WhatsApp Chat Summarization

Andrea Auletta
andrea.auletta@studenti.unipd.it

Davide Baggio
davide.baggio.1@studenti.unipd.it

Marco Bernardi
marco.bernardi.11@studenti.unipd.it

Marco Brigo
marco.brigo@studenti.unipd.it

Sebastiano Sanson
sebastiano.sanson@studenti.unipd.it

Abstract

This research project in Natural Language Processing (NLP) focuses on the development of an automated system for the summarization of chat messages, applicable to both group and individual conversations. The objective is to generate concise summaries of conversations, thereby obviating the need for users to review all messages individually. This application is especially pertinent in contexts where chat logs proliferate quickly, offering significant benefits in terms of time savings and cognitive load reduction for users.

1. Introduction

In the era of digital communication, chat platforms have become fundamentals in our daily interactions, whether they are personal or professional. This new type of transmission favors real-time messages exchange which allows users to engage in conversations that can quickly accumulate extensive chat logs. The huge volume of messages, especially in active group chats formed by dozens if not hundreds of users, often makes it difficult for users to stay updated without dedicating substantial time to review each message. This challenge highlights the need for an efficient method that enables users to quickly extract essential information from chat conversations.

This research project addresses this need by developing an automated system for summarizing chat messages with the goal to generate concise summaries, reducing the need to sort through all messages. By doing so, users can quickly obtain the main points of discussions, significantly saving time and reducing cognitive load.

The significance of this application is evident in various contexts: for instance, in corporate environments, project teams often heavily rely on group chats to coordinate tasks, share updates and exchange miscellaneous informations; similarly, social groups frequently engage in dynamic conversations. In both scenarios, an automated summarization system can enhance productivity and ensure important information is not overlooked.

This research aims to explore and implement state of the art NLP techniques to create an effective summarization system, focusing on the datasets used, the manipulation of them in order to obtain always a coherent data structure, how we obtained new examples to test the performances and the results we got after all.

TO DO: overview of our main results!

2. Datasets

In this section, we explore datasets essential for training and evaluating automated summarization systems for chat messages: an ideal dataset should have a variety of conversation styles and contexts to ensure the robustness and versatility of the summarization models. The datasets should contain real-life or realistic dialogues annotated with concise and accurate summaries. Additionally, the data should reflect diverse communication patterns, including informal and formal registers, and incorporate common conversational elements such as slang, emojis, eventual typos, the use of voice messages and images. We present below two notable datasets, SAMSum and DialogSum (both obtained by Hugging Face repository), which fulfill these criteria and provide rich resources for advancing the field of conversational AI.

2.1. SAMSum

The SAMSum [2] dataset contains about 16,000 messenger-like conversations with summaries splitted in:

- training: 14,732;
- validation: 818;

- test: 819.

Conversations were created and written down by linguists fluent in English: they were asked to create them similar to those they write on a daily basis, reflecting the proportion of topics of their real-life messenger conversations. The style and register are diversified - conversations could be informal, semi-formal or formal, they may contain slang words, emoticons and typos. Then, the conversations were annotated with summaries. It was assumed that summaries should be a concise brief of what people talked about in the conversation in third person. The SAMSum dataset was prepared by Samsung R&D Institute Poland and is distributed for research purposes.

2.2. Validation Dataset

To rigorously assess the performance of the selected models within our specific use case, we constructed a testing dataset by generating new examples derived from authentic WhatsApp conversations.

2.2.1 Data Collection

WhatsApp provides a robust functionality for exporting chat conversations in a .txt file format. This exported file encompasses all exchanged messages, complete with timestamps and sender names. Additionally, the file includes various media types, such as images, voice messages, and videos. This comprehensive dataset is instrumental for rigorous evaluation as it reflects real-world conditions and the diverse communication forms encountered in practical applications.

2.2.2 Data Preprocessing

The exported .txt file underwent a meticulous preprocessing phase to extract the pertinent information, including message content and sender details. This information was structured into a Python dictionary as follows:

```
dialogues = [
    {
        'text': "Hello, how are you?",
        'id': 0,
        'golden_summary': "A greeting"
    }
]
```

For conversations containing images or voice messages, the filenames of these media were embedded within the text field to facilitate subsequent processing by replacing the filenames with detailed descriptions for images and transcriptions for voice messages. The models utilized for generating these descriptions and transcriptions are discussed in the following sections. We also scraped out irrelevant information, like the date-time, in order to match the same structure used in the training dataset mentioned in the previous sections

This preprocessing approach ensures that the dataset is both comprehensive and structured, enabling effective analysis and model evaluation.

