59000-146 Machine Learning

Disaster Tweets Prediction

With Recurrent Neural Network

Presented By :
Da-Fang Lin, Tzu-Yun Huang

Agenda

Data Exploration 01

Data Preprocess 02

Model Building & Training 03

Learning and Insights 04

1 Exploring the Disaster Tweets Dataset

Train

Column	Non-Null Count	Missing Value(%)	Dtype
id	7613	0%	int64
keyword	7552	0.8%	Object
location	5080	33%	Object
text	7613	0%	Object
Target	7613	0%	int64

Test

Column	Non-Null Count	Missing Value(%)	Dtype
id	3263	0%	int64
keyword	3237	1.8%	Object
location	2158	33%	Object
text	3263	0%	Object

Load SpaCy Model

```
> import spacy
> nlp = spacy.load("en_core_web_sm")
```

Loads the pre-trained English NLP model from SpaCy (en_core_web_sm) to process text data.

- Tokenization
- Lemmatization
- Dependency Parsing

Setting Random Seed

```
> seed = 42
> np.random.seed(seed)
```

Why Set a Random Seed?

- Ensures the model produces the same results
- Easier to debug because results don’t change unpredictably

2-1 Text Preprocessing

Text Cleaning

- **Convert to lowercase**
- **Remove numbers**
- **Remove punctuation**
- **Remove HTML tags**
- **Remove extra spaces**

Tokenization and Lemmatizatoin

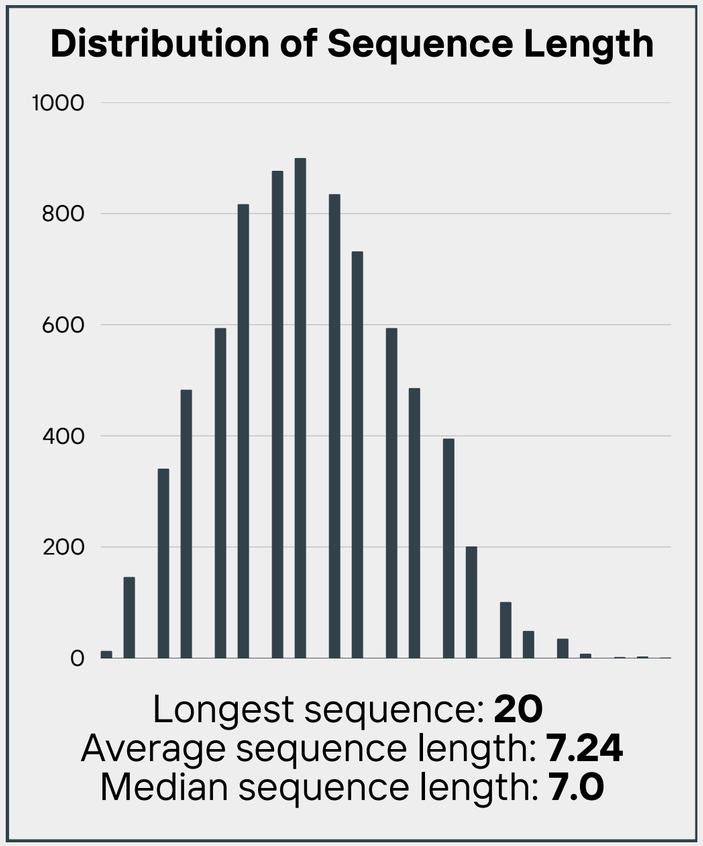
"The wildfires are spreading rapidly!"

