Twitter Opinion Summarization and Trend Analysis

Harsh Mishra (hmishr3@uic.edu) and **Jeet Paresh Mehta (jmehta27@uic.edu)**
University of Illinois at Chicago

Abstract

The goal of this project is to summarize a given tweet or a given set of tweets using Abstractive summarization, and to analyze the sentiments of those tweets through the produced summary. The main use case behind implementing the model was to summarize twitter threads as well as to get a concise summary of tweets for any given topic. This project was done as part of the CS 521 course project at the University of Illinois at Chicago.

1 Introduction

These days tweet threads, a series of 2 or more tweets connected to each other, are quite popular. These threads can often be very lengthy in size, sometimes equivalent to reading an article. Our project proposes to specifically address these twitter threads, by using Abstractive Summarisation. Our model can also be utilized to get an overall summary of any given topic/keyword. One of the use cases being, getting the summary of fan opinions on a given sports player, during a given time period. Our model could also help analyse the sentiments of fans and see if it tends to change overtime. The automated workflow that we provide will help the user to extract tweets containing keywords of their liking, from their defined time period. The user also has the option of directly providing the text that they would like to be summarized. Below we outline our methodologies that lead to the building of our project, the papers that helped understand the process of text summarisation, provide the results that we obtained and also describe the hurdles that we faced. Our code base and the documentation to use our models can be found at https://github.com/harshm16/NLP_project. The models can also be directly downloaded from our Hugging Face repo, https://huggingface.co/harshm16

2 Literature Review

In NLP, there are 2 approaches for Text summarisation. Extractive summarization and Abstractive summarization. In the former technique, summary is produced by choosing a subset of sentences in the given sentences. Say for instance we have 6 sentences but there is one such sentence which is of most importance in the entire text so this sentence will be chosen according to extractive summarization. The technique involves using frequency driven methods. Getting the Word Probability, using TFIDF (Term Frequency-Inverse Document Frequency) (Wikipedia contributors, 2022c) are some of the common approaches used.

While in abstractive summarization, we construct summary from learning from the most important words in the original text. So in this process there could be cases where we might encounter a word not previously seen in the source text. This way of potentially coming up with new relevant phrases can also be seen as paraphrasing. It includes using heuristic approaches to train the model to understand the whole context of the source text and generate a summary based on that understanding. Since our goal is to not just attain the most important sentence/words amongst the input sentences, but to generate a concise summary of the source text, we opt to use Abstractive summarization in our project.

(Liu et al., 2017) propose an adversarial process for abstractive text summarization. Similar to GANs (Goodfellow et al., 2014) they simultaneously train a generative model and a discriminative model. The generator takes the raw text as input and predicts the abstractive summary. While the discriminator attempts to distinguish the generated summary from the ground truth summary.

(Suhara et al., 2020) helped us explore the self supervised training approach for gaining summaries in domains where there is a lack of Golden summaries. Rather than sending in the opinions directly into the transformer (Vaswani et al., 2017) model, they initially extract opinion phrases and then send that into the transformer. This way they can use those phrases to reconstruct an abstractive summary of their own.

We also attained some useful knowledge of twitter opinion prediction from (Zhu et al., 2020). They model users' tweet posting behaviour as a temporal point process. They use the user's historical tweets and the tweets posted by their neighbours to predict the posting time and the stance label of user's next tweet.

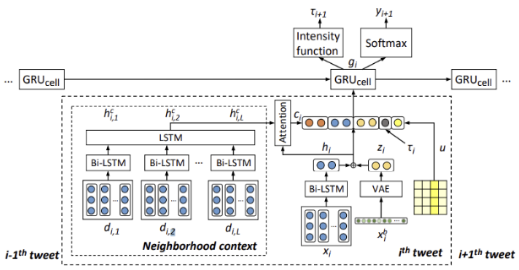
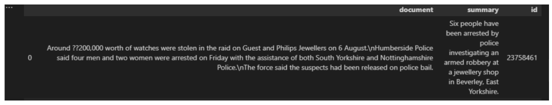


