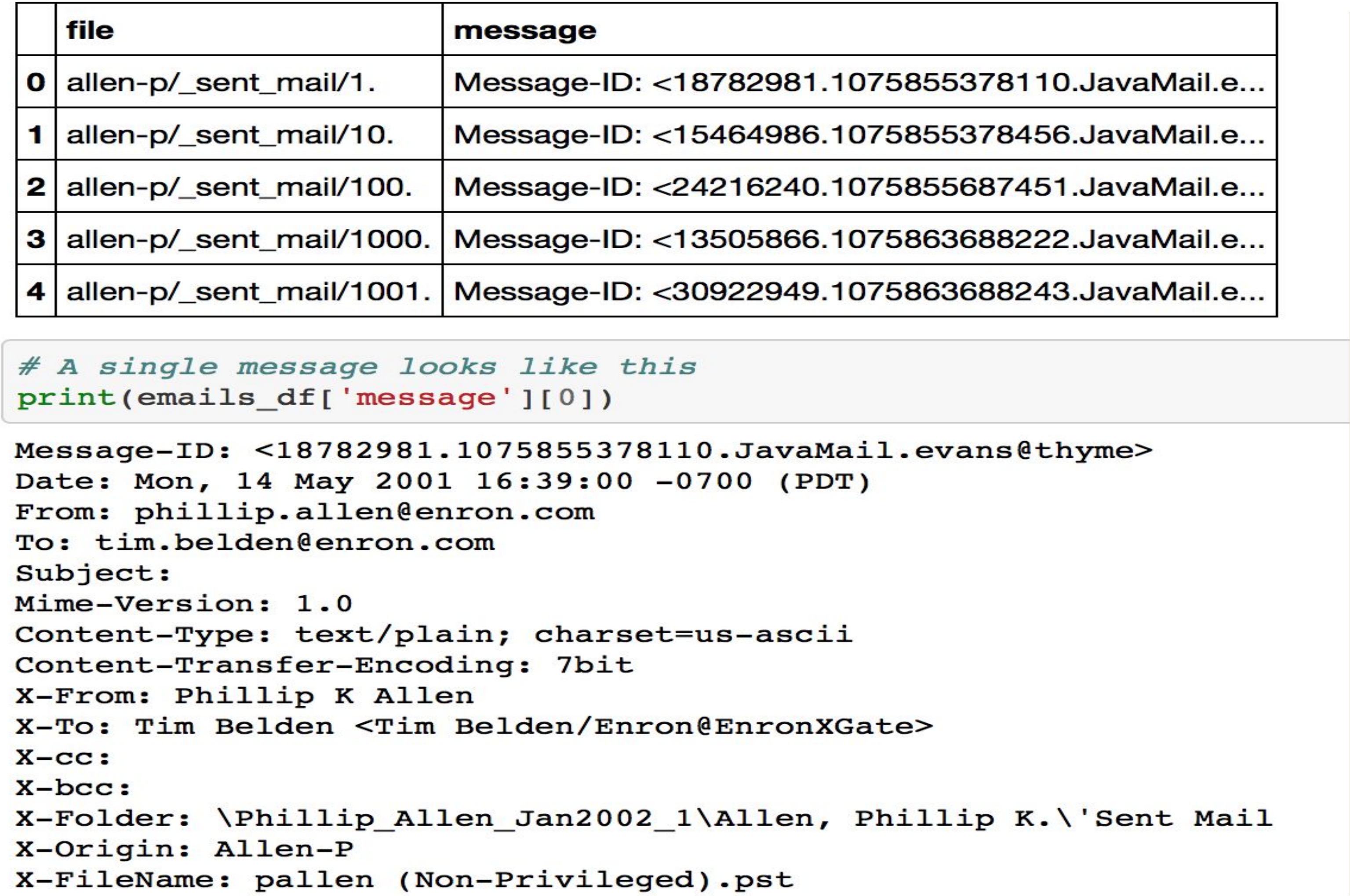
EMAIL AUTO COMPOSE with RNN (LSTM)

Wise Monk Team Members:
Yuanzhe Li, Yuhua He, Samual Yang, Jia Ma, Ying Laura Liu

Primary Dataset

- Primary: Enron Email dataset from Kaggle.com: https://www.kaggle.com/wcukierski/enron-email-dataset
- The Enron email dataset contains approximately 500,000 emails generated by employees of the Enron Corporation. It was obtained by the Federal Energy Regulatory Commission during its investigation of Enron's collapse.
- Original Dataset has two columns: File & Messages. We parsed the dataset into more number of features.