3. Models Involved

3.1. BART

BART [5] is a denoising autoencoder that maps a corrupted document to the original document it was derived from. It is implemented as a sequence-to-sequence model with a bidirectional encoder over corrupted text and a left-to-right autoregressive decoder.
In the architecture there are 6 layers both in the encoder and decoder (while the large model uses 12 layers in each). Each layer of the decoder performs cross-attention over the final hidden layer of the encoder. This pretraining enables BART to effectively learn robust representations of text.
BART is trained by corrupting documents and then optimizing a reconstruction loss between the decoder's output and the original document.
The part that interests us is the fine-tuning one, this model can be used in several ways for downstream applications like: sequence classification, token classification, sequence generation and machine translation tasks.

3.2. FLAN-T5

FLAN-T5 [1] is an enhanced version of the T5 model (Text-To-Text Transfer Transformer) that has been fine-tuned on more than 1,000 additional tasks, aiming to improve the performance on natural language understanding and generation tasks. It comes in various sizes, from 80 million parameters (Flan-T5-Small) to 11 billion parameters (Flan-T5-XXL). Due to our limited computational resources, we have chosen the base model, which includes 248M parameters.
The T5 model follows the Transformer architecture's encoder-decoder structure, which is crucial for handling text-to-text tasks and is characterized by:

- Encoder: processes the input embeddings and generates a sequence of hidden states. Each layer of it consists of a multi-head self-attention mechanism followed by a feed-forward neural network. The former allows the model to focus on different parts of the input sequence when encoding each token, while the latter consists of two linear transformations with a ReLU activation in between;

- Decoder: takes these hidden states and generates the output text, one token at a time, using autoregressive generation. It's similar to the encoder, but it includes

also an additional layer of attention over the encoder's outputs to ensure that the prediction for a particular token can only depend on the known outputs before it.

Finally, the output is generated by a softmax function that converts the decoder's hidden states into probabilities reflected on the vocabulary in natural language.

The comparison between T5 and FLAN-T5, both in the base version, highlights how the fine-tuned one outperforms the former model in every relevant benchmark for our scope:

- **MMLU** (Massive Multitask Language Understanding): it evaluates the performance across a wide range of tasks, subjects and disciplines, assessing the ability to understand and generate human language by testing them on various questions.
  - Direct propmpting: 25.7 - 35.9
  - CoT: 14.5 - 33.7
- **BBH** (Big-Bench Hard): is a collection of challenging tasks designed to evaluate the performance on difficult, nuanced, and often high-level natural language understanding tasks.
  - Direct propmpting: 27.8 - 31.3
  - CoT: 14.6 - 27.9
- **TyDiQA** (Typologically Diverse Question Answering): it evaluates the performance of question-answering systems across a diverse set of languages.
  - Direct prompting: 0.0 - 4.1

3.3. Multimedia models

Because chat messages often include multimedia contents, we have decided to use specific models to convert this type of data into text that can be summarized by the previously cited model. In particular, we transform images and voice messages, which are very frequently used nowadays. Since our project regards text chat summarization, we will not focus into the technical details of how the following models work in this paper. This approach ensures that our main objective is clearly addressed without overcomplicating the discussion with the intricate workings of the underlying models.

3.3.1 Florence-2 Large

To ensure that the images were processed by the summarization model, it was necessary to convert them into detailed textual descriptions. For this task, we employed Florence-2 by Microsoft [4], an advanced vision foundation model that utilizes a prompt-based approach to address a wide array of vision and vision-language tasks. Florence-2 is designed to interpret text prompts as task instructions and produce the desired textual outputs, including captioning, object detection, grounding, and segmentation. This model is built upon the extensive FLD-5B dataset, which comprises 5.4 billion annotations across 126 million images, thereby enhancing its multi-task learning capabilities. The sequence-to-sequence architecture of Florence-2 allows it to perform effectively in both zero-shot and fine-tuned scenarios, establishing itself as a robust and competitive vision foundation model.

3.3.2 Whisper

In addition to processing images, it was necessary to convert voice messages to text. For this task, we utilized Whisper by OpenAI [3], an automatic speech recognition (ASR) system. Whisper is trained on 680,000 hours of multilingual and multitask supervised data collected from the web. The extensive and diverse dataset enhances the system's robustness to accents, background noise, and technical language, which is particularly beneficial for our use case, as voice messages are often of suboptimal quality. Furthermore, Whisper supports transcription in multiple languages and translation from those languages into English.

