Event Oriented Abstractive Summarization

Aafiya Hussain
K.J. Somaiya College of Engineering
aafiya.h@somaiya.edu

Talha Chafekar
K.J. Somaiya College of Engineering
talha.c@somaiya.edu

Grishma Sharma
K.J. Somaiya College of Engineering
grishma.sharma@somaiya.edu

Deepak Sharma
K.J. Somaiya College of Engineering
deepaksharma@somaiya.edu

Abstract

Abstractive Summarization models are generally conditioned on the source article. This would generate a summary with the central theme of the article. However, it would not be possible to generate a summary focusing on specific key areas of the article. To solve this problem, we introduce a novel method for abstractive summarization. We aim to use a transformer to generate summaries which are more tailored to the events in the text by using event information. We extract events from text, perform generalized pooling to get a representation for these events and add an event attention block in the decoder to aid the transformer model in summarization. We carried out experiments on CNN / Daily Mail dataset and the BBC Extreme Summarization dataset. We achieve comparable results on both these datasets, with less training and better inclusion of event information in the summaries as shown by human evaluation scores.

1 Introduction

Summarization is the process of giving an overview of a piece of text. This is done to reduce the amount of time required to understand a topic by eliminating information that is not as relevant to the topic. In abstractive summarization, the model tries to grasp the source text and produce a summary that consists of novel words and phrases. As the sentences produced are generated by the model, the redundancy in the final summary is significantly reduced as compared to extractive text summarization. This task of summarization is a complex one for humans as well. The difficulty of this task is due to the fact that summarization is fairly subjective. People may assign importance to parts of text differently. Thus, the main focus of one person's summary may be just a passing mention in someone else's summary. Another reason for this difficulty is that there has to be a balance between novel text and text taken from the source article. For abstractive summaries we want the model to understand the source text, and then represent it in a concise manner. This is a tricky balance to maintain as we want to achieve saliency, but we also want to avoid direct copying from the source text. We use transformers to carry out abstractive text summarization on the CNN / Daily Mail dataset (Nallapati et al., 2016) and Extreme Summarization dataset (Narayan et al., 2018). The transformer architecture we have used is BART (Lewis et al., 2020).

In this paper, we propose a system, which modifies the existing BART (Lewis et al., 2020) architecture by adding an additional event attention block. Events can be described as the sub-topics around which the news articles revolve. Identifying these events and adding them separately, along with source text, prompts the model to focus the summaries around these events. We perform keyphrase extraction using KeyBERT (Grootendorst, 2020) to extract important events from the source text and use these events for prompting our model to generate event-oriented summaries.

We achieve comparable results for ROUGE (Lin, 2004) and BERTScore (Zhang* et al., 2020) metrics for CNN / Daily Mail and XSum datasets, with the base variant of the BART model. Moreover, with the help of human evaluation, we quantify the extent to which our generated summaries are influenced by the events input to the model.

2 Background

2.1 Problem Statement

Given an input document $\mathbf{X} = \mathbf{x_1}, ..., \mathbf{x_n}$, we aim to generate a summary $\mathbf{Y}' = \mathbf{y}'_1, ..., \mathbf{y}'_m$ where $\mathbf{n}$ and $\mathbf{m}$ denote article and summary lengths respectively. The summary is generated in reference to $\mathbf{Y} = \mathbf{y_1}, ..., \mathbf{y_p}$ where $\mathbf{p}$ is the length of the ground truth summary. We make use of auxiliary input, i.e. event tokens $\mathbf{E} = \mathbf{e_1}, ..., \mathbf{e_b}$ consisting of $\mathbf{b}$ events,

Proceedings of the 19th International Conference on Natural Language Processing (ICON), pages 99 - 108
December 15-18, 2022 ©2022 Association for Computational Linguistics