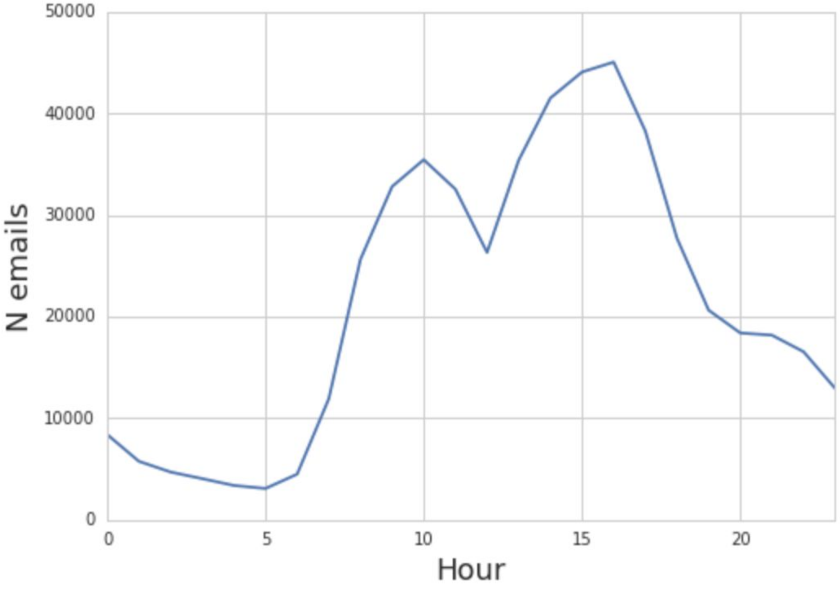


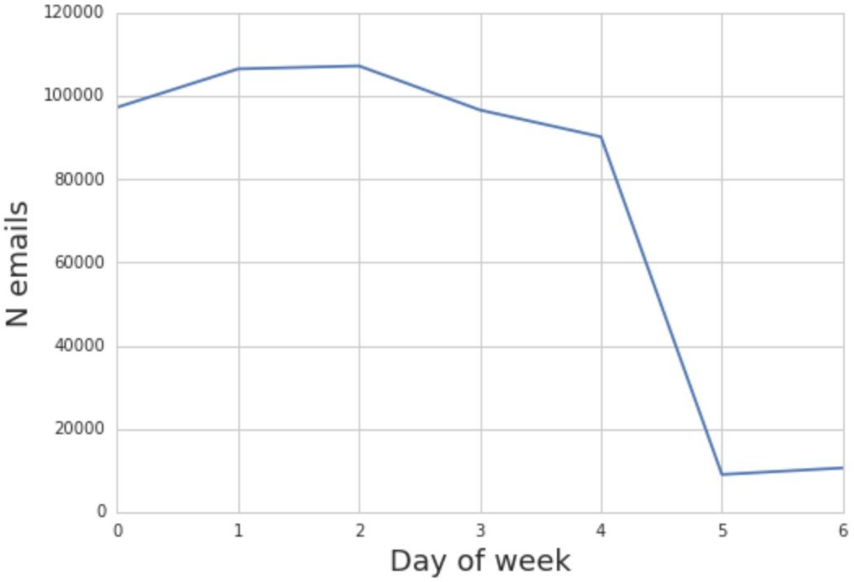
	file	message
0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...
1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...
2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...
3	allen-p/_sent_mail/1000.	Message-ID: <13505866.1075863688222.JavaMail.e...
4	allen-p/_sent_mail/1001.	Message-ID: <30922949.1075863688243.JavaMail.e...

```python
# A single message looks like this
print(emails_df['message'][0])
```

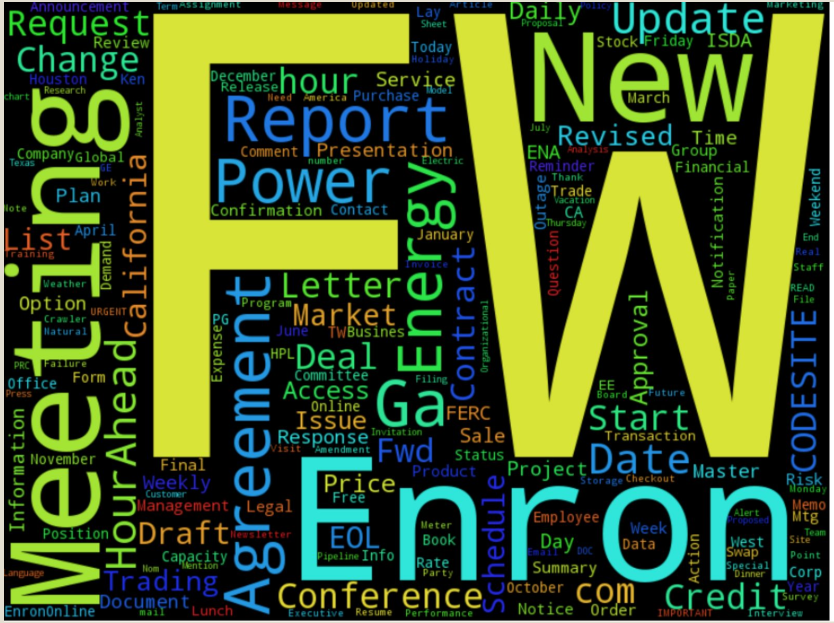
```
Message-ID: <18782981.1075855378110.JavaMail.evans@thyme>
Date: Mon, 14 May 2001 16:39:00 -0700 (PDT)
From: phillip.allen@enron.com
To: tim.belden@enron.com
Subject:
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: Phillip K Allen
X-To: Tim Belden <Tim Belden/Enron@EnronXGate>
X-cc:
X-bcc:
X-Folder: \Phillip_Allen_Jan2002_1\Allen, Phillip K.\'Sent Mail
X-Origin: Allen-P
X-FileName: pallen (Non-Privileged).pst
```

