Leveraging Emojis, Keywords and Slang for Enhanced Abstractive Chat Summarization

Alessandro Caruso
317028
Polytechnic of Turin
Turin, Italy
s317028@studenti.polito.it

Valerio Mastrianni
308781
Polytechnic of Turin
Turin, Italy
s308781@studenti.polito.it

Riccardo Moroni
320002
Polytechnic of Turin
Turin, Italy
s320002@studenti.polito.it

Ali Yassine
312920
Polytechnic of Turin
Turin, Italy
s312920@studenti.polito.it

***Abstract*—In this study, we introduce some novel extensions to enhance the task of abstractive chat summarization. Building upon previous research that highlighted the benefits of commonsense injection in the same task, our first extension explores the significance of emojis in dialogues and chat-like conversations. We examine how emojis can be a rich source of information, contributing to the generation of summaries that are both more accurate and contextually pertinent. Secondly, we investigate the impact of keywords injection, demonstrating their beneficial role in the context of dialogue summarization. Additionally, we introduce a preprocessing technique to handle slang effectively. The results from our framework are promising and lay the groundwork for future investigations in this field.**

I. Introduction

Dialogue summarization is a complex task, demanding the generation of concise and informative summaries for conversations. The abstractive method involves synthesizing key information using wording that may not be present in the original text. The inherent complexity of dialogues involves implicit information, shared knowledge, and nonverbal cues. In our research, we have concentrated on the intricate task of chat-like dialogues. When dealing with chats, the varying language styles and terminologies specific to different domains, where common words may have entirely different meanings, add to the complexity. Additionally, the widespread use of emojis in chat dialogues presents a challenge, as their interpretation can vary significantly among different user groups. These factors highlight the difficulty of creating effective summaries for chat-like dialogues, where context lies, not only in the words, but also in the unspoken and culturally specific cues. Our work aims to develop a model adept at handling these complexities and versatile enough to adapt to the evolving nature of chat-based dialogues. At present, models that utilize commonsense inferences for additional context are showing superior performance for abstractive dialogue summarization. Building on these frameworks, we aim to tackle the previously mentioned challenges with the following proposals:

1) Emoji Analysis: we initiate our approach by assigning conventional meanings to emojis, steering clear of any nuanced interpretations. Following this, we delve into an advanced emoji encoding technique. This technique involves creating Word2Vec (W2V) representations for emojis. The emojis are then replaced with words that closely resemble them in context. This technique is designed to learn and adapt these variations for each specific context, as different Word2Vec (W2V) models can be fine-tuned to suit domain-specific requirements, allowing for greater precision in understanding and interpreting emojis within varied contexts.

2) Keyword Extraction: we incorporate a keyword extraction method. This involves identifying key words that are relevant to the main topic of the dialogue. These extracted keywords are instrumental in guiding the summarization process, ensuring the focus is on the most pertinent information.

3) Slang Preprocessing: we address the challenge of slang by translating these expressions into more common language, making them easier for the model to process and understand.

To validate our enhancements, we first conducted a comparison with the results obtained using the classical commonsense injection approach with those after integrating our advancements. This comparative analysis aims to provide insights into the effectiveness and impact of our proposed techniques.

II. Related Works

A. Word2Vec

Word2Vec, developed by Tomas Mikolov and his team at Google in 2013, is a revolutionary approach in natural language processing [2]. It embeds words into a high-dimensional vector space, effectively capturing their semantic and syntactic relationships through contextual analysis. The essence of Word2Vec lies in its ability to discern nuanced word associations.

B. SICK - Commonsense Injection

The SICK framework (Summarizing with Injected Common Sense Knowledge) presented in the paper [3] offers a novel method for abstractive dialogue summarization. This framework distinguishes itself by utilizing commonsense knowledge inferences both as a unique input and a supervisory signal, enhancing the quality of dialogue summaries. The paper highlights the successful experimentation with the Comet model for generating common sense inferences, demonstrating its efficacy in dialogue contexts. SICK encompasses three main steps: the generation and selection of common sense inferences, training the model using these inferences, and applying

