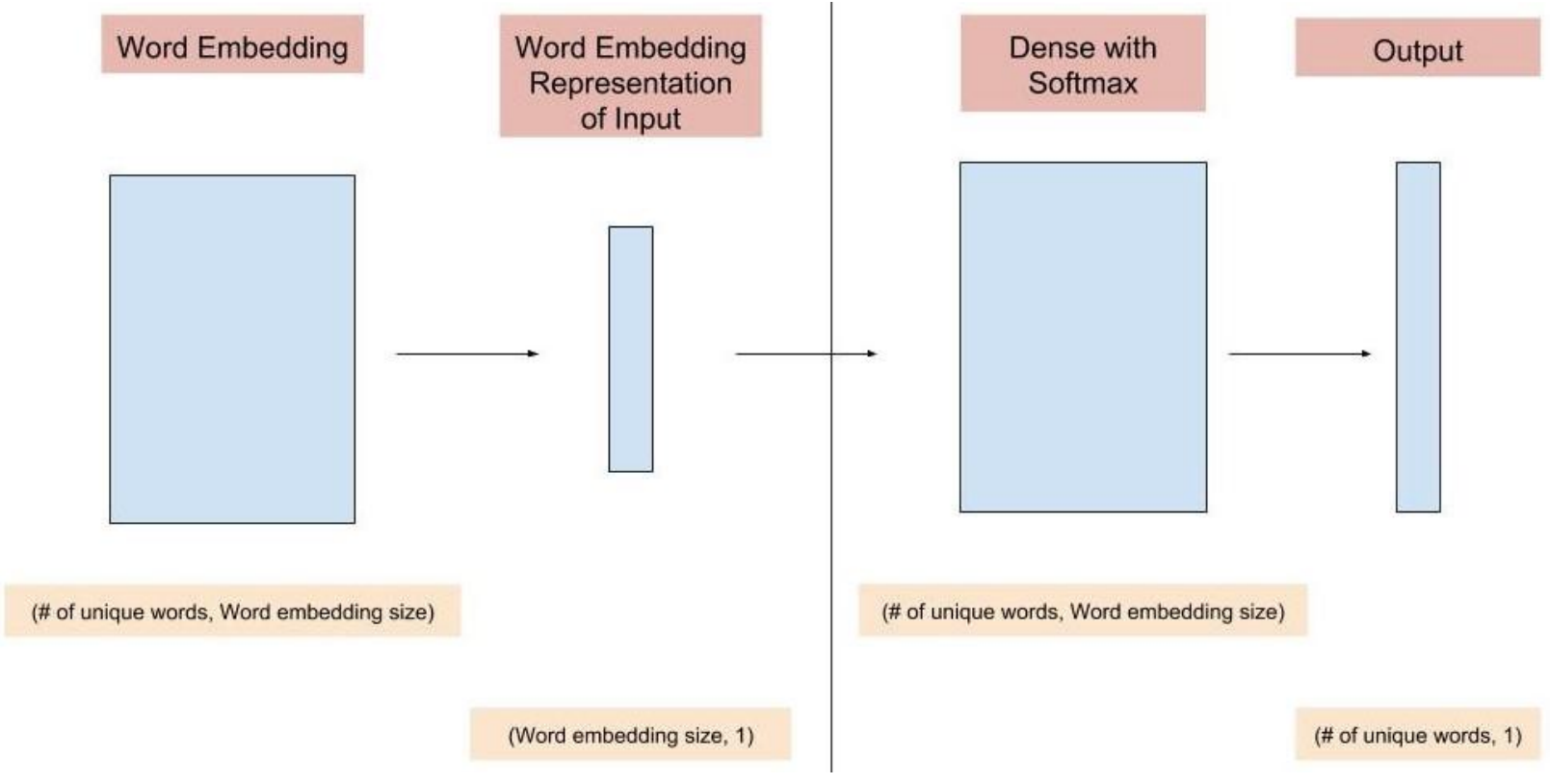
WORD2VEC

- **CBOW**
- **Skip-gram**

After the deduction of the costs of investing, beating the stock market is a loser's game.