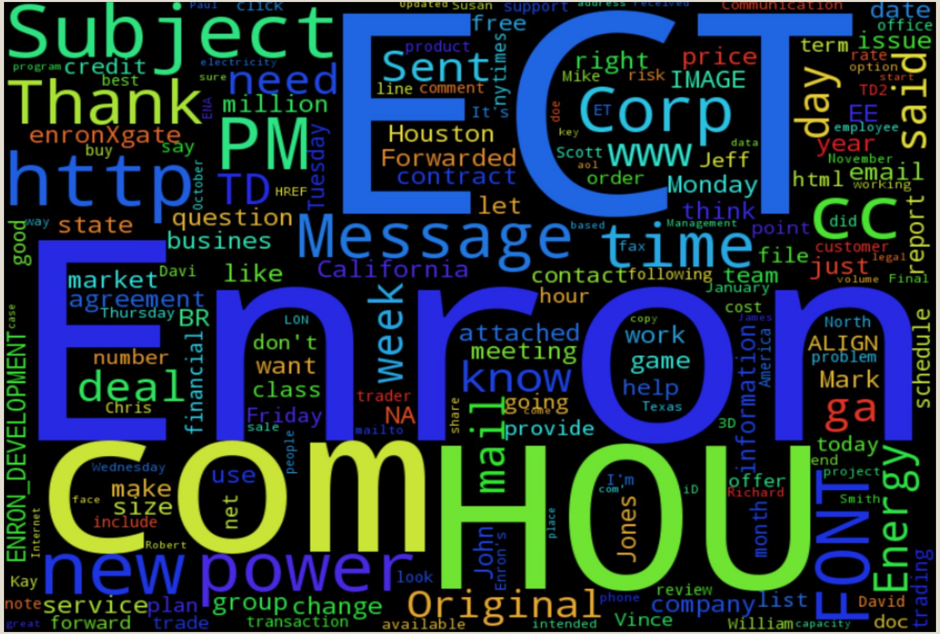
When do people send emails?





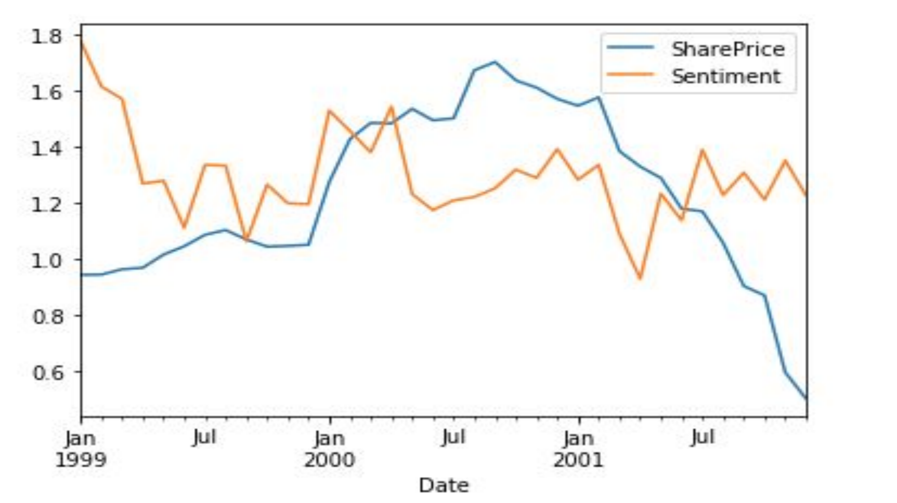
What do the emails say?





Enrichment Datasets

- Conversation between robot and human: https://www.kaggle.com/cnin0770/nlp-practice. This dataset is from chatbot (rDany)'s conversation used for Telegram, Kik and Messenger.
- Enron stock chart data: enronstockchat.xlsx
- Enron Events https://www.nytimes.com/2006/01/18/business/worldbusiness/timeline-a-chronology-of-enron-corp.html". We used beatiful soup to scrape the data from this url.