4. Metrics

4.1. ROUGE

We evaluated the results provided by different NLP models using the ROUGE set of metrics, which stands for Recall-Oriented Understudy for Gisting Evaluation. We chose this specific metric because it is currently the most effective way to assess the performance of automatic summarization systems. It compares the generated summary with one or more human made references by measuring the overlap of n-grams in them.

In particular, we have used the following metrics:

- **ROUGE-1**: as the number suggests, it evaluates the overlap of unigrams, meaning that it compares the single words between the summaries;
- **ROUGE-2**: this time it evaluates the overlap of bigrams, that are two consecutive words;
- **ROUGE-L**: this one instead evaluates the longest common subsequence, meaning that it considers the structure of the sentences and the word order, but not requiring the words to be consecutive;
- **ROUGE-LSum**: it is related to the previous one, but it uses a slightly different calculation method since it

applies the ROUGE-L's method at the sentence level and then aggregates all the results for the final score. This metric is seen as more suitable for tasks where sentence level extraction is valuable such as extractive summarization tasks.

Each metric is calculated as follows:

- **Precision**: it measures the accuracy of the positive predictions made by the model. It is the ratio of correctly predicted positive instances to the total predicted positive instances

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

- **Recall**: it measures the ability of the model to find all relevant positive instances in the dataset. It is the ratio of correctly predicted positive instances to the total actual positive instances.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

- **F1-Score**: is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall, which is useful when the dataset has an uneven class distribution or when the cost of false positives and false negatives is different.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The value returned for each ROUGE metric is the F1-Score, since it provides a balanced measure of a model's performance by combining both precision and recall. By doing so, it avoids scenarios where a model might have high precision but low recall values, or vice versa. This is especially important when both false positives and false negatives carry significant costs.

4.2. Human evaluation - Fleiss' Kappa

We evaluated the results of the model on the WhatsApp chats in this way: given the 12 chats we had, each member of the group voted if the summary was good or not. The final results were obtained by using the Fleiss' Kappa metric, which is a statistical measure for assessing the reliability of agreement between a fixed number of raters when assigning categorical ratings to a number of items or classifying items. It is defined as follows:

- Proportion of annotations for which all the members agree:

$$P_o = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{N(N-1)} (\sum_{j=1}^{N} n_{ij}(n_{ij} - 1)) \qquad (1)$$

where:

  - N is the number of raters;
  - n is the number of items;
  - k is the number of categories;
  - n_{ij} is the number of raters who assigned the i-th item to the j-th category.

- Agreement expected by chance:

$$P_e = \sum_{j=1}^{k} p_j^2 \qquad (2)$$

- Proportion of all assignments which were to the j-th category:

$$p_j = \frac{1}{nN} \sum_{i=1}^{n} n_{ij} \qquad (3)$$

- Fleiss' Kappa:

$$K = \frac{P_o - P_e}{1 - P_e} \qquad (4)$$

5. Fine Tuning

Before the fine-tuning the two models we have chosen, gave to us the following results of the ROUGE metrics:

Model	R-1	R-2	R-L	R-LSum
BART	0.0	0.0	0.0	0.0
FLAN-T5	0.45	0.21	0.37	0.37

Table 1. ROUGE metrics before fine-tuning

After the complete fine-tune of the models on the SAMSum dataset, we obtained the following results:

Model	R-1	R-2	R-L	R-LSum
BART	0.0	0.0	0.0	0.0
FLAN-T5	0.47	0.23	0.37	0.37

Table 2. ROUGE metrics after fine-tuning

As we can see ....
For the re-use of the models once they have been fine-tuned, we uploaded them to the Hugging Face repository:

- BART: `https://huggingface.co/Seba213/summarizer-bart-cnn`
- FLAN-T5: `https://huggingface.co/Seba213/flan-t5-base-samsum`

6. Testing

Putting the preprocessing and the fine-tuned (nome modello) together, we tested the functioning in the practice on the custom dataset we created. The results are shown in the following table:

Here there is an example of the output of the model on a chat of the custom dataset:

Po	Pe	K
1	1	1

Table 3. Results of the human evaluation

```
    {
        'text': "Hello, how are you?",
        'id': 0,
        'golden_summary': "A greeting",
        'summary': "A greeting"
    }
```

7. Conclusions

References

[1] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[2] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *CoRR*, abs/1911.12237, 2019.

[3] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022.

[4] Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks, 2023.

[5] Mike Lewis Yinhan Liu Naman Goyal Marjan Ghazvininejad Abdelrahman Mohamed Omer Levy Ves Stoyanov Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.