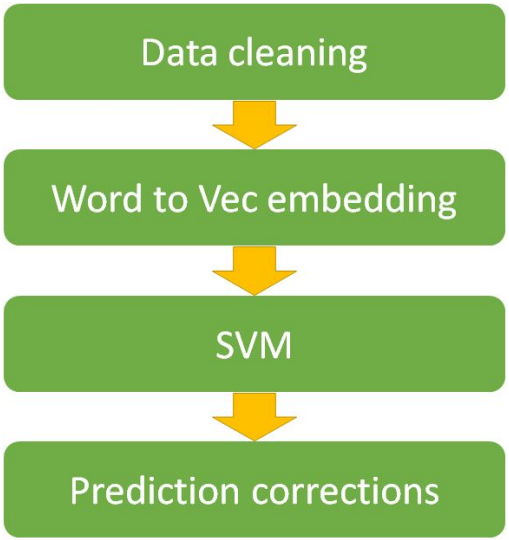
Window size of 5

Context word



GoogleNews-vectors-negative300

- Pre-trained **vectors** trained on a part of **Google News** dataset (~ 100 billion words)
- Contains 3 million English words mapped to 300-dimension vectors



➢ Take one representative from each of 79 “bad” equivalent classes

➔ Tweet -> average of word embeddings

➔ If a word is absent -> zero-vector

★ 5-fold cross validation: C = 2

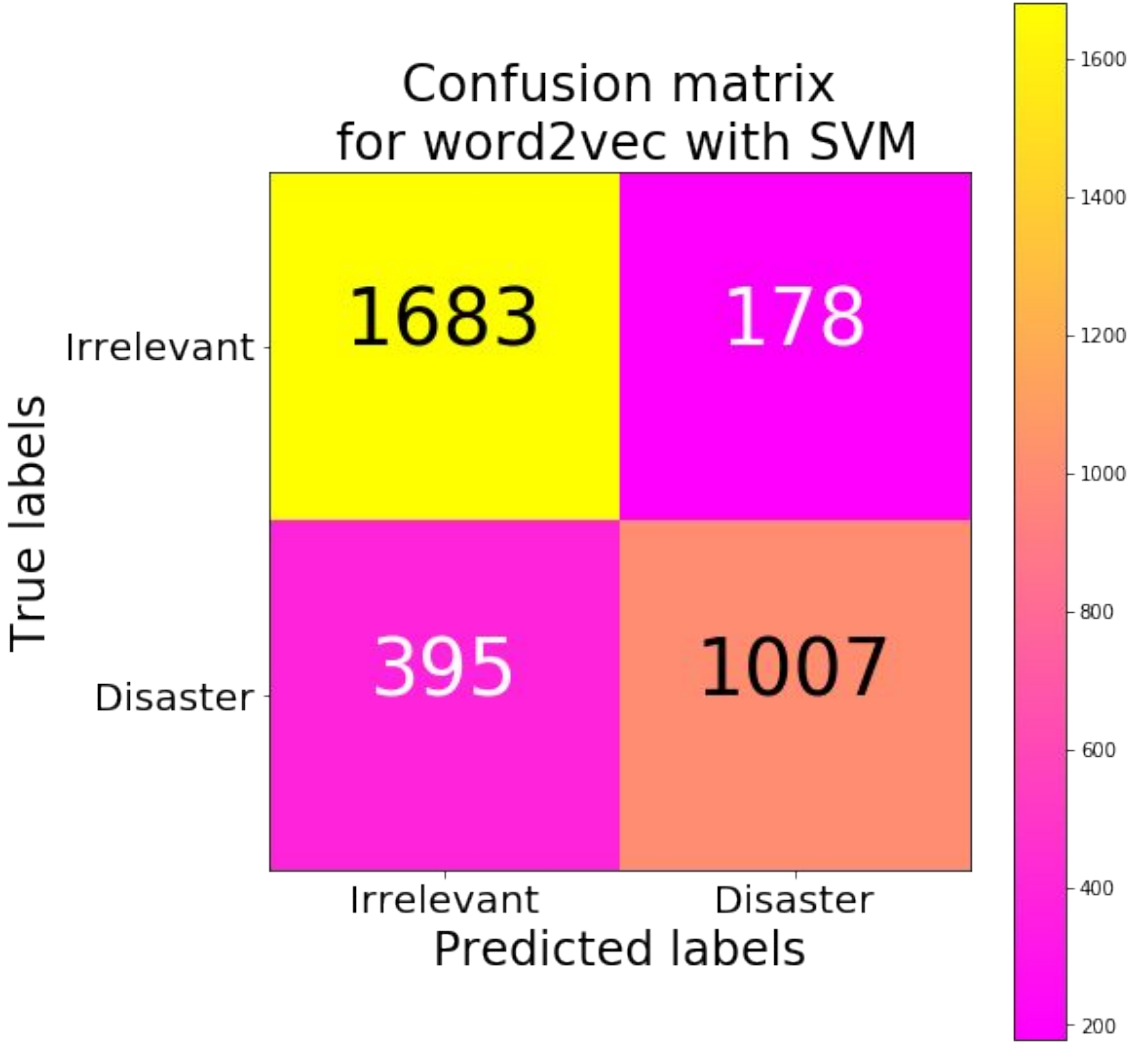
❑ Some “positive” keywords matter

WORD2VEC: RESULTS

TRAINING
accuracy = 0.8443