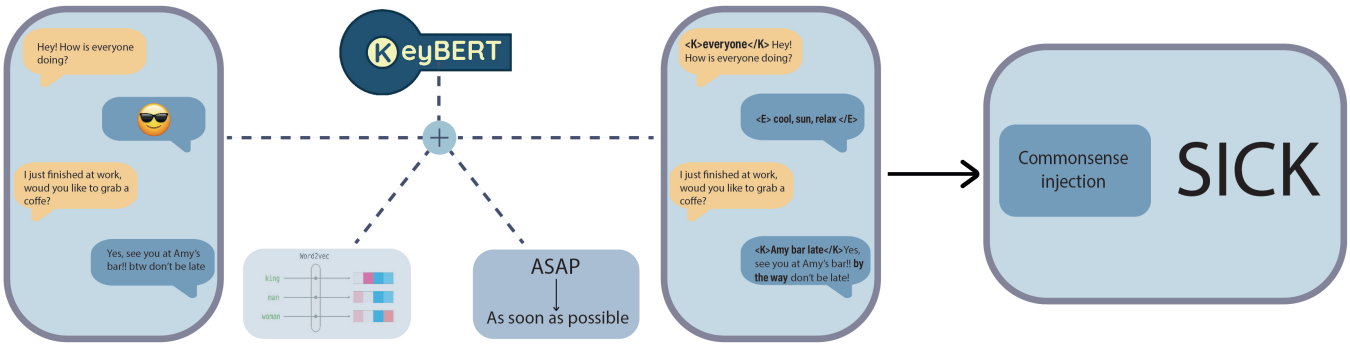

Fig. 1. Visualization of the proposed framework

a commonsense supervision mechanism. A notable achievement of the framework is the development of a similarity-based method for inference selection, which outperforms other variants in performance. The inclusion of common sense inferences as an additional input has been shown to significantly improve the model's capability in producing more accurate and nuanced dialogue summaries, representing a significant advancement in the field of dialogue summarization.

C. Keyword Extraction

Keyword extraction is a method that has been proved to improve the task of abstract summarization as shown in [5]. This method leverage on the usage of KeyBERT in order to retrieve the most important words in a given text. KeyBert is a keyword extraction tool grounded in a self-supervised contextual retrieval system that leverages BERT embeddings and cosine similarity. This system excels in pinpointing subphrases within a document that closely mirror the document's overall content. The process involves feeding the sentence S into BERT, generating a contextual feature vector W represented as $W = w1, w2, ..., w_n = BERT(S)$.

To derive the sentence embedding vector, the word vectors within a sentence are averaged. Subsequently, the method selectively identifies words in proximity to the sentence embedding vector to ensure that the extracted keyword accurately captures the essence of the sentence. The final step involves computing the similarity of the word embeddings to the sentence embedding using the cosine similarity metric: $Sim_i = \cos(w_i, W)$.

Here, Sim_i represents the cosine similarity between the word embedding vector w_i of a word i and the sentence embedding vector. Once the candidate keywords are identified, the tool further refines its output by applying the rule of adjacent keywords to extract meaningful keyphrases.

III. Proposed Framework

In this section, we detail our implementations. For a comprehensive overview of the model, please refer to Figure 1.

A. Task Description

Drawing inspiration from the structure utilized in the model presented in [3], our goal is to learn a mapping function $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{Y}$. This function transforms an input dialogue $\mathcal{D} = \{u_1, u_2, \ldots, u_n\}$, consisting of n utterances, into a final representation $\mathcal{Y} = \{y_1, y_2, \ldots, y_m\}$. This final representation is an abstractive summary composed of $m < n$ sentences. More precisely, we aim to extend our model to learn the mapping $\mathcal{M} : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is constructed as the cross-concatenation between the initial dialogue representation and commonsense:

$$\mathcal{X} = \mathcal{D} \oplus \mathcal{C} = \ldots ||u_i|| <\mathrm{I}> c_i <\!/\mathrm{I}> \ldots$$

Here, u_i represents the i-th utterance, composed by n words $w_1 \ldots w_n$, and c_i corresponds to the commonsense information associated with that utterance. In the subsequent steps, we will enhance this representation by incorporating additional information using the same cross-concatenation approach.

B. Emoji Analysis

In the context of emoji analysis, we examined two methods. In the first method, we utilized a dataset to map emojis to their respective aliases, which are non-deterministic in number. During this step, we made the simplifying assumption that users in chat dialogues employ emojis in accordance with their conventional meanings. However, it's essential to acknowledge that in reality, the use of emojis is much more intricate, and their meanings can vary significantly depending on the context and domain in which they are used. In this case, consider an utterance u_i composed of n words, where each word can be an emoji represented as e_k:

$$||u_i|| = w_1 \ldots e_k \ldots w_n$$

We obtain a new representation of u_i by substituting each emoji e_k with its list of aliases, denoted as

Fig. 2. Visualization of emoji embeddings in 2 dimensional space

$\langle alias_1, alias_2, \ldots \rangle$. These aliases are enclosed within special tokens <E> and </E>:

$$||u_i|| = w_1 \ldots \text{<E>} alias_1, alias_2, \ldots \text{</E>} \ldots w_n$$