Tokenization the process of splitting a sentence into individual words or tokens

Lemmatization reduces words to their base or dictionary form

"wildfire spread rapidly"

Pad Sequence



Longest sequence: **20**
Average sequence length: **7.24**
Median sequence length: **7.0**

2-2

Feature Engineering

Keywords

One-hot encoding creates binary columns

Fatalities	Deluge	Armageddon
1	0	0
0	1	0
0	0	1

After One-Hot Encoding:
Train Shape : (7613, 220)
Test Shape : (3263, 220)

Location

One-hot encoding creates binary columns

NYC	San Francisco	Los Angeles
1	0	0
0	1	0
0	0	1

After One-Hot Encoding:
Train Shape : (7613, 3102)
Test Shape : (3263, 3102)

Sentiment Score

Analyzes the polarity of words by using VADER

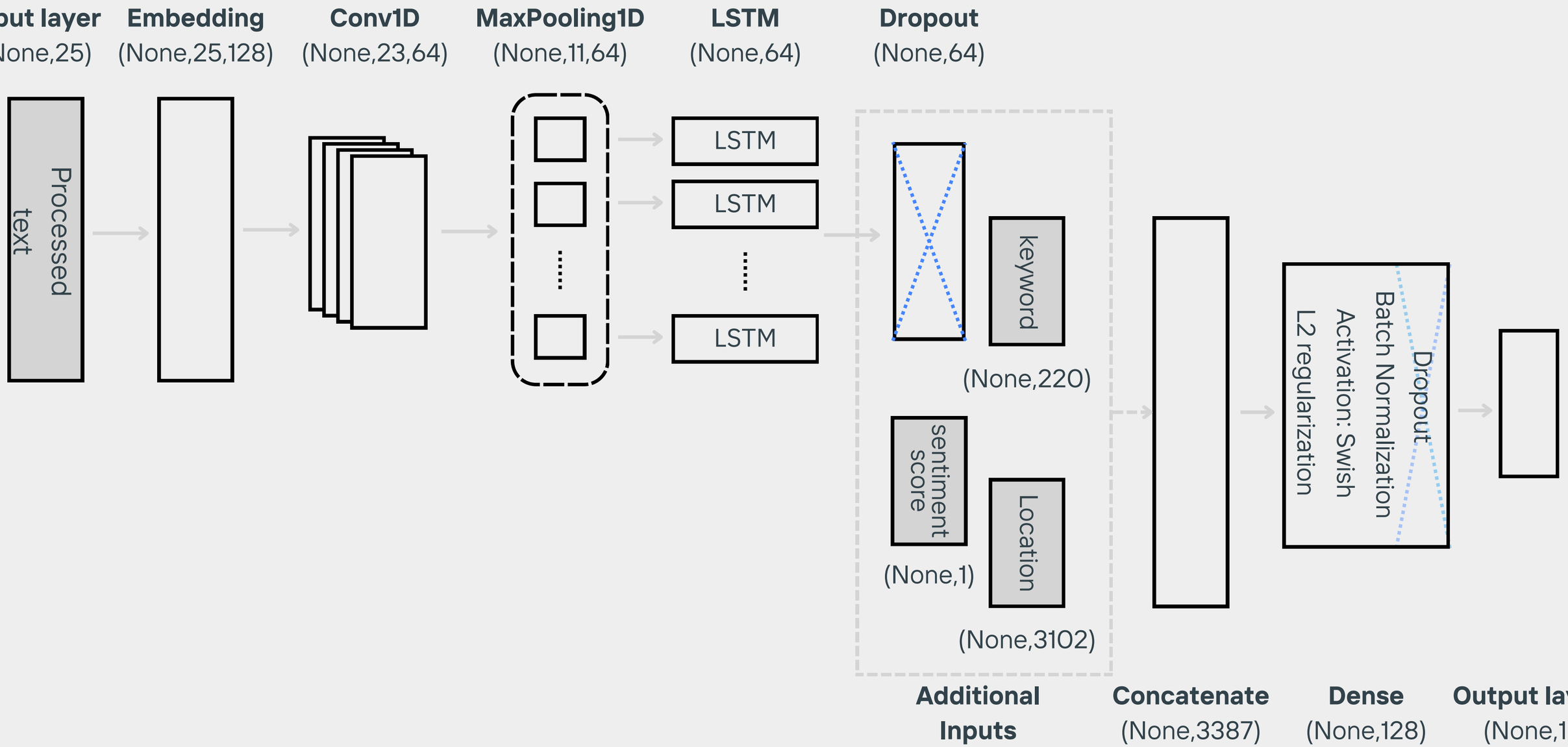
Positive (%) | Neutral (%) | Negative (%)