f1 = 0.8426
precision = 0.8461
recall = 0.8443

TEST
accuracy = 0.8244
f1 = 0.8219
precision = 0.8270
recall = 0.8244

USED TO BE
accuracy = 0.8195
f1 = 0.8166
precision = 0.8227
recall = 0.8195



Commit History

SVM with Words2Vectors Mapping (version 17/17) — 0.81083
5 hours ago by Kiril Kuzmin
From "SVM with Words2Vectors Mapping" Script

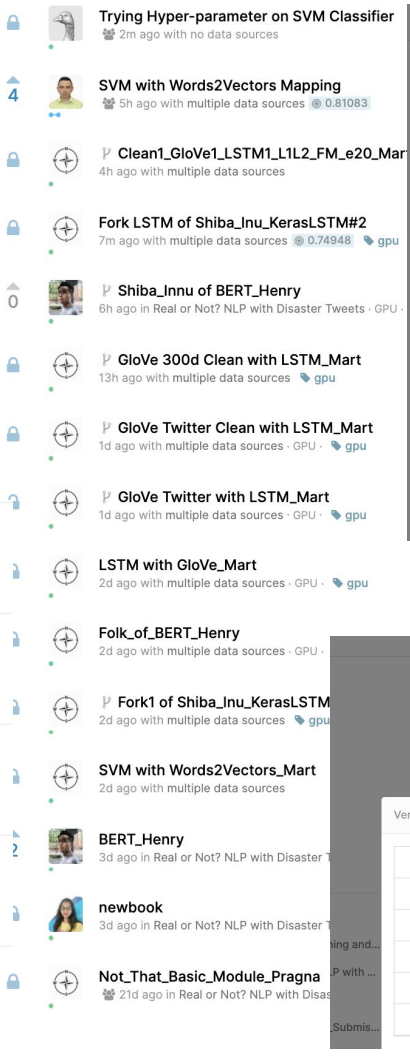
submission.csv — 0.82617
9 hours ago by henryascend
BERT - RELU + SIGMOID ###### Comment for this version Basic Cleaning Adam: lr=6e-6 MAX_LEN = 160 validation_split=0.2, epochs=3, batch_size=16