Figure 1: The architecture of the Neural Temporal model used by (Zhu et al., 2020)

They use a stacked Bi-LSTM and LSTM (Staudemeyer and Morris, 2019) layer to extract context/features from neighborhood tweets. These neighborhood features are inputted into an attention layer. A Variational Autoencoder (Kingma and Welling, 2019) is used to extract the topic from the bag of words of the user's tweet. In order to extract neighborhood features relevant to the user's tweet, the user topic and their tweet features are sent as an input to the attention layer. The output from the attention layer is then concatenated with the user's tweet, tweet topic, the time between the user's last tweet and a unique user id. This combined representation is then sent to the GRU (Chung et al., 2014) cell. The intensity function of this GRU cell then predicts the future posting time of the user, and the Softmax function predicts the stance label of that future tweet.

(Jiang et al., 2021) helped us understand the use of Graph Neural Networks in the Natural Language Processing domain. They developed a novel Attention-based Relational Graph Convolutional Network (ARGCN) for Target-Oriented Opinion Words Extraction. It works on exploiting syntactic information of a given sentence using dependency graphs. They use BIO tagging scheme (Wikipedia contributors, 2022a) to generate a target-aware representation and then use the syntactic dependency graphs to create the Graphical structure. They also showed the importance of reshaping the Syntactic Dependency Graphs. They reversed the dependency edge when it linked the target words and its type was nominal subject or direct object. This was done so that information of the subject or object could flow through the predicate. They also highlighted the importance of using sequential models like LSTMs, to encode the sequential information, which could possibly be overlooked in Graphical Networks.

3 Methodology

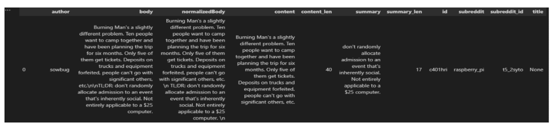


	document	summary	id
0	Around ??200,000 worth of watches were stolen in the raid on Guest and Philips Jewellers on 6 August.\nHumberside Police said four men and two women were arrested on Friday with the assistance of both South Yorkshire and Nottinghamshire Police.\nThe force said the suspects had been released on police bail.	Six people have been arrested by police investigating an armed robbery at a jewellery shop in Beverley, East Yorkshire.	23758461

Figure 2: A row from the Extreme Summarization dataset

3.1 Datasets

We work with 2 datasets, Extreme Summarization (XSum) (Narayan et al., 2018) and Reddit Dataset (V"olske et al., 2017). The Xsum dataset contains news articles and their respective one line summary, see figure 2. While the latter includes reddit posts and their summary, see figure 3. Since our downstream task is to summarize Tweets, therefore we emphasized on choosing a dataset that involved social media data. As there were no datasets of sufficient size, with tweets and their golden summaries, we chose the Reddit dataset to act as a proxy for the same. As seen in figure 3 the features included author, body, normalized version of the body, content, summary, content and summary length, subreddit name, subreddit_id and title. We only used 2 of these features during training, Content and Summary.



	author	body	normalizedBody	content	content_len	summary	summary_len	id	subreddit	subreddit_id	title
0	sowbug	Burning Man's a slightly different problem. Ten people want to camp together and have been planning the trip for six months. Only five of them get tickets. Deposits on trucks and equipment forfeited, people can't go with significant others, etc.\n\nTL;DR: don't randomly allocate admission to an event that's inherently social. Not entirely applicable to a $25 computer.	Burning Man's a slightly different problem. Ten people want to camp together and have been planning the trip for six months. Only five of them get tickets. Deposits on trucks and equipment forfeited, people can't go with significant others, etc. \n TL;DR: don't randomly allocate admission to an event that's inherently social. Not entirely applicable to a $25 computer. \n	Burning Man's a slightly different problem. Ten people want to camp together and have been planning the trip for six months. Only five of them get tickets. Deposits on trucks and equipment forfeited, people can't go with significant others, etc.	40	don't randomly allocate admission to an event that's inherently social. Not entirely applicable to a $25 computer.	17	c401hri	raspberry_pi	t5_2syto	None

