Email Subject Line Generation with Transformers

Mackenzie Lee[1] and Rochelle Li[1]

[1]*School of Information, University of California, Berkeley, CA 94720,*
4 December 2021

In an increasingly remote work environment, multiple mediums for communication have emerged. A modern focus on innovative direct messaging platforms like Slack have become frontrunners in recent years, but a staple form of communication has remained constant: email. The subject line of an email is the first thing one reads and is the deciding factor for determining priority, importance, and context. This paper explores the use of transformers using T5 for the task of Subject Line Generation and results show that T5 can outperform LSTMs.

INTRODUCTION

A good email subject does not necessarily summarize the email itself, but rather conveys the importance and topic of the email body. As humans, we have different ways to interpret text and convey importance often leading to miscommunication and emails are no exception [9]. This presents an interesting challenge of generating email subject lines given an email body. By automating this process, we aim to explore whether machines can more successfully standardize how email subjects are written leading to better communication and prioritization.

This paper discusses using the transformer T5 model to generate email subject lines given an email body. This is not a straightforward summarization task, since email subject lines often are not simple summaries of the emails. Subject lines can often direct action or needs.

This paper focuses on new emails and not replies, email forwards, or email threads.

BACKGROUND

The idea of Subject Line Generation (SLG) emerged in 2019 using multiple neural networks including reinforcement learning approaches combined with Long short-term memory (LSTM) neural networks to break down the task of SLG into multiple components [1]. This work had achieved strong results with a ROUGE-L F1 score of 23.44 against the original subjects as reference. The ROUGE-L F1 score is useful for evaluating summarization of texts, especially in comparing an automatically produced summary against human-produced references. While there is evidence that suggests automatic metrics like ROUGE do not correlate well with human judgement [4], we also chose to use this score as a comparison against the state of the art results.

Prior work had solely focused on email summarization. There has been prior work done with news headline generation, which is the closest work to SLG, but email subject lines are typically much shorter than news headlines and can convey different meaning and intention leading to further complications [5]. We aim to use transfer learning applying the leverage of T5 and its exposure to multiple tasks to the task of SLG. By fine-tuning T5 by training it on a specific email dataset with subjects, we believe that T5's pretraining and transformer architecture will be more flexible and allow it to better capture the nuance and context that email subjects contain.

DATA

The data for this project was the Annotated Enron Subject Line Corpus (AESLC). This dataset is a special subset of the Enron dataset [8] and only the first emails in threads that were at least 3 sentences and 25 words long were kept in order to have enough information to generate a subject line and avoid duplicates. The email data consisted of the email body as well as the subject line as a label. There is subjectivity in what a good subject line should look like and some of the labelled emails may have actual subject lines that could be improved. For our purposes, we assume that the labelled subject lines are good examples of subject lines and will be the standard for the model to learn.

The dataset includes annotated subject lines that were generated from Amazon Mechanical Turk workers. We decided to select one of these annotations and use those labels to train the model to compare the test results to see if third party annotations would lead to better subject lines evaluated with a ROUGE-L F1 score. The reasoning behind this is that perhaps these third party subject lines may be objectively more useful for training T5 than the ones written by the person writing the email.

APPROACH/METHODS

Baseline

Our baseline model was the T5-base pretrained Transformer run on the set of emails with the original subjects with the baseline hyperparameters (see code). The ROUGE-L F1 score was 27.86, which is higher than the prior state of the art. This high of a ROUGE-L F1 score indicates T5 as a more suitable model than the LSTMs used prior.

Modeling

To improve baseline results, we experimented with different data cuts and tokenizers. We trained models on the In addition, we changed the tokenizer experimenting with the T5 tokenizer based on SentencePiece and two versions of the BERT tokenizer WordPiece. All the emails in the dataset are English, so we wanted to experiment with using a tokenizer that required pre-tokenization using spaces to separate words and see if the results would improve. In addition, we tried BERT tokenizers trained on cased and uncased text. The logic behind this being that emails can often contain acronyms and context-specific vocabulary, especially in corporate settings. Case-based tokenizers may be able to differentiate specific vocabulary and improve SLG.