Ground Truth: The rapper assaulted the photographer at Los Angeles International Airport in 2013. West apologized as part of the settlement, the photographer's lawyer says. **Generated summary:** Kanye West has settled a lawsuit with a paparazzi photographer he assaulted. Daniel Ramos had filed the civil suit against West after the hip-hop star attacked him and tried to wrestle his camera from him. **Events:** west has settled lawsuit, civil suit against west, ramos had filed the, photographer he assaulted
Ground Truth: New research finds direct link between exam stress and performance. London headteacher Michael Ribton says revision plans, flashcards and cram techniques can all help children prepare for the exam season. He advises parents that extra tuition and bribes shouldn't be necessary. **Generated summary:** Study by Lancashire's Edge Hill University and the University of South Australia found a direct link between anxiety and performance. Pupils who worry about their exam performance are more likely to do badly than those who are less anxious. **Events:** exam performance are more, worry about their exam, between anxiety and performance, exams and grades achieved
Ground Truth: Smoke from massive fires in Siberia created fiery sunsets in the Pacific Northwest. Atmospheric winds carried smoke from the wildfires across the Pacific Ocean. Smoke particles altered wavelengths from the sun, creating a more intense color. **Generated summary:** A fiery sunset greeted people in Washington Sunday. The deep reddish color caught Seattle native Tim Durkan's eye. **Events:** fiery sunset greeted people, siberia the dramatic sunset, reddish color caught seattle, sunset began showing up
Ground Truth: Villagers in Shangdong are seen using bags as long as six metres. It is becoming a common behaviour in some villages since 2011. Previous investigation suggested gas in the bag are often stolen. Gas carriers have little understanding of dangers claiming it to be safe. **Generated summary:** Residents from Lijin village in Dongying city carry the explosive in bags as long as six metres on rickshaws. Worried passers-by compared this behaviour to carrying a bomb on their backs. **Events:** the explosive in bags, stealing gas from large, gas this reckless behaviour, carrying bomb on their

Table 1: Few results for samples from CNN DailyMail dataset.

with each event having a fixed number of tokens **k**. We make use of seq2seq architecture, specifically a transformer encoder and decoder network, along with input prompting, generalized pooling and an additional event attention block which focuses on event embeddings to generate event-oriented summaries.

2.2 Sequence Models

Since abstractive summarization is a sequence based task, the initial application of deep learning models to abstractive summarization started with an attention based encoder and a sequence decoder making use of beam search (Rush et al., 2015). Since the decoder was not a recurrent model, later approaches to recurrent based summarization systems (Nallapati et al., 2016) performed better in generating summaries. The encoder mechanism consisted of an attention block attending to different encoder time steps, and a decoder RNN, which would take into account the encoder's attention outputs for the decoding step. A pointer generator network model (See et al., 2017) was introduced, which would dynamically decide whether to generate new tokens or to copy tokens from the article text, thus making the summarization model more factually correct. However, with the rise of transfer learning, language models like BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) gave a far better performance as compared to LSTMs for the same task.

2.3 Transformer Models

With the introduction of transformers (Vaswani et al., 2017), sequence to sequence tasks have become much easier using pre-training. Transformers were first used for training on machine translation tasks on the WMT 2014 English-French dataset (Bojar et al., 2014). Transformers outperformed previous models on the BLEU metric (Papineni et al., 2002). Consequently, the application of transformers to summarization was done by a BERT encoder used to feed embeddings to a transformer decoder (Zhang et al., 2019). A two stage mechanism, where masked language modelling as used in BERT (Devlin et al., 2019) is applied for refined word prediction in the later stage of the model. Raffel et al. (2020) pre-trained a transformer model on the C4 dataset, along with an analysis of different pre-training objectives such as prefix language modelling. PEGASUS (Zhang et al., 2020) follows the same pre-training objective as BERT, however, they introduce a summarization specific objective, i.e. to mask and generate sentences, similar to an extractive summary. Results indicated a significant increase in ROUGE scores with previous SOTA methods. Another training objective proposed was denoising in BART (Lewis et al., 2020), i.e. corrupting the input sequences with a range of operations including replacement, masking, text infilling, and sentence permutation.

3 Related Work

3.1 Event Extraction

Örs et al. (2020) use pre-trained transformer models, namely BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) for predicting if a pair of sentences point to the same event, and later use the prediction scores to capture the degree of relatedness between different sentence pairs. Xu et al. (2021) propose a graph-based model to capture the relation between different sentences and entity mentions. A tracker module is used which stores the global information about the extracted events, which can be used to query the stored information for interdependency relations. Rule-based systems (Ritter et al., 2012; Valenzuela-Escárcega et al., 2015) follow a syntactic and a word feature-based approach for extraction of events. A more general approach is used by Sun et al. (2021) where a multi-task training objective is followed over pre-trained language model embeddings for n-grams to capture both their informativeness and phraseness. Instead of following a complex method, we use KeyBERT (Grootendorst, 2020), which is more simple and minimalistic as it uses BERT embeddings and cosine similarity. Moreover, we are able to set the hyperparameters for keyphrase extraction such as keyphrase n-gram length, and the number of keyphrases, which helps in modifying the data consistently.