Data Preprocessing

- Tf-IdF Vectorization
-
- Removed Stop words
-
- PCA
- Label-Encoder
- LDA: topic modelling to find out the 10 frequent topics by using Genim library
- Bag of words

Unsupervised Learning & Supervised Learning

- K-means clustering
- GMM clustering

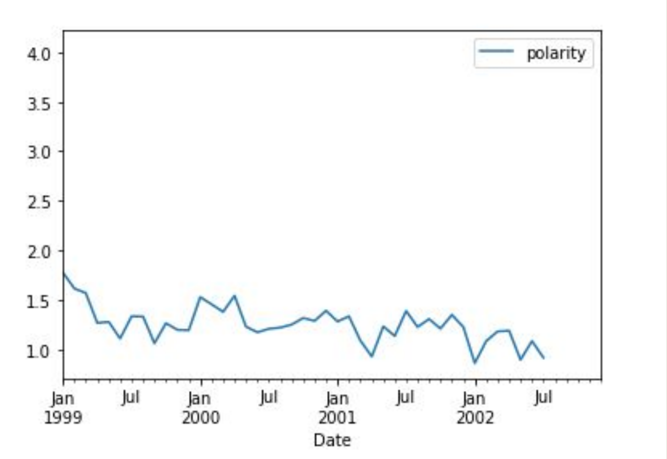
We used the clustering methods to identify four major clusters: emails with similar words

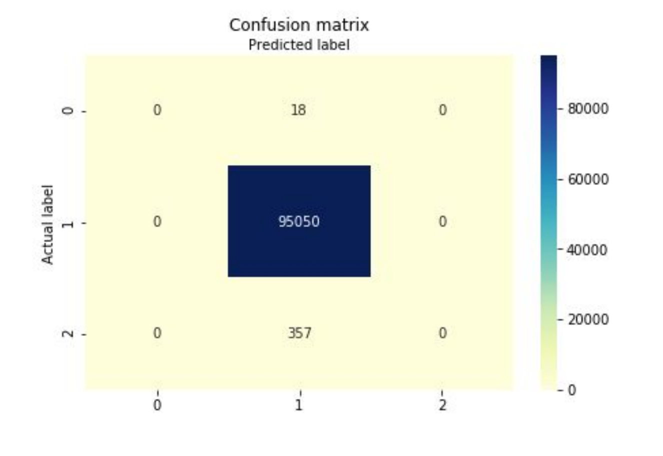
- Linear Regression
- Logistic Regression
- Random Forrest.

We evaluated the prediction accuracy by confusion matrix, and ROC curve

Sentiment Analysis

- Enron sentiment started decreasing at Jan 2001, as Skilling becomes CEO while Lay stays on as chairman.
- December 2, 2001 - Enron files for Chapter 11 bankruptcy protection.
- Used logistic Regression as prediction model. The model is to predict label of “subjectivity” or “polarity”.



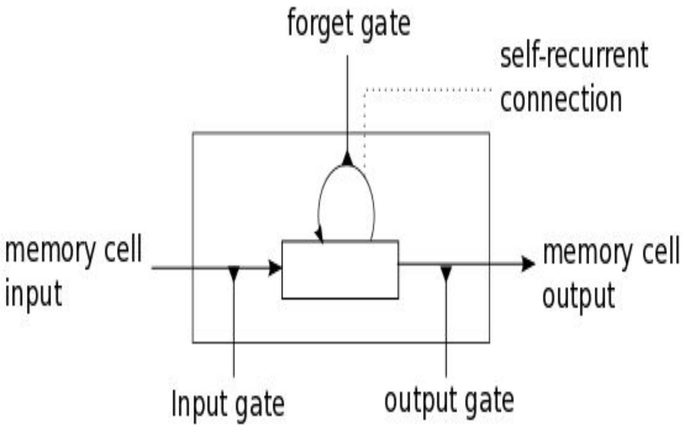


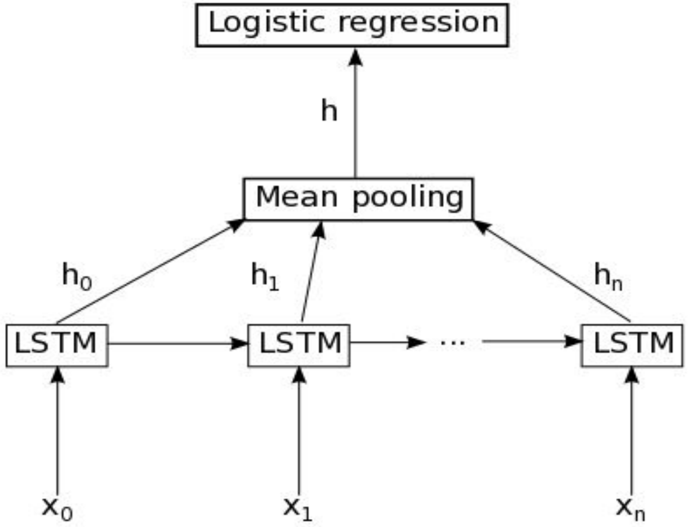
Technology

- **Deep Learning Technology Used in our project:** Recurrent Neural Network (RNN) using the Long Short Term Memory (LSTM) implemented with Theano