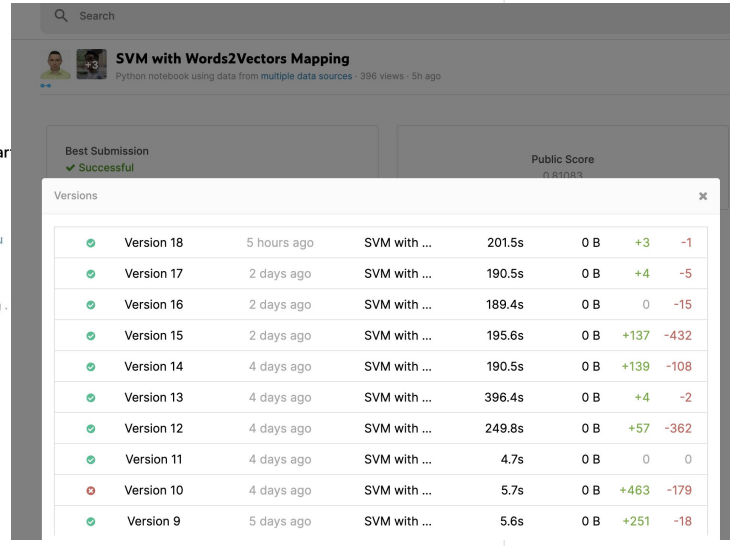
submission.csv — 0.82617
2 days ago by henryascend
BERT text basic cleaning with lr 2e-6 4 epochs

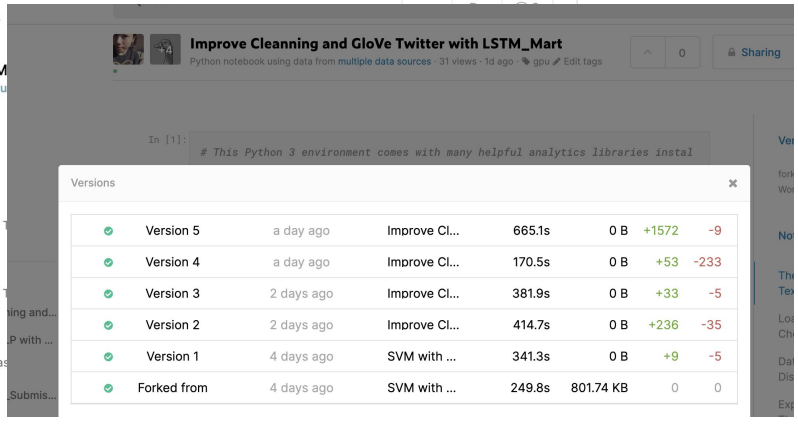
submission.csv — 0.84253
3 days ago by henryascend
BERT - with text cleaning