Figure 3: A row from the Reddit dataset

3.2 Primary Model

The 2 datasets with their Golden summary provided were used to finetune Google's Text-To-Text Transfer Transformer (T5) model (Raffel et al., 2020) for the summarisation task. The model was already pre-trained on a on a multi-task mixture of unsupervised and supervised tasks. The datasets used by them include: Common Crawl's web crawl corpus (Raffel et al., 2019), Wikipedia dataset (Karpukhin et al., 2020), Sentence completion datasets (Roemmele et al., 2011), Question answering (Khashabi et al., 2018), (Clark et al., 2019), (Zhang et al., 2018) and many more. We use the latest checkpoint of the T5-small model, downloaded from Hugging Face.

3.3 Finetuning Specifications

The reddit dataset contained 3,848,330 posts, out of which we only took 10% of the data due to compute restrictions. In order to map the training data to the expected test data, we chose posts consisting of at least 20 words and at max 50 words. An additional filter was put on the summary as well, posts with summaries containing at least 10 words and at max. 30 words were chosen. No such modifications were made on the XSum dataset. The dataset was then split into: Test : Train : Validation :: 80 : 10 : 10. The datasets were divided into batch sizes of 4 and the models were trained for 1 epoch each, with the learning rate 2e-5 and weight decay 0.01. The max. input length was specified as 1024 and the max. target length as 128 tokens.

3.4 Workflow

We offer users the ability to extract tweets of their need. Once the tweets are ready, we use cardiffnlp's (Loureiro et al., 2022) pre-trained model to analyze the sentiment of those tweets and characterize them into Positive, Negative and Neutral. Their model is based upon Roberta (Liu et al., 2019) and is finetuned on analysis with the TweetEval benchmark (Barbieri et al., 2020). We then concatenate tweets with the same sentiment to make each tweet at least 40 word long. After each input tweet is at least 40 word long, we use general pre-processing steps to remove hyperlinks, emojis and twitter ids. We finally send in these pre-processed tweets as the input to the Abstractive Summarisation models. The users can also extract keywords from their tweets by using the Keybert (Grootendorst, 2020) library.

The final pipeline includes taking tweets in the form of text input, analysis of sentiment on those and then finally summarising the sentiment-wise aggregated input. The trend analysis of the changes in the sentiments is then done.

3.5 Model Utilisation

Our models can be utilized in two ways. The jupyter notebooks used to train the models have code blocks which can directly be used to test the summarisation capabilities of the model on any text input. Examples of this method can be seen in Figures 6 and 7. The other way of utilizing the model is to provide keywords and start/end time as search parameters for tweets. The "Extract_tweets" code will be utilized to extract those tweets. The "Tweet_sentiment" code will then be used to analyze the sentiments of those tweets. These tweets will then be passed in as the input to the summarisation models. We also provide the user a way to compare the summaries produced by our two models. The "Keyword_extraction" code can be utilized to act as a validation proxy for the generated summary. The more the number of keywords the generated summary contains, the better the summary.

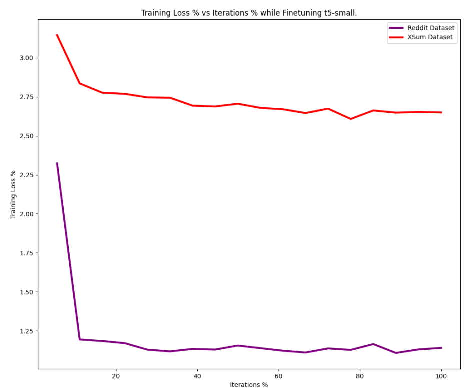


Figure 4: Training Loss vs Iterations for both the models

4 Evaluation and Results

For evaluation of our models we use Rouge Scores (Wikipedia contributors, 2022b). ROUGE-1 refers to the overlap of unigram (each word) between the system and reference summaries. ROUGE-2 refers to the overlap of bigrams between the system and reference summaries. Whereas, ROUGE-L refers to the Longest common sub-sequence. It takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams.