Final sentiment score (ranges from -1 to 1)

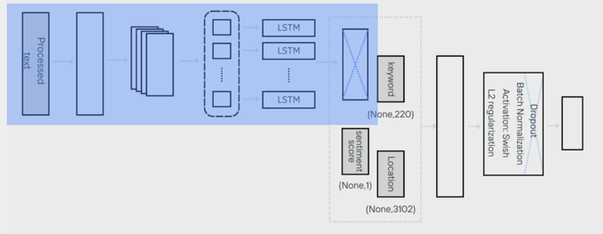
"hear an earthquake in different city stay safe" — **0.44**

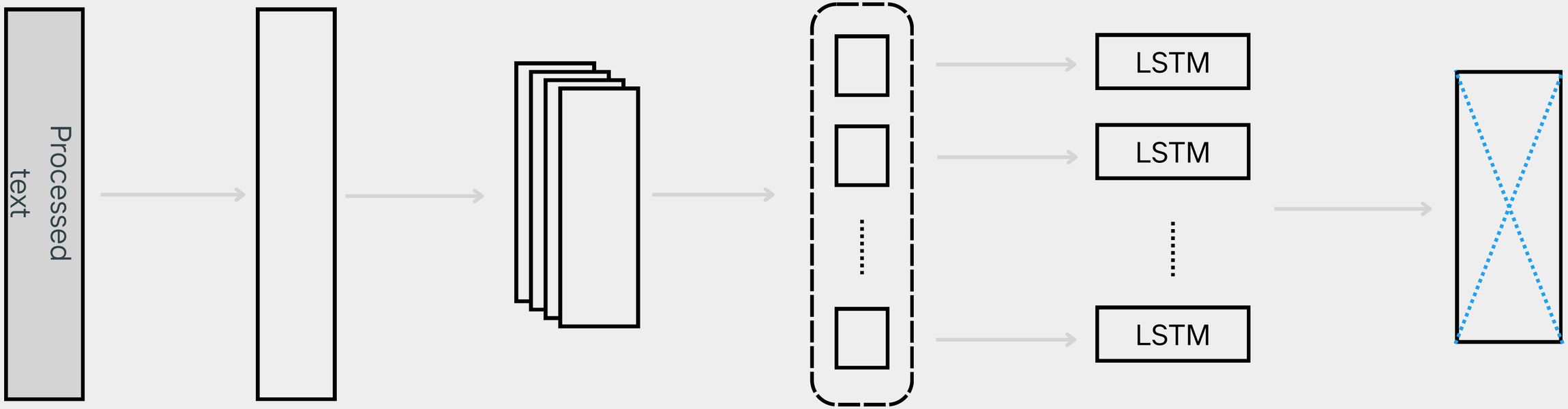
"typhoon soudelor kill people in china and taiwan " — **-0.69**

3-1 Model Structure



3-1 Model Structure





Input layer
(None,25)

Embedding
(None,25,128)

```
input_dim=5000
output_dim=128
input_length=25
```

Conv1D
(None,23,64)

```
filters=64
kernel_size=3
activation='relu'
```

MaxPooling1D
(None,11,64)

```
pool_size=2
```

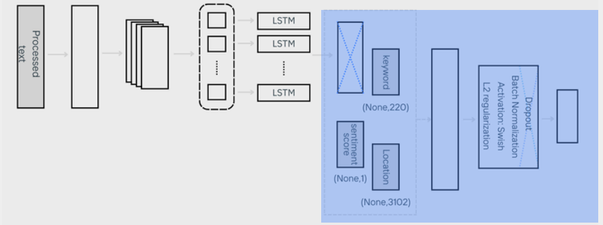
LSTM
(None,64)