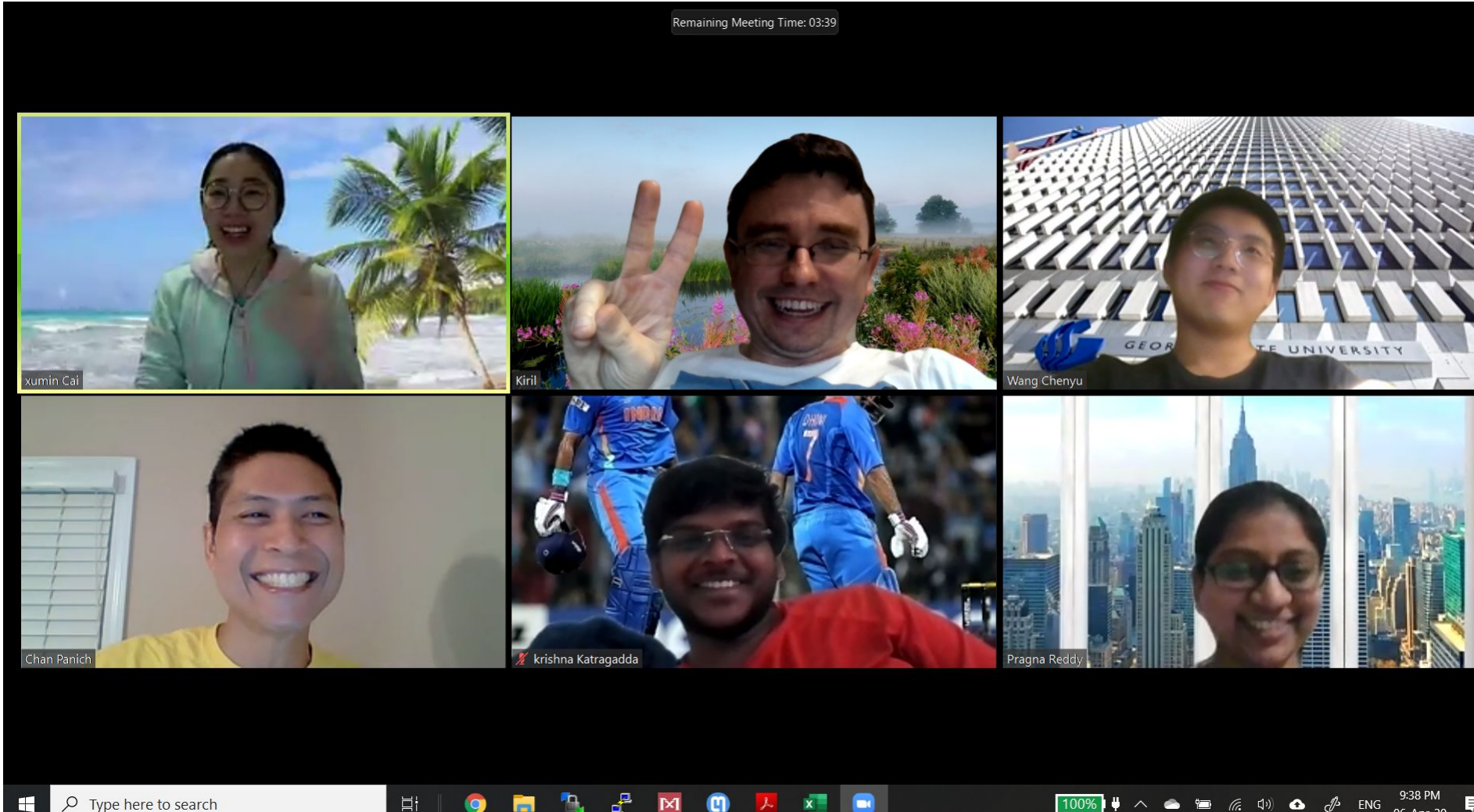
SVM with Words2Vectors Mapping_from_Kiril (version 2/8) — 0.80163
4 days ago by Sumi
From "SVM with Words2Vectors Mapping_from_Kiril" Script

SVM with Words2Vectors Mapping_from_Kiril (version 1/8) — 0.80674
4 days ago by Sumi
From "SVM with Words2Vectors Mapping_from_Kiril" Script