3.2 Input Prompting

Prompting refers to the addition of instructions in the model input, to generate conditional outputs. Jiang et al. (2020) follow a mining based method which follows a relation extraction mechanism followed by a paraphrasing method, which generates identical yet diverse prompts compared to the original prompt. Manually designed rules or a complex selection of input prompts from a discrete space as proposed by Shin et al. (2020) and Gao et al. (2021) can be used. However, selecting prompts from a discrete space would require more training and optimization. For our scenario, where the selection of prompts is done by a separate pipeline, we proceed with the keyphrases extracted from KeyBERT. Our method is similar to Puri et al. (2020), where instead of a question and passage tokens, we have article tokens and event tokens.

3.3 Sentence Embeddings

Word embeddings such as Glove (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013) provide a vector space for representing words. Word2Vec works on the principle of a context window, where it takes into account a fixed set of previous and next words for modelling its embedding. Glove leverages local and global information for generating word embeddings. Contextualized Embeddings, where a word's embedding depends on the context it is used in, were proposed in BERT (Devlin et al., 2019), which follow a masked language modelling and next sentence prediction tasks, where masked language modelling is a word level task and next sentence prediction is a sentence level task. Sentence-BERT (Reimers and Gurevych, 2019) leverages these contextualized embeddings along with a pooling layer to get a single embedding for the sentence, and a triplet loss function for generating sentence embeddings. However, this method requires having labelled data for positive and negative sentences. Chen et al. (2018) use a generalized pooling method, using a vector based weight multiplication, instead of a simple operation like max, or average. This method is trainable in an end to end fashion, without any additional data requirements. We perform a similar pooling operation on our event embeddings to get event representations which will be used by the decoder.

4 Methodology

4.1 Event Extraction

We use input prompting to guide the summarization task performed by BART (Lewis et al., 2020). This auxiliary input has an event sequence $\mathbf{E} = \mathbf{e_1}, ..., \mathbf{e_b}$ where $\mathbf{b}$ is the number of events and $\mathbf{e_i}$ is an event. Each event consists of $\mathbf{k}$ different tokens. Thus, an event $\mathbf{e_i} = \mathbf{e_{i1}}, ..., \mathbf{e_{ik}}$ where $\mathbf{e_{ij}} \,\varepsilon\, \mathbb{R}^{d_h}$, where d_h is the size of hidden representation. For event extraction, we use KeyBERT (Grootendorst, 2020) to extract keyphrases. KeyBERT uses BERT embeddings to create keywords and keyphrases that have maximum similarity to the document. This similarity is calculated using cosine similarity.

A word overlap threshold $\mathbf{t}$ is set to factor in diversity. Each key phrase is tokenized and padded to reach a fixed-length $\mathbf{k}$. The events are concatenated and inserted before the source text. Thus, input to the BART encoder is a sequence of events followed by the source text.

4.2 Input prompting

To make use of the events extracted from the article, we need to prompt the event data. The tokenized events are added before the tokenized source text. Thus the input to the encoder is $\mathbf{E_{1:b}X_{1:n}}$, where $\mathbf{n}$ is the length of the source text and $\mathbf{b}$ is the number of events. Individual events and event information and source text are separated by separator tokens.

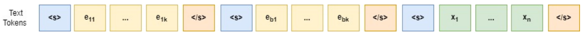

Figure 1: Event information prompted by adding event tokens prior to article tokens.

Event attention masks and source text attention masks are generated to distinguish event information from the source text. This modification is made to the input of the BART architecture to incorporate event information. This kind of concatenation helps us generate event embeddings in a similar manner to source text embeddings.

4.3 Summarization model

The summarization model follows a sequence-to-sequence transformer architecture consisting of an encoder and a decoder. We use BART-base as our base model for the architecture. BART-base consists of $\mathbf{N}$ = 6 layers of encoder and decoder, with the encoder consisting of multihead self-attention block, and the decoder consisting of multihead masked self-attention and cross-attention mechanism. Pooling is performed on the event embeddings to generate event representations. These event embeddings are sent to the decoder. In addition to these attention blocks, we propose the use of an additional attention block, called event attention, which applies cross attention between ground truth summary and the events extracted from our source text. The purpose of this block is to understand how much importance the events hold with respect to the ground truth summary.