Model Tuning

We experimented with two different T5 sizes, three different hyperparameters (learning rate, batch size, and epochs) and three different optimizers. Despite existing literature suggesting that hyperparameter tuning does not improve model results, we saw improved results by varying three parameters [2]. Using different combinations of these, we calculated the ROUGE-L F1 score as shown in the tables in the Results section (TABLE I, TABLE II, and TABLE III)

RESULTS

The results of out most relevant runs are summarized in the tables in this section (TABLE I, TABLE II, and TABLE III).

Models 23 and 31 were trained with BERT tokenizer. Model 18 was trained with a different set of train data (see more detail in Fun Experiment subsection in the Model and Error Analysis section).

Model	Batch Size	Epoch	ROUGE-L F1
01	2	2	27.86
03	3	2	28.75
16	8	20	24.21
17	5	20	22.91
23	5	25	10.77
28	8	1	29.97

TABLE I: *T5-base models with* $LearningRate(LR) = 1e-4$ *and Optimizer = Adam. Model 23 was trained with BERT tokenizer.*

Model	LR	Batch Size	Epoch	ROUGE-L F1
02	1e-4	20	2	26.06
04	1e-4	8	1	29.62
05	1e-4	20	50	28.05
13	1e-4	10	50	25.07
15	5e-5	10	50	27.01
18	1e-4	20	50	22.26
24	1e-5	20	50	27.23
25	5e-4	20	50	19.61
26	1.5e-4	20	50	25.06
27	2e-4	20	50	23.22
31	1e-4	8	1	4.45

TABLE II: *T5-small models with Optimizer = Adam. Model 31 was trained with BERT tokenizer.*

Model	LR	Batch Size	Epoch	ROUGE-L F1
06	1e-4	20	2	9.52
07	1e-4	20	25	16.17
08	1e-4	10	25	18.27
09	1e-2	10	25	21.07
10	1e-2	15	25	25.34
11	1e-1	15	25	0
14	1e-4	10	50	20.50
29	1e-4	8	1	9.99

TABLE III: *T5-small models with Optimizer = SGD*

MODEL AND ERROR ANALYSIS

T5 Model Size

The T5 model was pretrained on the c4 dataset [11]. In our experiments, we used two different sizes of the model: T5-base and T5-small. We were limited to the two smallest sizes of T5 due to memory constraints. We started off with T5-base in the baseline model and were curious about how significantly the size of the model affects our evaluation metric. We expected the larger model size to bring with it a clear cut advantage over the smaller model.

However, we found that with learning rate = 1e-4, batch size = 8, and optimizer = Adam, the T5-base model 28 only performed 1.2% better than the T5-small model 04 (ROUGE-L F1 score for T5-small is 29.62 and for T5-base is 29.97). The base model took significantly longer to train compared to the small model. We did not compare the two model sizes with another set of constant parameters, so our experiment does not show that the base model consistently only marginally outperformed the small model.

FIG. 2: *Training Loss of Model 24: Learning rate = 1e-5, ROUGE-L F1 score = 27.63*

Learning Rate

To see how the learning rate affects the ROUGE-L F1 score, we trained models with different learning rates while holding constant batch size = 20, number of epochs = 50, optimizer = Adam, and T5 model = T5-small. In the plot of learning rate vs ROUGE-L F1 score (FIG. 1), the relationship between the two are linear in our selected range of the domain ($LR \in [1e{-}5, 3e{-}4]$). In the specified range, increasing the learning rate typically decreases the ROUGE-L F1 score.