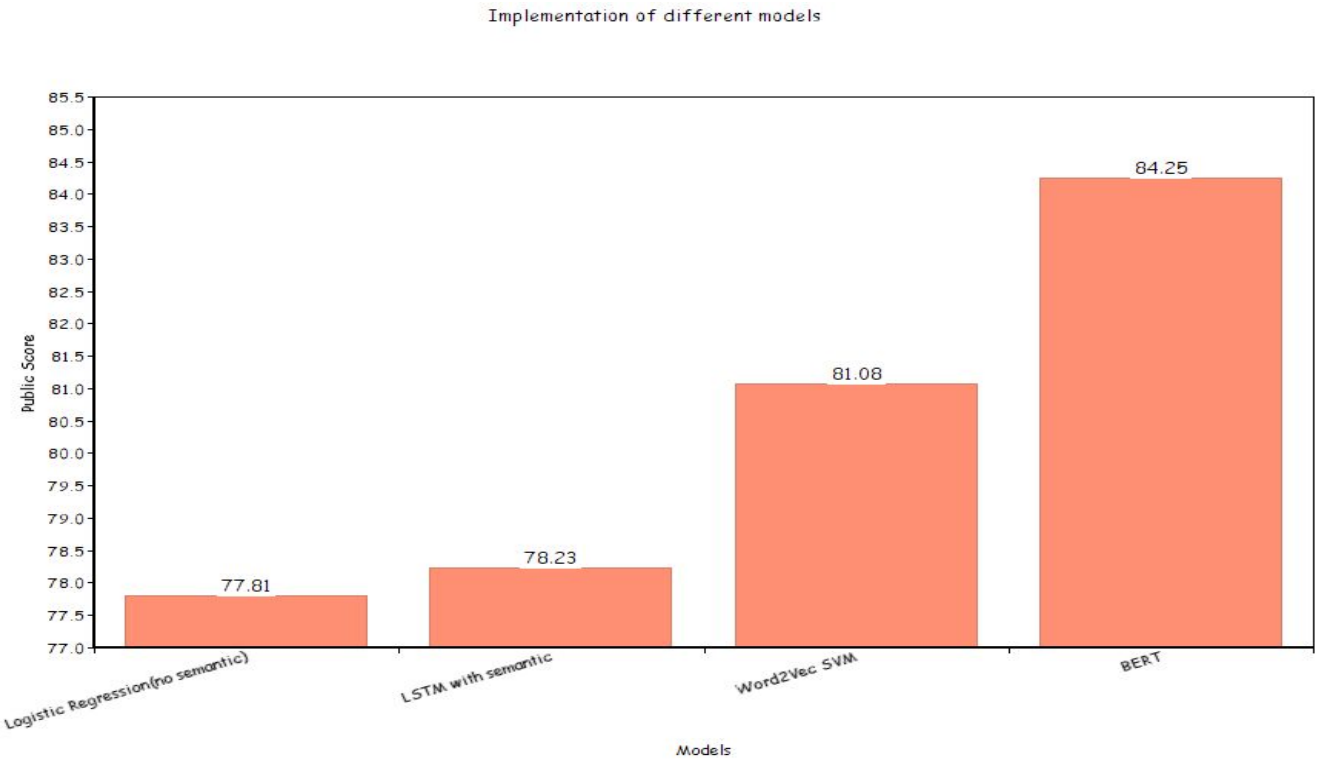





Zoom
Remaining Meeting Time: 03:39
xumin Cai
Kiril
Wang Chenyu
Chan Panich
krishna Katragadda
Pragna Reddy
Type here to search
100%
ENG
9:38 PM

Results

Implementation of different models



Results: Kaggle Leaderboard

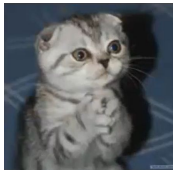
Overview Data Notebooks Discussion Leaderboard Rules Team My Submissions **Submit Predictions**

322	Chris X	0.84253	1	2mo
323	Nakul Agrawal	0.84253	11	1mo
324	Town Meng	0.84253	16	1mo
325	X.Y.Z	0.84253	9	20d
326	noob at best	0.84253	45	1mo
327	H.Eren	0.84253	19	1mo
328	Sudhir Kumar	0.84253	13	1mo
329	xuyiqi	0.84253	10	12h
330	Shiba_Inu	0.84253	25	4h

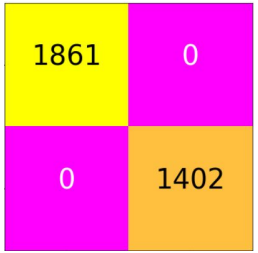
Your Best Entry ↑

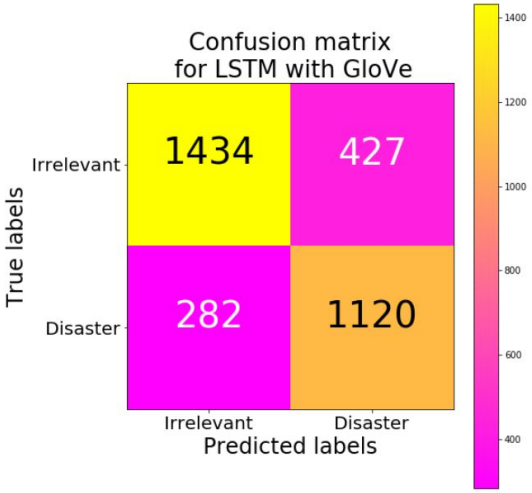
Your submission scored 0.82617, which is not an improvement of your best score. Keep trying!