4.3.1 Encoder

The encoder for BART-base consists of $\mathbf{N}$ = 6 layers, each with a self attention mechanism, feed forward layers and residual connections between the layers. The encoder's input is $\mathbf{E_{1:b}X_{1:n}}$, source text attention mask, and event attention mask, where $\mathbf{n}$ is the length of the source text and $\mathbf{b}$ is the number of events. The positional embeddings are added to these tokens and fed into the encoder. Output of each encoder layer is fed into the next encoder layer, for all layers from $\mathbf{L}$ = 1 to $\mathbf{L}$ = $\mathbf{N}$ - 1. Each layer produces embeddings of dimension $\mathbb{R}^{d_h}$. After the layer $\mathbf{L}$ = $\mathbf{N}$, the output of the $\mathbf{N}^{th}$ layer is separated to get event and article embeddings. Generalized pooling is performed on the event embeddings to get a representative embedding for each of the events.

The attention mechanism in the encoder consists of multiple heads. The attention block takes three inputs: queries $\mathbf{Q}\,\varepsilon\,\mathbb{R}^{d_q}$, keys $\mathbf{K}\,\varepsilon\,\mathbb{R}^{d_k}$ and values $\mathbf{V}\,\varepsilon\,\mathbb{R}^{d_v}$ where $\mathbf{d_q}$, $\mathbf{d_k}$, and $\mathbf{d_v}$ are the dimensions of queries, keys and values respectively. A similarity (dot product) between the keys $\mathbf{K}$ and queries $\mathbf{Q}$ is computed followed by the softmax function. Multiplying these scores with values $\mathbf{V}$ gives us the output for the attention block.

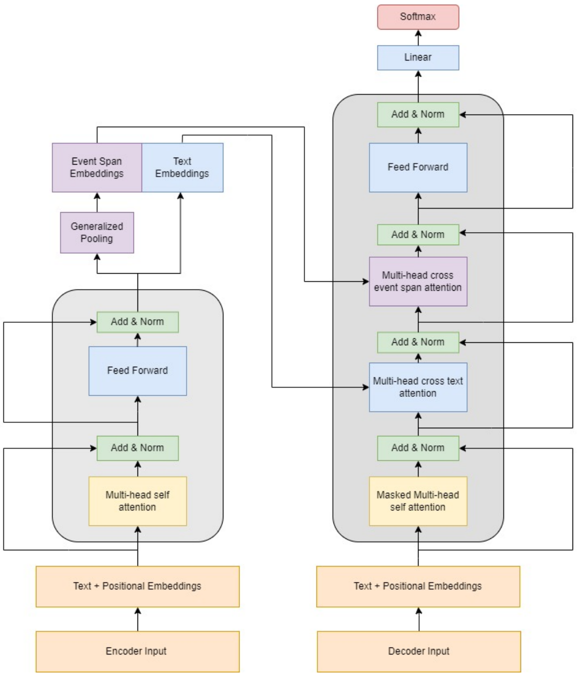
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-head cross event span attention
Add & Norm
Multi-head cross text attention
Add & Norm
Masked Multi-head self attention
Text + Positional Embeddings
Decoder Input
Event Span Embeddings
Text Embeddings
Generalized Pooling
Add & Norm
Feed Forward
Add & Norm
Multi-head self attention
Text + Positional Embeddings
Encoder Input

Figure 2: Model architecture consisting of a transformer encoder and decoder. Generalized pooling is performed on event information. Pooled event embeddings and encoder output is sent to the decoder. Decoder consists of an additional attention block, where attention is computed between events and ground truth summaries.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

The attention block consists of multiple heads, with each head focusing on a different representation. Outputs of each of these heads are concatenated and multiplied by a weight matrix to get the output for the attention block.

The queries, keys and values come from the encoder input for $\mathbf{L}$ = 1. For layers, $\mathbf{L}$ = 2 to $\mathbf{N}$, queries, keys and values come from the previous encoder layers. Encoder self-attention is not masked hence, can attend to all the positions of the input.