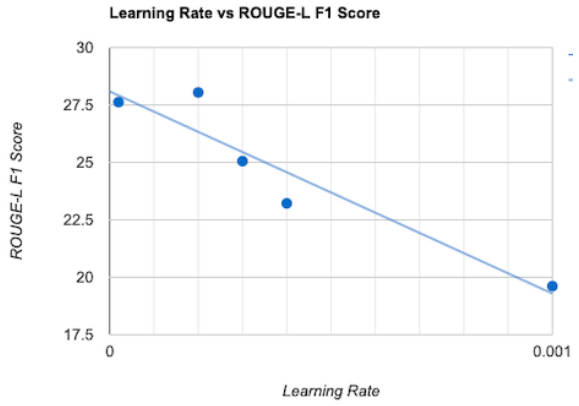

FIG. 1: *Plot of how ROUGE-L F1 score changes with learning rate for models with batch size = 20, number of epochs = 50, optimizer = Adam, and T5 model = T5-small.*

The training is outputted in the training logs. Model 24 had a learning rate of $1e{-}5$ with ROUGE-L F1 score of 27.63 and model 25 had a learning rate of $5e-4$ with ROUGE-L F1 score of 19.61. Comparing FIG. 2 and FIG. 3, we can see that smaller loss does not equate to higher ROUGE-L F1 score.

FIG. 3: *Training Loss of Model 25: Learning rate = 5e-4, ROUGE-L F1 score = 19.61*

Epochs

We expected a greater number of epochs to improve model performance because more epochs means more training time, so we did some experiments to test our hypothesis. Using the T5-base model with learning rate = 1e−4, batch size = 8, and optimizer = Adam, model 28 with 1 epoch (ROUGE-L F1 score = 29.97) performed 23.8% better than model 16 with 20 epochs (ROUGE-L F1 score = 24.21). In contrast, using the T5-small model with learning rate = 1e−4, batch size = 20, and optimizer = Adam, model 05 with 50 epochs (ROUGE-L F1 score = 28.05) performed 7.6% better than model 02 with 2 epochs (ROUGE-L F1 score = 26.06).

Batch Size

We expected a greater batch size to improve model performance because more data is seen between updates. We did different experiments for the different model sizes. Using the T5-base model with learning rate = 1e−4, epochs = 8, and optimizer = Adam, model 16 with batch size of 8 (ROUGE-L F1 score = 24.21) performed 5.6% better than model 17 with batch size of 5 (ROUGE-L F1 score = 22.91). Similarly, using the T5-small model with learning rate = 1e−4, epochs = 25, and optimizer = SGD, model 10 with batch size of 15 (ROUGE-L F1 score = 25.34) performed 20.2% better than model 09 with batch size of 10 (ROUGE-L F1 score = 21.07). Our hypothesis was consistent with the results for T5-small and T5-base.

Optimizers

Optimizers are mathematical functions that modify the weights of a network given the gradients and other information. Stochastic gradient descent (SGD) is vanilla gradient descent with batch size of 1. Adaptive Moment Estimation (Adam) computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients, Adam keeps an exponentially decaying average of past gradients [10].

To compare the performance of optimizers Adam and SGD, we trained model 13 with Adam and model 14 with SGD, while holding learning rate = 1e−4, batch size = 10, epochs = 50, and T5 model = T5-small as constants. Model 13 (Adam) with ROUGE-L F1 score = 25.07 performed 22.3% better than model 14 (SGD) with ROUGE-L F1 score = 20.50. Our results are in line with existing literature suggesting that Adam outperforms SGD [12].

Tokenizers

All our models were trained with the T5 tokenizer except for model 23 and model 31, which were trained with BERT. To compare the performance of the T5 tokenizer and BERT, we trained model 31 with learning rate = 1e−4, batch size = 8, epochs = 1, optimizer = Adam, and T5 model = T5-small, which is consistent with model 04. Model 04 with ROUGE-L F1 score = 29.62 performed 566% better than model 31 with ROUGE-L F1 score = 9.99.

Best Performing

Our best performing model was model 28, which was trained with the T5 tokenizer, T5-base, learning rate = 1e−4, batch size = 8, epochs = 1, and optimizer = Adam. Model 28 with ROUGE-L F1 score = 29.97 performed 7.6% better than our baseline model (model 01) with ROUGE-L F1 score = 27.86. Compared to the baseline, model 28 had a decrease of 1 epoch and an increase in batch size by 6, but all else was constant.