The second approach does not involve selecting words to replace emojis from a pre-defined knowledge set. Instead, we aim to learn the most similar words to replace emojis using the Word2Vec (W2V) technique. To do this, we first train our model on a large corpus of Tweets. At this stage, we assumed a similarity in the usage of emojis between Tweets and chats. Subsequently, we fine-tune this W2V model based on our specific data. To evaluate the performance of our model, we reduce the embedding dimensions to 2 using Principal Component Analysis (PCA) and visualize the results with emojis only. As shown in Figure 2, it is notable that emojis with similar meanings are clustered closely together in the space, thereby demonstrating the effectiveness of this approach for emojis. Once we have obtained embeddings for words and emojis, we compute for each emoji the T most similar words $similar_1, \ldots, similar_T$, based on cosine similarity. Please note that we make this selection only among the most similar words and not emojis. These selected words are used in the same way as the aliases in the previous approach. For this second approach, the new representation of an utterance u_i is:

$$||u_i|| = w_1 \ldots \text{<E>} similar_1, \ldots, similar_T \text{</E>} \ldots w_n$$

The strength of this approach lies in its adaptability. Through fine-tuning, it can adjust to different domains and learn distinct representations for various contexts. This adaptability is crucial because emojis can take on different meanings based on the domain in which they are used. Furthermore, these representations are not language-dependent and can also provide emoji representations for text in other languages.

C. Keyword extraction

Integrating keywords into the process of abstractive summarization, especially in the context of dialogues, can significantly enhance the effectiveness of the summarization system. Keywords act as anchor points in the dialogue, helping the system to identify and retain the most important topics and themes. This is crucial in chats, where the topic can shift rapidly. By focusing on keywords, the system can generate summaries that are more contextually relevant and coherent as they provide clues about the speaker's focus. To address this issue, we chose to extract the most relevant keywords from each utterance u_i in the dialogue using a pre-trained keyword extractor, KeyBERT [6]. During this process, we identified the top keywords in each utterance by applying a threshold τ. This threshold ensures only the most pertinent words in a sentence were chosen, thereby avoiding the inclusion of words that do not contribute meaningful context to the summarization task. While we maintained this parameter as constant, it is possible to adjust and fine-tune it for different applications.

We obtain a new representation of u_i that includes the corresponding keywords, denoted as $<keyword_1, keyword_2, \ldots>$, the result is:

$$||u_i|| = \text{<K>} keyword_1, keyword_2, \ldots \text{</K>} \; w_1 \ldots w_n$$

D. Slang processing

Given that our use case involves chat conversations, it is essential to consider the informal language, including slang. As noted in [7], understanding the diversity in text data, which varies from formal language to slangs in social media, is crucial for effective summarization. These expressions are important as they can convey significant information that should be incorporated into our analysis. Slang often substitutes standard expressions, thus requiring special consideration in our study. In order to grasp the context provided by those words we employed an approach similar to the first one proposed for the emoji analysis. We utilized a dataset to map the slang inflection to their respective full form in order to insert those word in the final summary providing insights to the model. Unlike emoji analysis, where meaning can be highly contextual, slang meanings are often more fixed, as they are typically employed to express a concept in a more concise manner. For example, "ASAP" is used instead of "As soon as possible". Consequently, this mapping approach appears to be sufficient for our purposes.

The proposed new representations of the utterances, depending on the approach, are the ones used in the cross-concatenation process described in the subsection earlier.

IV. Experimental Setup

A. Dataset

We performed experiments on the SAMSum dataset [8]. SAMSum is widley used for abstractive dialogue summarization task. It consists of natural messenger-like conversations in English created by linguists with manually annotated summaries. This dataset contains emoticons or emojis in approximately 48% of the dialogues, making it suitable for our analysis.

B. Implementations details

We employed two evaluation metrics as: (i) ROUGE scores, which includes ROUGE-1, ROUGE-2, and ROUGE-L, for simplicity denoted as R-1, R-2, R-L respectively. These metrics compares word-level uni-gram, bi-gram and the longest common sequence overlap with respect to the golden summary [9]; (ii) BERTScore, denoted as B-S, it is a popular metric for text generation that computes the similarity score between generated and golden summary reference, using contextual embeddings [10].
In the keyword extraction of the second implementation, we use a threshold $\tau = 0.35$. This value was chosen in order to strike a good balance between relevant and not relevant words, without forcing KEYBert to insert some keywords which could not be end up to be relevant for the task. It is important to notice that, despite working well and improving the performances, the τ value could be further fine-tuned in order to obtain even better results.

Our baseline for comparison is the result obtained by the commonsense injection model SICK. In all of our experiments, we used a learning rate of 3e-6, a batch size of 32 and beam search with beam size of 20 when fine-tuning, as suggested in the original paper. It's important to note that, due to limitations in computational resources, we decided to use a lower number of epochs, specifically 4. These parameters are consistent across all our experiments. All the experiment are run on one V100 NVIDIA GPU provided by Google Colab.