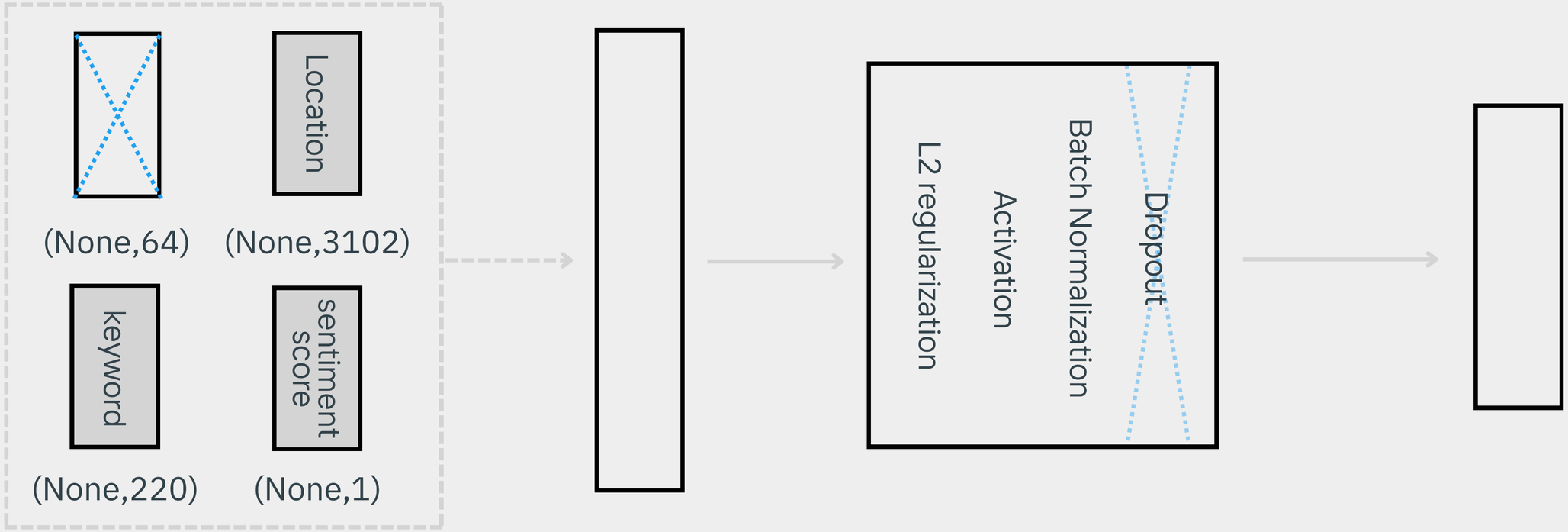
```
kernel_regularizer=l2(0.001)
```

Dropout
(None,64)

```
Dropout(0.6)
```

3-1 Model Structure





Input layer

Concatenate
(None,3387)

Dense layer
(None,128)

```
kernel_regularizer=l2(0.001))
Activation('swish')
BatchNormalization()
Dropout(0.4)
```

Output layer
(None,1)

```
activation='sigmoid'
```

3-2 Model Training

ModelCheckpoint

```
monitor='val_loss'
save_best_only=True
```

Automatically save the best model, ensuring that the model used for prediction is the best version, rather than the result from the final epoch.

EarlyStopping

```
monitor='val_loss'
patience=3
restore_best_weights=True
```

If ‘val_loss’ does not improve for 3 consecutive epochs, the training will stop to avoid wasting time on training an overfitted model.

ReduceLROnPlateau

```
monitor='val_loss'
factor=0.5
patience=3
min_lr=0.00001
```

When ‘val_loss’ does not improve, the learning rate will be automatically reduced to help the model converge more stably.

```
epochs=50, batch_size=32
```

❌ Use default thresholds

✅ Find the best threshold

Different data may require different thresholds.