Fun Experiment: Using Dev and Test data as the Train Dataset

For all our models, we held out the same subset of train data to use as the test data for predictions. As a fun experiment, we combined the dev and test data from the full ENRON dataset and used it as the train dataset for model 18. The test data we used to create this new train dataset is different from the test data we used for predictions, and there is no overlap. Model 18 was the same as model 05 except for the data it was trained on. Model 18 with ROUGE-L F1 score = 22.26 performed 20.6% worse than model 05 with ROUGE-L F1 score = 28.05. This could be due to the fact that model 18 was trained on less than a third of the amount of data as model 05.

CONCLUSION AND NEXT STEPS

Using T5 for SLG resulted in a high ROUGE-L F1 score that outperformed the prior state of the art using LSTMs. We were able to improve baseline results through different hyperparameters and T5 sizes, but, overall, our results show that T5 outperforms LSTMs in the task of Subject Line Generation. It is important to note that the T5 paper was published 3 months after the original SLG paper. Human comparison of T5 results compared to original subjects may also help reveal the true usefulness of the model.

Subject Line Generation remains a difficult topic due to the subjectivity in what a good subject line consists of. To further improve results, it would be useful to expand the emails to include a wider range of domains as well as find ways to incorporate sender and recipient identity.

BIBLIOGRAPHY

[1] Zhang, Rui and Tetreault, Joel. "This Email Could Save Your Life: Introducing the Task of Email Subject Line Generation." Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, arXiv:1906.03497, 2019, 446-456. https://aclanthology.org/P19-1043.pdf

[2] Smith, Leslie N.. "A DISCIPLINED APPROACH TO NEURAL NETWORK HYPER-PARAMETERS: PART 1 – LEARNING RATE, BATCH SIZE, MOMENTUM, AND WEIGHT DECAY." US Naval Research Laboratory Technical Report, arXiv:1803.09820, 2018. https://arxiv.org/pdf/1803.09820v2.pdf

[3] Raffel, Colin. Shazeer, Noam. Roberts, Adam. Lee, Katherine. Narang, Sharan. Matena, Michael. Zhou, Yanqi. Li, Wei. Liu, Peter J.. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." Journal of Machine Learning Research, 21, 2020, 1-67. https://arxiv.org/abs/1910.10683

[4] Zhang, Shiyue. Celikyilmaz, Asli. Gao, Jianfeng. Bansal, Mohit. "EMAILSUM: Abstractive Email Thread Summarization." arXiv:2107.14691. https://arxiv.org/abs/2107.14691

[5] Gu, Xiaotao. Mao, Yuning. Han, Jiawei. Liu, Jialu. Yu, Hongkun. Wu, You. Yu, Cong. Finnie, Danial. Zhai, Jiaqi. Zukoski, Nicholas. "Generating Representative Headlines for News Stories." WWW '20: Proceedings of The Web Conference 2020, 1773-1784. https://arxiv.org/abs/2001.09386

[6] Briggs, James. "The Ultimate Performance Metric in NLP." Towards Data Science, https://towardsdatascience.com/the-ultimate-performance-metric-in-nlp-111df6c64460. Accessed October 2021.

[7] Zhang, Rui. GitHub, https://github.com/ryanzhumich/AESLC. Accessed October 2021.

[8] Cohen, William W. "Enron Email Dataset." School of Computer Science, Carnegie Mellon University, https://www.cs.cmu.edu/ enron/. Accessed October 2021.

[9] "Email Subject Line Generator." keap, https://keap.com/tools/email-subject-line-generator. Accessed October 2021.

[10] Park, Sieun. "A 2021 Guide to improving CNNs-Optimizers: Adam vs SGD." Medium, https://medium.com/geekculture/a-2021-guide-to-improving-cnns-optimizers-adam-vs-sgd-495848ac6008. Accessed December 2021.

[11] "t5-small." Hugging Face, https://huggingface.co/t5-small/blob/main/README.md. Accessed December 2021.

[12] Choi, Dami. Shallue, Christopher J. Nado, Zachary. Lee, Jaehoon. Maddison, Chris J. Dahl, George E.. "On Empirical Comparisons of Optimizers for Deep Learning." ICLR 2020 Conference, 2019. https://arxiv.org/pdf/1910.05446.pdf