Metric	Xsum Dataset	Reddit Dataset
Rouge1	29.0994	15.7252
Rouge2	8.2844	2.906
RougeL	22.9664	12.9949

Figure 5: Rouge scores for both the models.

As seen in Figure 5, we see that the summary from the XSum trained model had higher Rouge scores compared to the model trained on the reddit dataset. This signifies that the summaries produced by the XSum model was able to retain more information. The low Rouges scores are not an issue, as in a summarisation task you would expect the loss of information, going from the source text to its summary. We tried hypertuning our parameters by increasing the number of epochs and altering the learning rate, and eventually ended up with the specifications mentioned in section 3.3. Since we used the pre-trained T5 model, there was a restriction placed on the max. input length the max. target length of the tokens.

```python
model_checkpoint = "finetune\\t5-small-finetuned-xsum\\checkpoint-51000"
```
✓ 0.2s Python

```python
input_tweet = """BREAKING: Twitter is in the final stretch of negotiations about a sale to Elon Musk and
could reach a deal as soon as Monday. The social media company is working to hammer out terms of a transaction and
could reach an agreement as soon as Monday if negotiations go smoothly, according to a person with knowledge of the matter """
```
✓ 0.3s Python

```python
inputs = tokenizer(input_tweet, max_length=1024, return_tensors="pt")

summary_ids = model.generate(inputs["input_ids"],min_length = 10, max_length= 50)
tokenizer.batch_decode(summary_ids, skip_special_tokens=True, clean_up_tokenization_spaces=False)[0]
```
✓ 0.4s Python

```
'Twitter is in negotiations about a sale to Elon Musk and could reach a deal as soon as Monday, according to a person with knowledge of the matter.'
```

Figure 6: Summary generated by the XSum model.

Figures 6 and 7 show the output summaries for the given text input, using the latest checkpoints from our models. The models, as trained on different datasets, generate varying summaries for the same input. The users also have the option to play around with the length of the output summary. The results shown in figures 6 and 7 are generated with the minimum summary word length set to 10 and the maximum set to 50.

```python
model_checkpoint = "finetune\\t5-small-finetuned-reddit_10per\\checkpoint-23500"
```
✓ 0.4s Python

```python
input_tweet = """BREAKING: Twitter is in the final stretch of negotiations about a sale to Elon Musk and
could reach a deal as soon as Monday. The social media company is working to hammer out terms of a transaction and
could reach an agreement as soon as Monday if negotiations go smoothly, according to a person with knowledge of the matter """
```
✓ 0.4s Python

```python
inputs = tokenizer(input_tweet, max_length=1024, return_tensors="pt")

summary_ids = model.generate(inputs["input_ids"].cuda(),min_length = 10,max_length=50)
tokenizer.batch_decode(summary_ids, skip_special_tokens=True, clean_up_tokenization_spaces=False)[0]
```
✓ 0.9s Python

```
'Twitter is working to hammer out terms of a sale to Elon Musk and could reach a deal as soon as Monday if negotiations go smoothly.'
```

Figure 7: Summary generated by the Reddit model.

As mentioned in section 3.5, we compared the performance of our models using the number of Keywords found, as its qualitative proxy. On over 10k tweets containing 40 words or more, 96.1 % of the summaries generated by the Reddit model contained at least 1 keyword, compared to 95.6 % for the XSum model. Our hypothesis behind doing this comparison was that the model trained with the reddit dataset was more likely to perform better on social media data. Since the results are so close to each other, we think that the method fails to decisively differentiate between the models. Finding better ways to qualitatively compare the two summarization models is something that we would like to explore in the future as well.