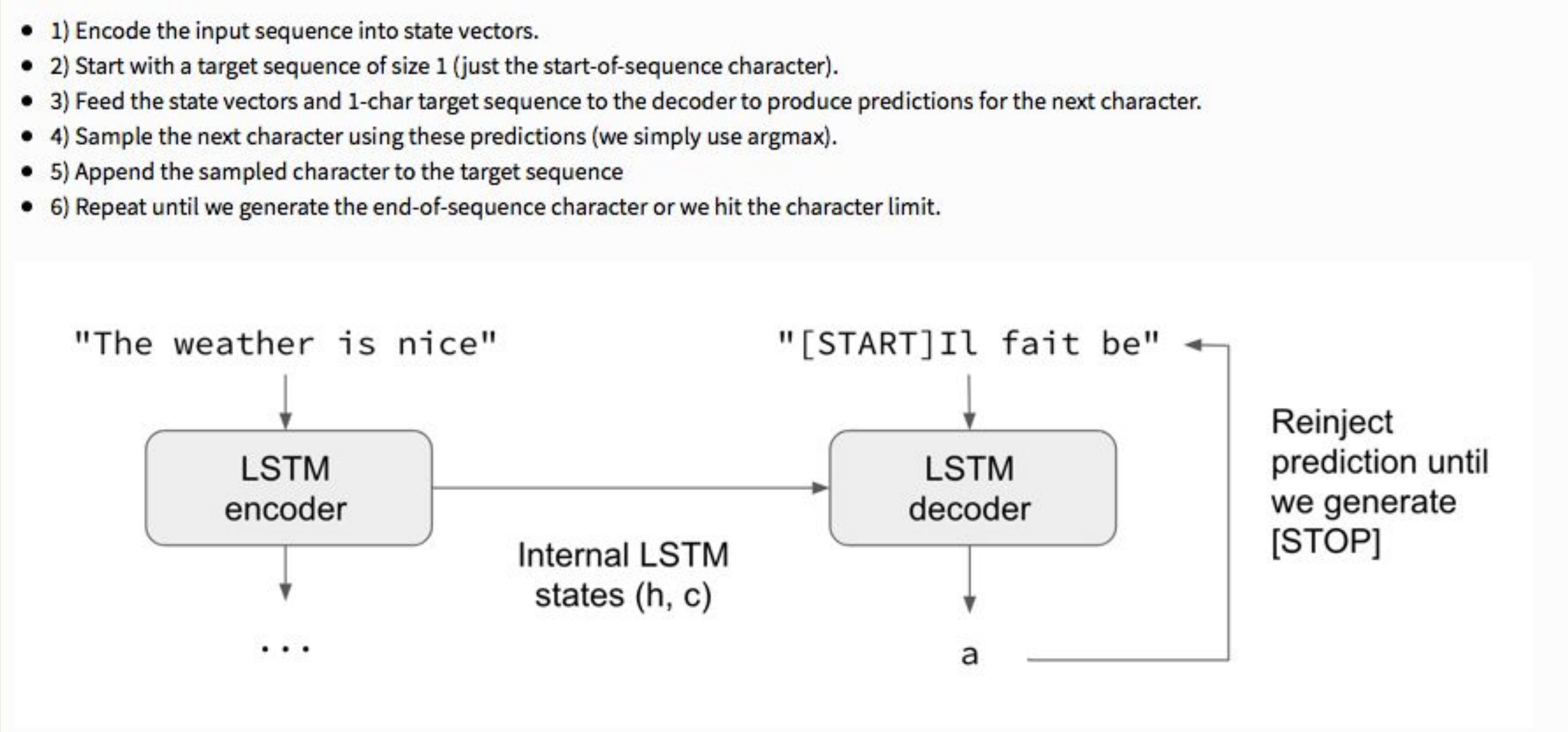
- **Github code has other library used:**
- import tensorflow as tf
- from keras.models import Sequential, load_model
- from keras.layers import Dense, Activation
- from keras.layers import LSTM, Dropout
- from keras.layers import TimeDistributed
- from keras.layers.core import Dense, Activation, Dropout, RepeatVector
- from keras.optimizers import RMSprop
- Etc..