4.3.2 Generalized Pooling

To get a representation for each event, we employ a method to go from token embeddings to an embedding for each event. We split the output of the final encoder layer into event embeddings and source text embeddings. We use a weighted pooling mechanism to form a single vector representation of each event. However, we use a one-dimensional convolutional layer, instead of a feed forward layer (Chen et al., 2018). The idea behind using a convolutional layer is to get a sliding window so that each event can be handled without recursively passing it through the feed-forward layer. For an input $\mathbf{X_{1:n}}$ having events $\mathbf{E} = \mathbf{e_1}, \ldots, \mathbf{e_b}$ where $\mathbf{e_i} = \mathbf{e_{i1}}, ..., \mathbf{e_{ik}}$, a single representation is calculated for every event. $\mathbf{PE}$ represents the weighted pooled embeddings. Thus, the input to the decoder is $\mathbf{PE_{1:b}X_{1:n}}$ where $\mathbf{PE} = \mathbf{pe_1}, ..., \mathbf{pe_b}$.

$$pe_i = Conv1D(e_i) \quad (2)$$

4.3.3 Decoder

The decoder for BART-base consists of $\mathbf{N}$ = 6 layers, each with a masked self-attention mechanism, masked cross attention and a masked event attention block. The input to the decoder is $\mathbf{Y_{1:p}}$ as ground truth summary and $\mathbf{PE_{1:b}X_{1:n}}$ as encoder output, where $\mathbf{p}$ is the length of the ground truth summary. The output of each decoder layer is fed into the next decoder layer, for all layers from $\mathbf{L}$ = 1 to $\mathbf{L} = \mathbf{N} - \mathbf{1}$. Each layer produces embeddings of dimension $\mathbf{d_h}$. After the la yer $\mathbf{L}$ = $\mathbf{N}$, the output is passed through a feed-forward layer and softmax activation over the vocabulary to get the logits which are used for generating the summary.

The attention mechanism in the decoder is masked so that the decoder does not attend to inputs of time steps ahead of the current time step. These attention blocks are similar to the attention block in the encoder. The self-attention block uses ground truth summary as query, key, and value. For the cross attention mechanism, $\mathbf{X_{1:n}}$ is used for keys and values, whereas queries are obtained from the previous decoder layer. Similarly, for the event attention block, the queries are obtained from the previous decoder layer and the keys and values are obtained from the pooled event embeddings $\mathbf{PE_{1:b}}$.

5 Experimental Setup

5.1 Dataset

We have used CNN / Daily Mail (Nallapati et al., 2016) and XSUM (Narayan et al., 2018) datasets for our experimentation. CNN / Daily Mail consists of news articles and their abstractive summaries. XSUM consists of BBC articles which cover a wide variety of domains. Since we are using event-based summarization, we need to first extract the events and add them to the model input along with the article. In our experiments, we have extracted 4 events, each of these being keyphrases consisting of 4 words. We use BART tokenizer to tokenize the dataset. This may result in some words being split into multiple tokens. Thus every event is allocated 10 tokens including the start and end tokens. Since the input size for the encoder is 512, the articles are truncated to 472 tokens to accommodate the 40 tokens for the event sequence. Allocating more tokens to an event or increasing the number of events would decrease the number of tokens that can be taken from the source article. This results in event information vs article length trade-off. For CNN / Daily Mail the ground truth summaries are truncated to 128 tokens, and for XSUM the ground truth summaries are truncated to 64 tokens. This difference between ground truth summary lengths is because, in CNN / Daily Mail, the summaries are highlights from the news article, but in XSUM the summaries are mostly a sentence long. The different splits for the above-mentioned datasets are specified in Table 2. The number of events extracted from some of the articles is less than 4. In such cases, padding is added to reach the 40 tokens allocated for event information.

	CNN / Daily Mail	**XSUM**
Train	287113	204045
Validation	13368	11332
Test	11490	11334

Table 2: Number of data points in training, validation and testing sets for each of the datasets.

5.2 Implementation Details

We chose BART-base as our base model, instead of BART-large, due to insufficient resources for training the larger variant of the model. The encoder and decoder each consist of $\mathbf{N}$ = 6 layers. As BART is pre-trained, these pre-trained weights are taken to be the initial model weights. The weights for generalized pooling and event attention blocks are randomly initialized. BART-base consists of 12 attention heads for encoder and decoder, with hidden dimension size $\mathbf{d_h}$ = 768, and a vocabulary size $\mathbf{V}$ = 50,265. For the generalized pooling block, kernel size $\mathbf{k}$ = 10, a stride of 10, and the number