TABLE I
COMPARISON OF PROPOSED MODELS

Model	R-1	R-2	R-L	B-S
Baseline	51.65	26.74	42.49	70.77
Emoji (mapping)	51.89	26.76	42.64	70.86
Emoji (W2V)	51.38	26.11	42.28	70.55
Emoji (W2V) + Keyword	51.92	**26.99**	**42.70**	**70.92**
Emoji (W2V) + Keyword + Slang	**52.01**	26.86	42.65	70.85

V. Results

The obtained results by different proposed models are displayed in table I. Starting from the first proposed implementation, the naive emoji-mapping approach slightly improves over all the metrics of the baseline. Moving to the W2V emoji-only implementation instead we observed slightly lower results compared to the baseline. This slight drop in performance is likely due to the fact that the dataset was not specifically designed for emoji processing. In these relatively structured chats, emojis are predominantly used with their original meanings, which makes our approach less suitable for this dataset, favouring the mapping-based one. Additionally, we trained our model using data from a large corpus of Tweets, and this might have caused a domain shift. We are proud that in a more real-world scenario, our W2V model can accurately infer the specific meanings that emojis have in particular contexts. In fact, the newly proposed approach offers a significant advantage compared to a one-to-one mapping approach for two main reasons: first, as discussed earlier, emojis have varying meanings across different groups of people, and it is impractical to assume a universally valid mapping for all; secondly, a mapping approach is language-specific, whereas our approach allows emojis to be learned for every language. When adding the keywords, we obtain the best results for all the metrics except R-1. This result is promising as it indicates that the model benefits from the introduction of keywords for the task of chat abstractive summarization. The best result for the R-1 metric of our model is obtained by leveraging all three proposed analyses, indicating that slang is also worth being analyzed.

The obtained results are promising and lay the groundwork for future investigations in this field, as they have demonstrate the benefits of the analysis of emojis, keywords and slang. Further implementation may include experiments with new datasets and conducting more computationally expensive tests, as well as analyzing performance across different domains and languages.

References

[1] Implementation code can be found at: GitHub Repository

[2] Tomas Mikolov, Kai Chen, Greg Corrado, & Jeffrey Dean (2013). *Efficient Estimation of Word Representations in Vector Space*. *arXiv:1301.3781*. Retrieved from https://arxiv.org/abs/1301.3781.

[3] Seungone Kim, Se June Joo, Hyungjoo Chae, Chaehyeong Kim, Seung-won Hwang, Jinyoung Yeo. *Mind the Gap! Injecting Commonsense Knowledge for Abstractive Dialogue Summarization*. Accepted at COLING 2022. *arXiv:2209.00930 [cs.CL]*. DOI: 10.48550/arXiv.2209.00930.

[4] Chongjae Yoo and Hwanhee Lee. *Improving Abstractive Dialogue Summarization Using Keyword Extraction*. *Applied Sciences*, vol. 13, no. 17, 2023, article 9771. https://www.mdpi.com/2076-3417/13/17/9771. DOI: 10.3390/app13179771.

[5] Chongjae Yoo and Hwanhee Lee (2023). *Improving Abstractive Dialogue Summarization Using Keyword Extraction*. *Applied Sciences*, 13, 9771. DOI: 10.3390/app13179771.

[6] Maarten Grootendorst (2020). *KeyBERT: Minimal keyword extraction with BERT*. Zenodo. Version v0.3.0. DOI: 10.5281/zenodo.4461265. Retrieved from https://doi.org/10.5281/zenodo.4461265.

[7] Diedrichsen, E. (2017). *Linguistic challenges in automatic summarization technology*. *Journal of Catalan Linguistics, 1*, 40-60. DOI: 10.4995/JCLR.2017.7787. Retrieved from https://doi.org/10.4995/JCLR.2017.7787.

[8] Gliwa, B., Mochol, I., Biesek, M., & Wawer, A. (2019). *SAMSum Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization*. *arXiv:1911.12237*. Retrieved from https://arxiv.org/abs/1911.12237

[9] Chin-Yew Lin (2004). *ROUGE: A Package for Automatic Evaluation of Summaries*. In *Text Summarization Branches Out*. Barcelona, Spain. Association for Computational Linguistics. URL: https://aclanthology.org/W04-1013. Pages: 74–81.

[10] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi (2019). *BERTScore: Evaluating Text Generation with BERT*. *arXiv e-prints*. Keywords: Computer Science - Computation and Language. DOI: 10.48550/arXiv.1904.09675. arXiv: 1904.09675. Primary class: cs.CL. URL: https://ui.adsabs.harvard.edu/abs/2019arXiv190409675Z.