Long Short Term Memory Architecture





LSTM Model applied to unknown sequence

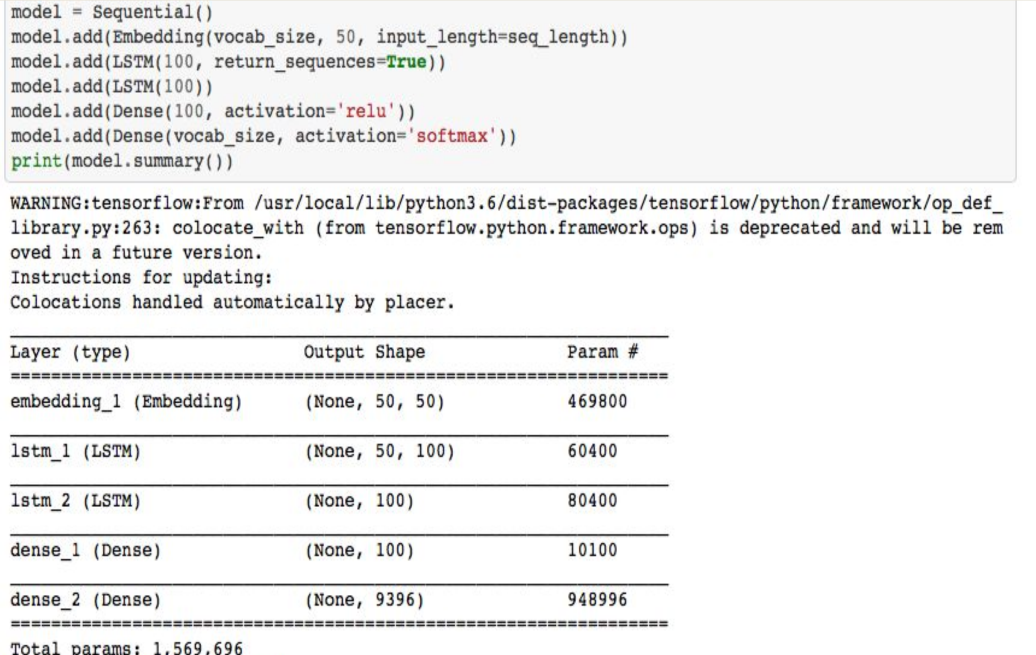


- 1) Encode the input sequence into state vectors.
- 2) Start with a target sequence of size 1 (just the start-of-sequence character).
- 3) Feed the state vectors and 1-char target sequence to the decoder to produce predictions for the next character.
- 4) Sample the next character using these predictions (we simply use argmax).
- 5) Append the sampled character to the target sequence
- 6) Repeat until we generate the end-of-sequence character or we hit the character limit.

"The weather is nice"
LSTM encoder
...
Internal LSTM states (h, c)
"[START]Il fait be"
LSTM decoder
a
Reinject prediction until we generate [STOP]

Model Summary

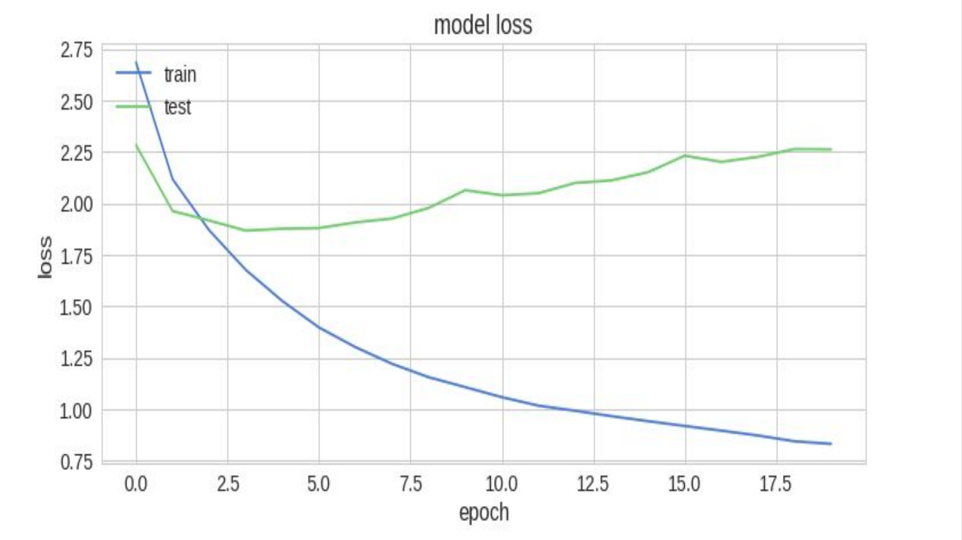
- ❏ Used ReLu activation function in dense layer since it’s non-linear and less costly
- ❏ Used Softmax
- ❏ We trained 100 epochs.
- ❏ Accuracy: 73.68%
- ❏ Cross-Entropy loss: 1.5