5 Discussion and Conclusions

Tweets in general are to the point and short. Summarizing a single sentenced tweet doesn't make much sense. The biggest hurdle that we came across was defining the minimum size of a tweet that could be summarized. After some experimental runs we ended up concatenating tweets with less than 40 words. The side effect of this approach is seen when our automated workflow is used. As tweets with same sentiments but no common context can be concatenated together, and end up having a messy summary.

Deciding the size/word length of generated summaries was the other challenging task. Even though the user has control of increasing or decreasing the number of words that they would like in their summary, we decided that our summary would by default be in between 10-50 words. Increasing the summary length, especially in situations where the length of the input tweet was considerably short (50 words) led to repetition of sentences in the output summary.

We also saw that in the instances when the input sentences were short or had little common context, the abstractive summarisation performed similar to Extractive summarisation. That is, we ended up getting one of the existing sentences in the input text as the summary. This issue was also overcome by adjusting the minimum and maximum word length of the generated summary.

Overall, we feel that we were successful in reaching our objective of summarising the given tweet by using Abstractive summarisation methods and automating the process of utilisation of our models.

6 Future Work

Our model currently produces a single line summary for a given input, so using it to summarize a whole thread of tweets in one go would lead to loss of much information. The obvious enhancement to this would be to return a multiple line summary for a thread of tweets, in a single call to the model.

In the process of development we also tried ways to generate artificial Golden summaries. Using GPT-3 (Brown et al., 2020) to generate text from the keywords generated by Keybert (Grootendorst, 2020) was a technique we tried out, but failed. As the text generated lacked context and was no where similar to the actual source text. In the social media domain, especially twitter, golden summaries are hard to attain. Thus finding ways to obtain model generated Golden summaries is something that should be explored more.

As of now, our Github repository contains all the code that can be run through an IDE to use our models. One of our future goals is to expose the models through the Gradio app interface. The gradio app when ready, can be found in the Github repository mentioned in section 1.

7 Acknowledgment

We are very pleased to present to one and all our research project on Twitter Opinion Summarisation and Trend Analysis. We are very thankful to University of Illinois at Chicago for supporting our ideas and giving us an opportunity to express them. We are very thankful to our guide Dr. Natalie Parde, Assistant Professor, Computer Science Department Department, University of Illinois at Chicago, for her continuous guidance, support and feedback throughout the course of the project.

References

Francesco Barbieri, Jose Camacho-Collados, Leonardo Neves, and Luis Espinosa-Anke. 2020. Tweeteval: Unified benchmark and comparative evaluation for tweet classification.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *CoRR*, abs/1905.10044.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Junfeng Jiang, An Wang, and Akiko Aizawa. 2021. Attention-based relational graph convolutional network for target-oriented opinion words extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1986–1997, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Diederik P. Kingma and Max Welling. 2019. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative adversarial network for abstractive text summarization.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and José Camacho-Collados. 2022. Timelms: Diachronic language models from twitter. *CoRR*, abs/2202.03829.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Melissa Roemmele, Cosmin Bejan, and Andrew Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning.

Ralf C. Staudemeyer and Eric Rothstein Morris. 2019. Understanding lstm – a tutorial into long short-term memory recurrent neural networks.

Yoshihiko Suhara, Xiaolan Wang, Stefanos Angelidis, and Wang-Chiew Tan. 2020. OpinionDigest: A simple framework for opinion summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5789–5798, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Michael V"olske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark. Association for Computational Linguistics.

Wikipedia contributors. 2022a. Inside–outside–beginning (tagging) — Wikipedia, the free encyclopedia. [Online; accessed 28-April-2022].

Wikipedia contributors. 2022b. Rouge (metric) — Wikipedia, the free encyclopedia. [Online; accessed 28-April-2022].

Wikipedia contributors. 2022c. Tf–idf — Wikipedia, the free encyclopedia. [Online; accessed 28-April-2022].

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *CoRR*, abs/1810.12885.

Lixing Zhu, Yulan He, and Deyu Zhou. 2020. Neural temporal opinion modelling for opinion prediction on twitter.