Personalized Sales Email Generation

Team - Soumya Gupta, Ting Wang, Samuel Lin, Aman Tripathi, Tejas Baindur

Introduction

Have you reached out to an unknown person and asked for some favour (job referral etc.) out of the blue? What has been the success rate? Surely it would not have been very high.
Now imagine trying to sell a product or service via email! The odds are not great unless you put in a lot effort in researching the person you are trying to sell to. Over the course of this project we tried to automate this laborious process with a focus on B2B Sales Lead generation.

Summary of the journey

We explored multiple approaches and datasets, in order to meet the requirements of the project -

1. Define the value of sender and his/her email (detailed below)
2. For low value emails, we choose using **Seq2seq** algorithm and build automatic email generation system(detailed below). After enormous training, emails will be replied flexibly and automatically with the pretrained seq2seq model.
3. For high value emails, we decide using **Template based approach**(detailed below), which was suggested by Prof Ikhlaq Sidhu as a preferred approach to the problem statement. We found **Chatterbot**, a python library, with simple NLP training it can help select the right template based on keywords and sentiment analysis of emails received.

Work Done

1. How to define the value of sender and his/her email

We define the professional, personal and company persona in order to understand the value of the client. We started with LinkedIn scraping using Selenium Webdriver. We successfully scraped 1000+ profiles. We used this data to score client into 'high/medium/low' value clients based on different weightage given to characteristics like title held in the company, level of education, years of experience and experience relevance. This enable us to determine if our code will directly respond to an incoming email or ask the user for manual intervention, when a high-value client is identified.

2. Automatic Email Generation with Seq2seq Approach

Dataset selection:

Initially, we tried to use Enron email dataset to train a seq2seq model, but found matching reply email to the original email is challenging. Also considering the various length of emails, there won't be enough samples in each bucket of length. Thus, we turned to using another popular dataset --Movie Dialogue Corpus, provided by Cornell University.

Data cleaning:

First, we splitted the dialogue into questions and answers according to their position in the dialogues--the answer of first conversation is also the question of next conversation. Using the length filter (from 5 to 50 words), we have got 16,000 dialogues. To simplify the training process, all sentences are padded into length of 50.

Seq2seq model building:

We build 2 LSTM layers instead. At first, the model was trained without pre-trained word embeddings, which didn't provide satisfying replies. So we embedded each word into 100 dimension vectors using GloVe, provided by Stanford, to see the benefit of word embedding. The embedded questions are input for first LSTM, where output of a 300-dimension vector is generated for each question, which is the summary of the question as well as the input for the second LSTM layer. The sentences generated by these 2 LSTM layers are compared to the right answer. Using GPU(TITAN X), it took 1 week to train 16,000 dialogues for 160 epochs. The loss decreased from 8 at beginning to 0.03.

Email Generation

By using impalib and email libraries in python, we can get the sender's email address, name, timestamp, and the subject and body of email. The email will be replied using the subject and email address, the greeting of the email will be in format of "Good morning/ afternoon/ evening, sender", changing according to timestamp and sender's name. The email body will generate from the trained Seq2seq model.

We have created a test account: chatbot2017fall@gmail.com. If executing the code(fetch_email.py), it will start polling the inbox for unread emails every 5 seconds, which can be adjusted by time.sleep(). From testing, we find the reply can be generated and sent in several seconds automatically.

3. Template based approach with sentiment analysis:

The Dataset:

We used the Enron email dataset for the purposes of sentiment analysis. The dataset was obtained from Kaggle. The dataset has 517401 rows consisting of the file name and the email message as two columns. A sample of the email message file is given below:

Data preprocessing:

Most of the data above was irrelevant to our analysis. We extracted the data from the message file using python tools pandas, sys, os and email into a pandas dataframe. The emails were then sorted in ascending order of datetime. All the columns except for the subject and the body of the email (or content) were removed.

After extracting the relevant data, rows with NaN values were removed, the content of the emails was cleaned and irrelevant characters removed using nltk and regex.

Data Classification:

The 'content' data was manually classified into 4 categories as follows:

1. Positive: The emails with an overall positive connotation.

2. Neutral: Generic emails like information, notices, etc
3. Negative: The emails with an overall negative connotation
4. Enquiry: The emails which 'ask' for details and information

Sentiment analysis and reply:

The manually labelled dataset was used for training the random forest classifier model using 50 estimators. Based on the trained model, when there is an incoming mail, it can be fed into the model to predict the sentiment. Based on the predicted sentiment, conditional statements are used to select a pre-drafted reply.

Future Steps

Reinforcement Learning:

The seq2seq model for replying emails precisely is challenging inherently. We can look for reinforcement learning algorithm to improve this. Whenever there is human intervention or editing of the outgoing emails, both the incoming email and outgoing email (or sentiment of incoming email) can be fed back to the training dataset. This will cause the model to get more accurate and agile over time specifically suited to the user.

Template-based replying:

Once we have email dataset from one specific company or industry, we can use our mastered NLP skills to train with chatterbot in Python, which can analyze emails and select the pre-written template.

Collaboration

Collaboration: The project was done under the guidance of Maverick.ai CEO and CTO, Yaron Oren and Kevin Wu, respectively. The mentors were extremely helpful in providing relevant reading resources and brought us up to the speed. Within team we employed a democratic model of decision making with each member's input taken into account for the direction of the project.

Ting Wang has trained the seq2seq model with keras, and wrote python scripts that polls emails and reply them by calling the API of the trained keras model.