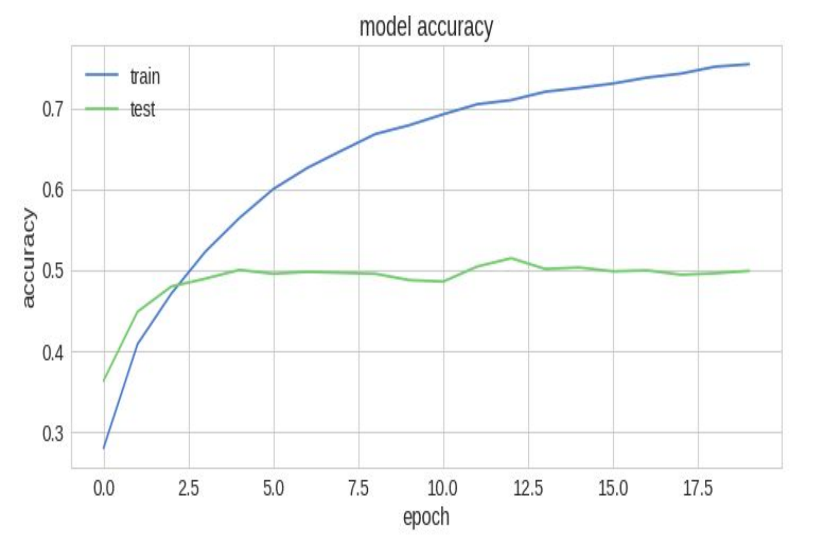


```python
model = Sequential()
model.add(Embedding(vocab_size, 50, input_length=seq_length))
model.add(LSTM(100, return_sequences=True))
model.add(LSTM(100))
model.add(Dense(100, activation='relu'))
model.add(Dense(vocab_size, activation='softmax'))
print(model.summary())
```

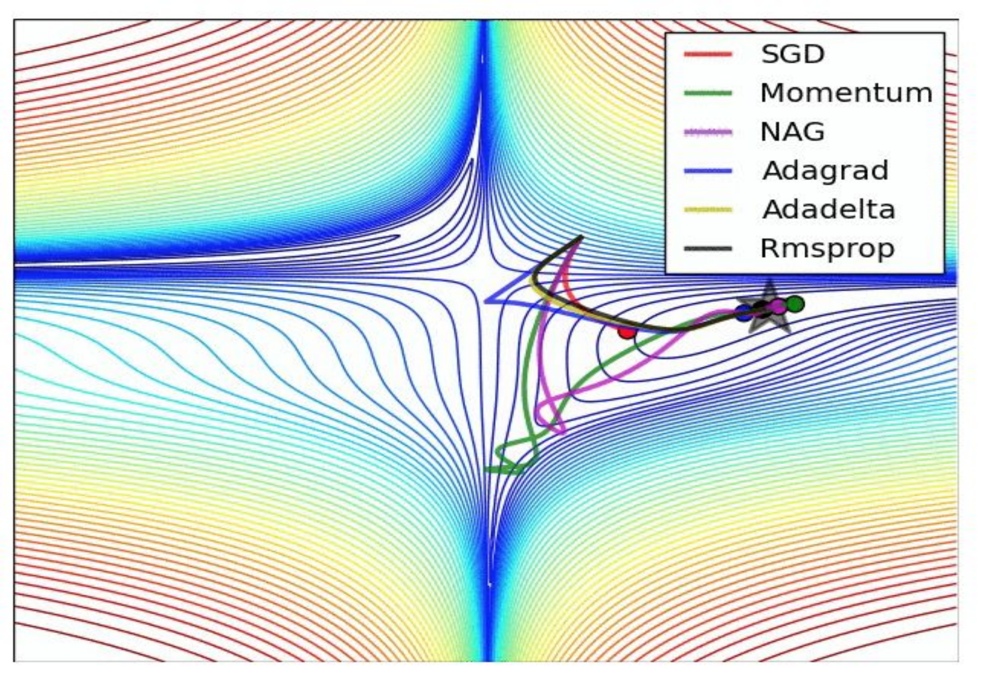
```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/op_def_
library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be rem
oved in a future version.
Instructions for updating:
Colocations handled automatically by placer.
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_1 (Embedding)      (None, 50, 50)            469800    
_________________________________________________________________
lstm_1 (LSTM)                (None, 50, 100)           60400     
_________________________________________________________________
lstm_2 (LSTM)                (None, 100)               80400     
_________________________________________________________________
dense_1 (Dense)              (None, 100)               10100     
_________________________________________________________________
dense_2 (Dense)              (None, 9396)              948996    
=================================================================
Total params: 1,569,696
```