Without the best threshold, the model might classify suboptimally, missing the chance to achieve the best F1 score and impacting overall performance.

```
best_thresh = t
```

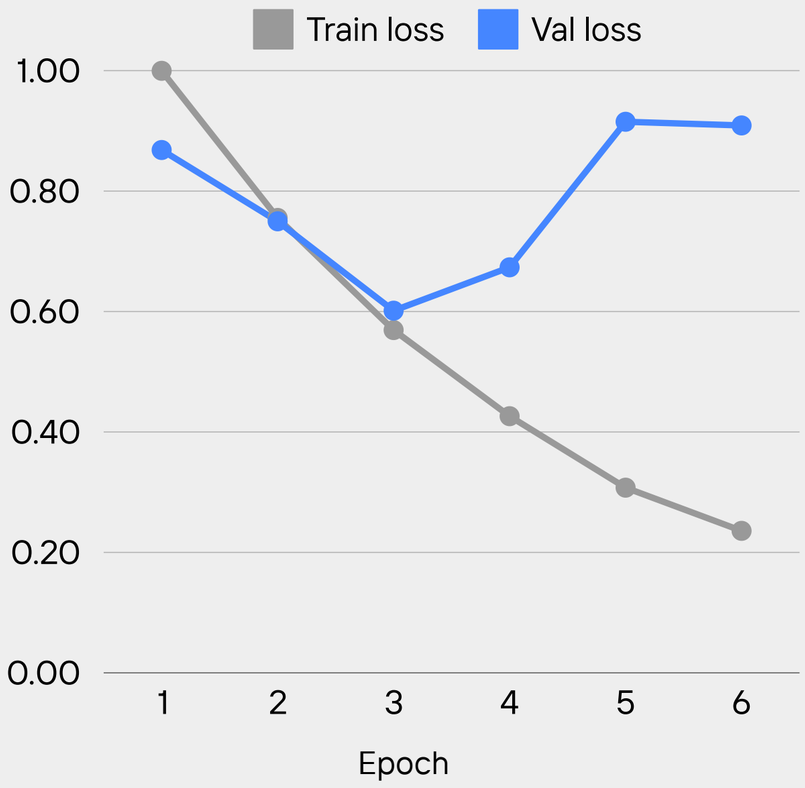
If the predicted value is greater than `t`, it is classified as 1; otherwise, it is classified as 0.

3-4 Model result

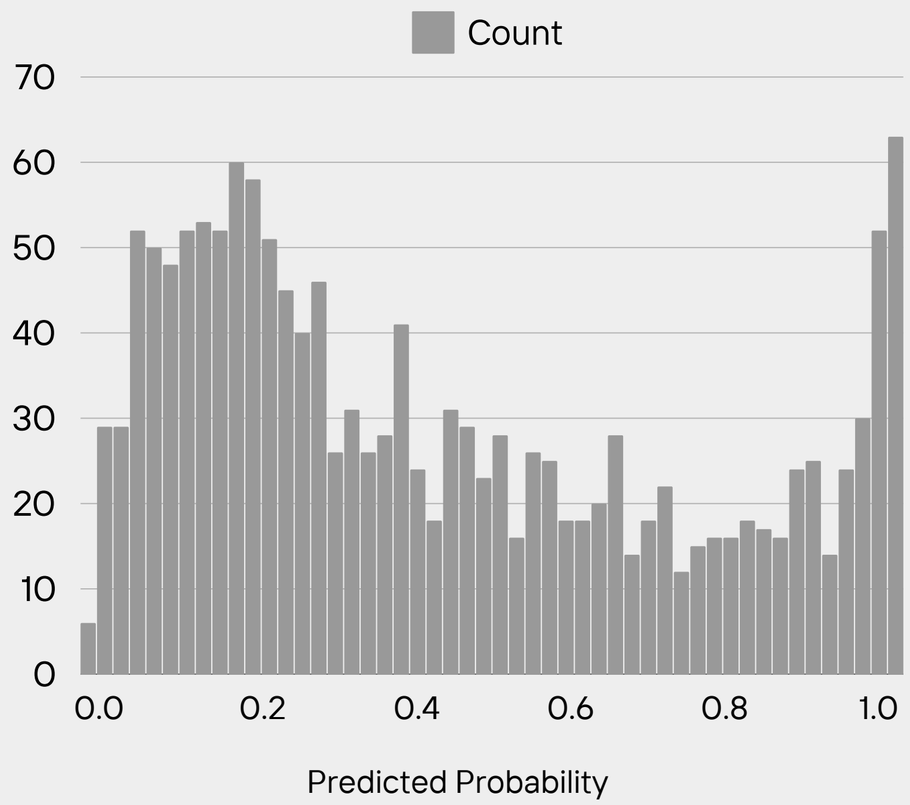
Best Threshold: 0.5000000000000001
Best F1: 0.7417004048582996