331	jeslyn&nuulllll	0.84151	2	2mo
332	Gibran Otazo	0.84151	7	2mo
333	neptunexxx	0.84151	8	1mo
334	Shravan	0.84151	2	2mo
335	Ivtang	0.84151	16	2mo

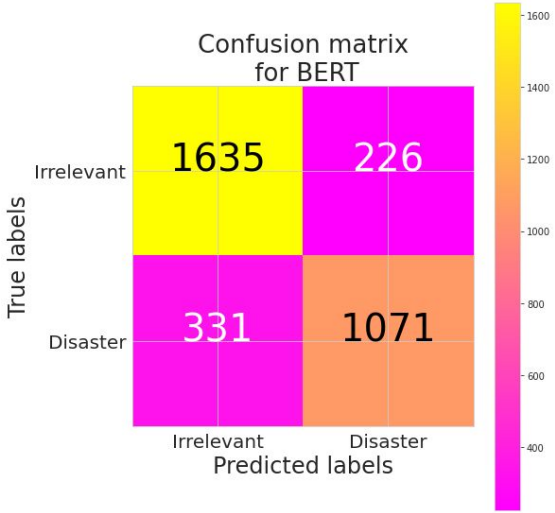


Confusion Matrix Comparison

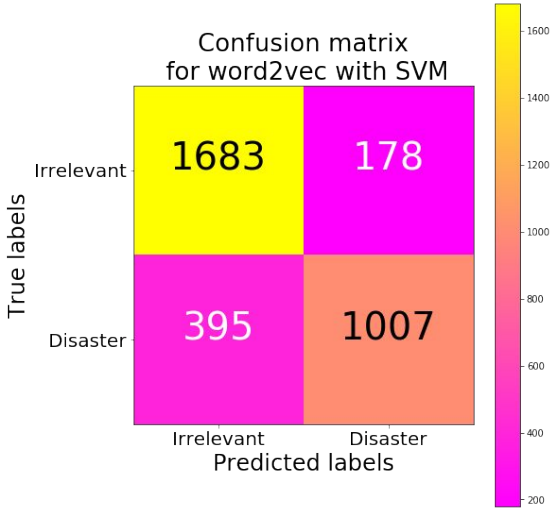




LSTM Glove

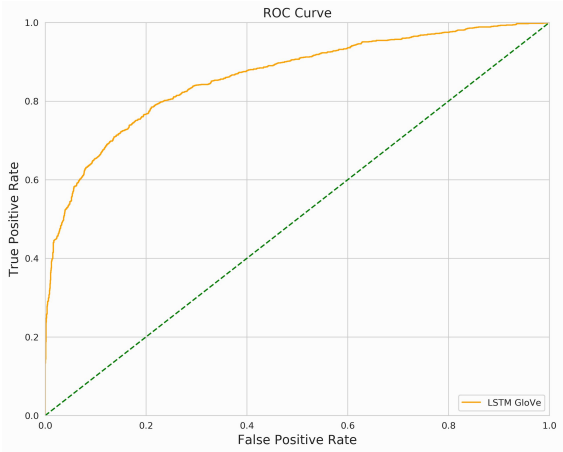


BERT

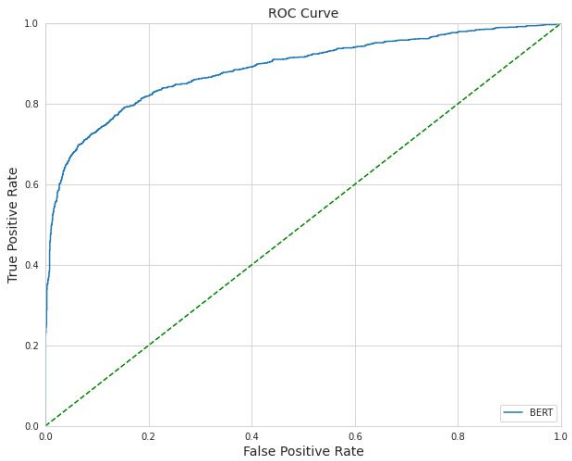


Word2ve SVM

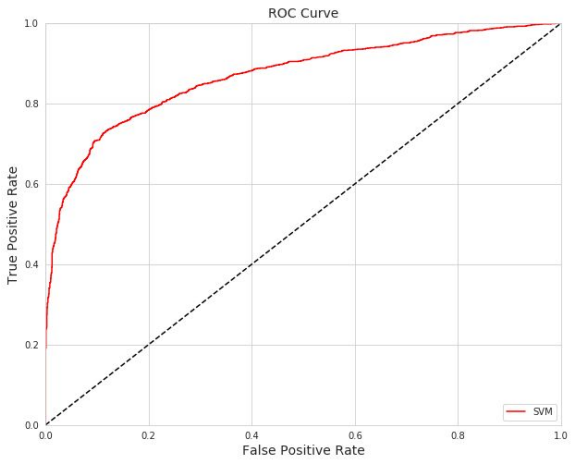
ROC Curve Comparison



LSTM Glove

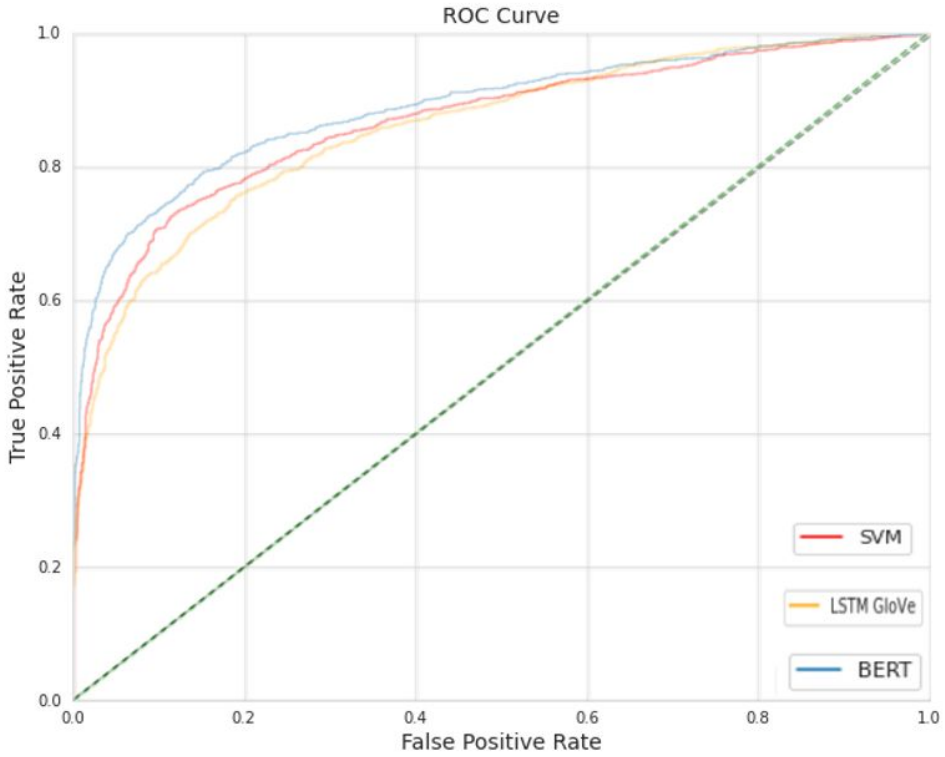


BERT



SVM

ROC Curves together



Conclusion

1. Some tweets have contradictory labels
2. BERT showed the best results:
    a. public score: 0.8425
    b. accuracy = 0.8774 f1 = 0.8474 precision = 0.8759 recall = 0.8276
3. Word2Vec is next:
    a. public score: 0.8108
    b. accuracy = 0.8244 f1 = 0.8219 precision = 0.8270 recall = 0.8244
4. Tomáš Mikolov is smart
5. Sometimes it can go off…
6. We have learnt:
    a. how to use Kaggle platform
    b. way to clean text data
    c. NLP
    d. shallow vs deep models for NLP
    e. how to work in a team

The End.

Q&A