LSTM Model Performance



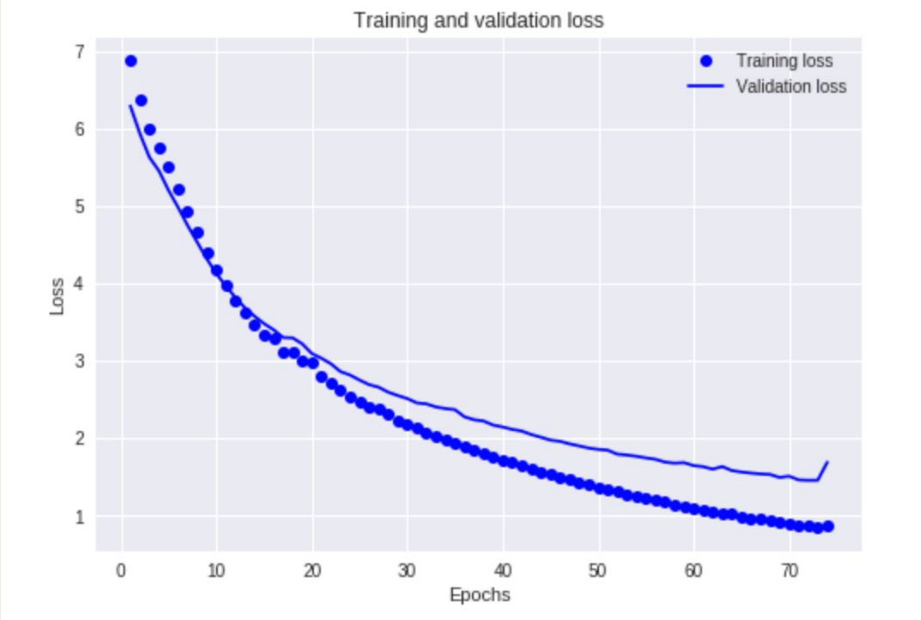


Keras Optimizer



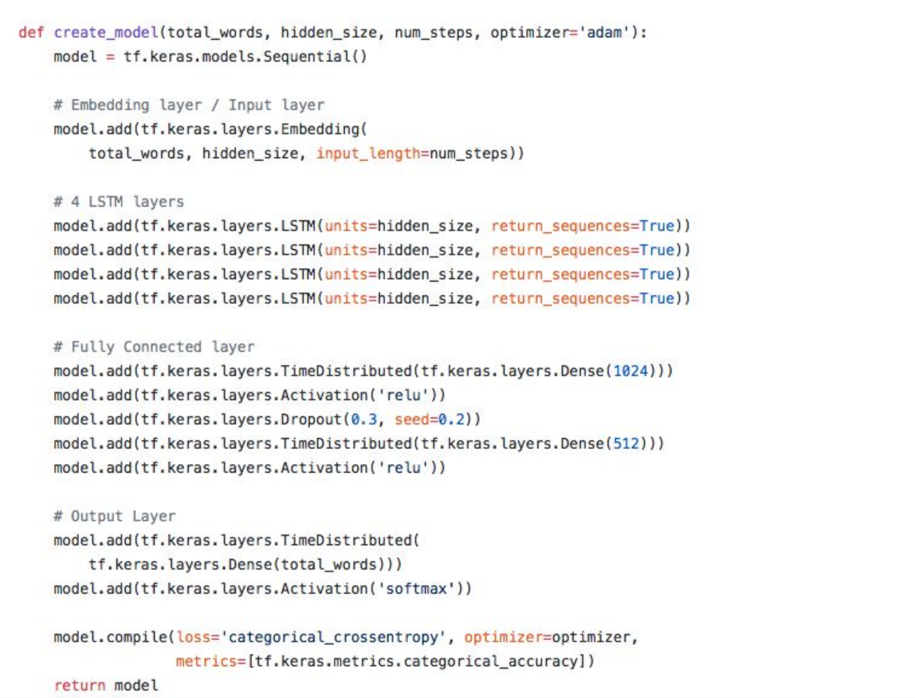
We chose RMSprop & Adam for deep neural nets

Auto Complete Model #1: Sentence prediction



We used RNN and Long short memory algorithms and trained an auto email complete model.
Our model can generate 50 words of email.

Auto Complete Model #2: word prediction



```python
def create_model(total_words, hidden_size, num_steps, optimizer='adam'):
    model = tf.keras.models.Sequential()

    # Embedding layer / Input layer
    model.add(tf.keras.layers.Embedding(
        total_words, hidden_size, input_length=num_steps))

    # 4 LSTM layers
    model.add(tf.keras.layers.LSTM(units=hidden_size, return_sequences=True))
    model.add(tf.keras.layers.LSTM(units=hidden_size, return_sequences=True))
    model.add(tf.keras.layers.LSTM(units=hidden_size, return_sequences=True))
    model.add(tf.keras.layers.LSTM(units=hidden_size, return_sequences=True))

    # Fully Connected layer
    model.add(tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(1024)))
    model.add(tf.keras.layers.Activation('relu'))
    model.add(tf.keras.layers.Dropout(0.3, seed=0.2))
    model.add(tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(512)))
    model.add(tf.keras.layers.Activation('relu'))

    # Output Layer
    model.add(tf.keras.layers.TimeDistributed(
        tf.keras.layers.Dense(total_words)))
    model.add(tf.keras.layers.Activation('softmax'))

    model.compile(loss='categorical_crossentropy', optimizer=optimizer,
                  metrics=[tf.keras.metrics.categorical_accuracy])
    return model
```