Public Score on Kaggle:
0.79589

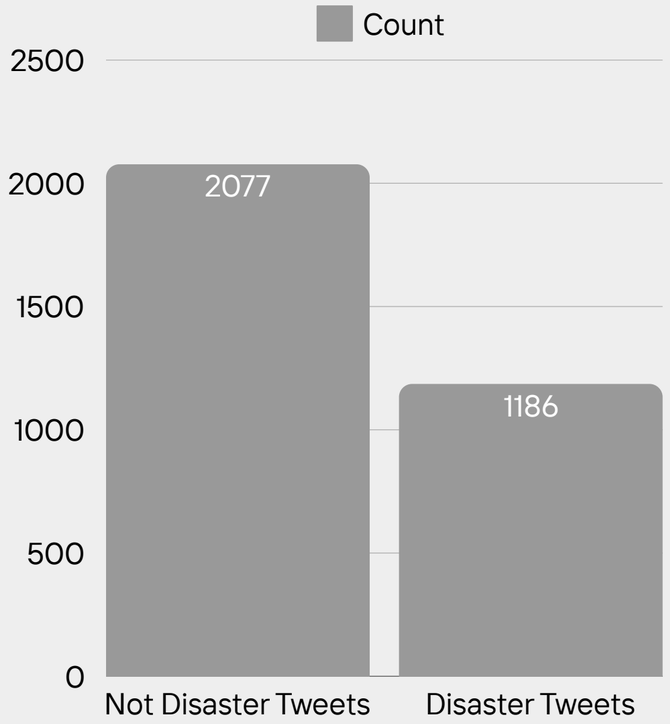
Training & Validation Loss



Distribution of Predicted Probabilities

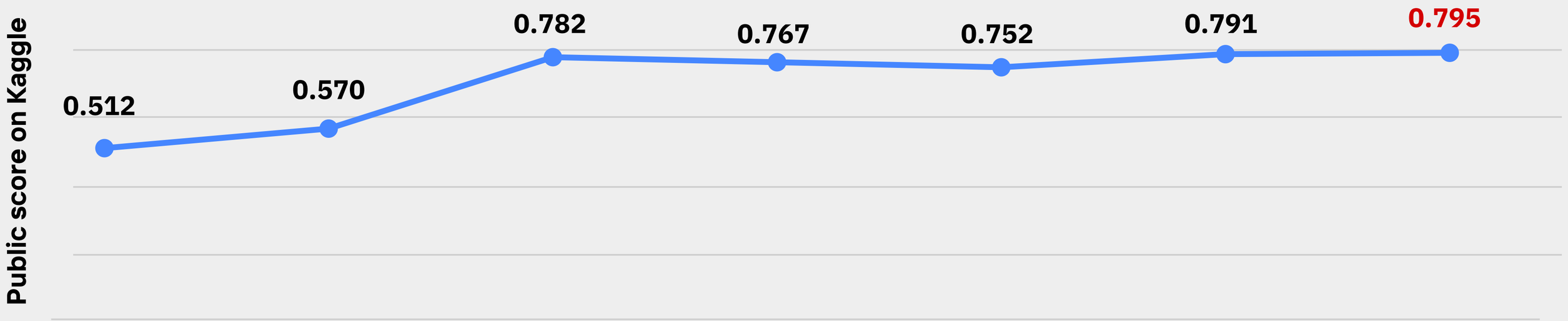


Prediction Results Distribution



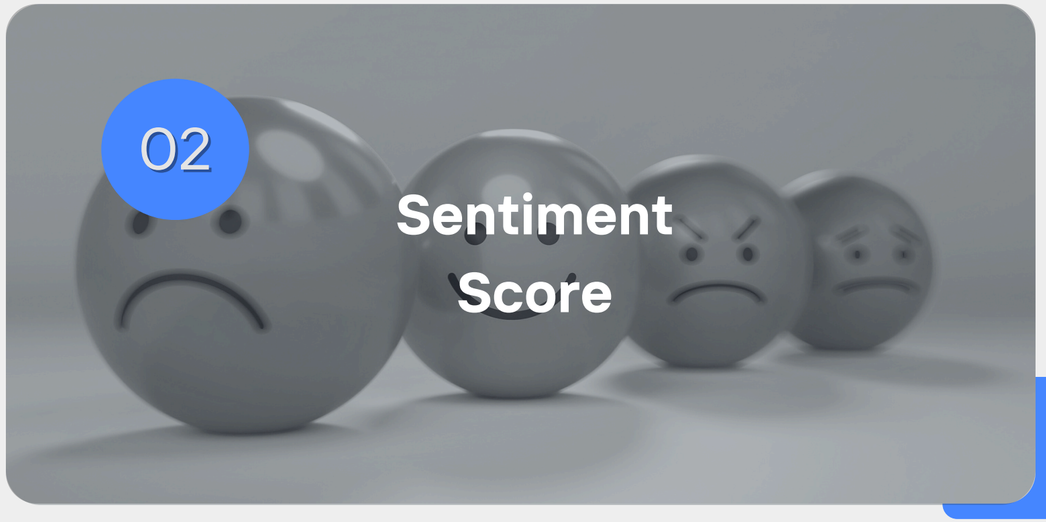
3-5

Model tuning



Embedding + LSTM + Dropout Concatenate all input + Dense (w/ LeakyReLU, BatchNormalization)	Embedding + LSTM + Dropout Concatenate all input + Dense (w/ LeakyReLU, BatchNormalization) ＊Find the Best Threshold	Embedding + Conv1D + MaxPooling1D + LSTM Concatenate all input + Dense (w/ L2, Swish, BatchNormalization, Dropout)	Embedding + Conv1D + MaxPooling1D + BiLSTM Concatenate all input + Dense (w/ L2, Swish, BatchNormalization, Dropout)	Embedding + Conv1D + MaxPooling1D + BiGRU Concatenate all input + Dense (w/ L2, Swish, BatchNormalization, Dropout)	Embedding + Conv1D + MaxPooling1D + LSTM Add on sentiment score as input + Dense (w/ L2, Swish, BatchNormalization, Dropout)	Embedding + Conv1D + MaxPooling1D + LSTM Add on sentiment score as input + Dense (w/ L2, Swish, BatchNormalization, Dropout) ＊Fine tune parameter

Learning



👉 Our text is relatively short. As a result:

- LSTM can effectively capture the main semantics.
- BiLSTM/BiGRU introduces backward information, which may add noise instead of value, reducing model performance.

Without sentiment scores, some non-disaster tweets may be misclassified.

👉 We use VADER to enhance classification by detecting negative or urgent tones in disaster-related posts.

Thank you

Disaster Tweets Prediction

Presented By :
Da-Fang Lin, Tzu-Yun Huang