Model	R1	R2	RL	RLSum	BertScore
Event prompted BART-base	41.57	19.89	29.52	**38.88**	63.79
BART Large-CNN	44.16	21.28	40.90	36.42	**64.14**
Pegasus Large-CNN	**44.17**	**21.47**	**41.11**	36.39	62.52

Table 3: Metric values for CNN / Daily Mail Dataset

Model	R1	R2	RL	RLSum	BertScore
Event prompted BART-base	40.51	18.64	33.27	33.26	66.51
BART Large CNN	45.14	22.27	37.25	36.47	68.64
Pegasus Large CNN	**47.21**	**24.56**	**39.25**	38.63	**69.99**

Table 4: Metric values for XSUM Dataset

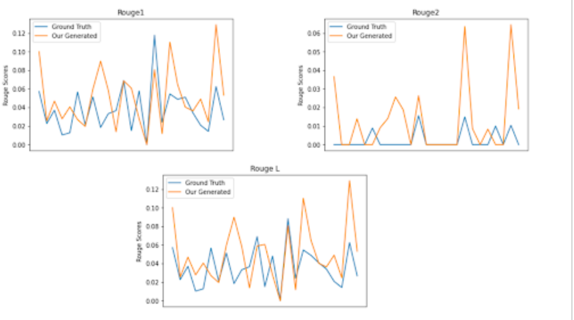

Figure 3: Rouge Scores between events vs ground truth, and events vs summaries generated by Event prompted BART-base for CNN / Daily Mail.

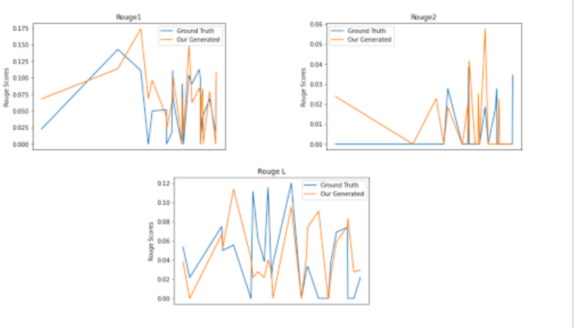

Figure 4: Rouge Scores between events vs ground truth, and events vs summaries generated by Event prompted BART-base for XSUM.

of input and output channels as 768 are used.

We use a cross-entropy loss objective for training the model. Our model has a total of 159M parameters, 139M parameters due to BART-base with an additional 20M parameters due to generalized pooling block and event attention block. We use NVIDIA Quadro RTX 6000 16GB GPU for running experiments on the model. We use a batch size of 8 for training and validation. The learning rate is set to 1e-05 with a linear learning rate scheduler for CNN / Daily Mail and a polynomial learning rate scheduler for XSUM. Weight decay is set to 5e-04 for CNN / Daily Mail and 1e-04 for XSUM and the models are trained to 500k steps for CNN / Daily Mail and 250k steps for XSUM. While decoding, we use beam search with a beam size of 5 for both datasets.

6 Results and Analysis

6.1 Metric Evaluation

The results of our model are quantified using Rouge1, Rouge2, RougeL, RougeLSum and BERTscore (Zhang* et al., 2020). Rouge1 and Rouge2 measure the uni-gram and bi-gram matches respectively. RougeL measures the longest common subsequence. RougeLSum is a variation of RougeL and it differs from RougeL in the treatment of the newline character. BERTscore computes the semantic similarity between generated and ground truth summary. We observe in Table 3 and Table 4 that rouge scores from our model are comparable with BART-Large and Pegasus-Large.

Since our summaries revolve around events, we compute the rouge scores between the identified events vs ground truth and identified events vs generated summaries. This will showcase the overlap between the events and their respective generated summaries. We randomly select 25 summaries from each of the datasets and compute rouge scores between each of the events vs the ground truth, and each of the events vs the summaries generated by our model. We average out the scores returned between a summary and each of the identified events to get a single value. We observe in Figure 3 and Figure 4, that the rouge scores between events and summaries generated by our model is close to or higher than the rouge scores between events and ground truth summaries.

CNN / Daily Mail and XSUM are both datasets used for abstractive summarization, however, the expectations from the generated summaries are different in both cases. CNN / Daily Mail consists of longer ground truth summaries which explain the article in a few sentences. On the other hand, XSUM consists of significantly shorter ground truth summaries. As observed in Figure 5 and Figure 6, the length of summaries generated by our model is similar to the length of ground truth summaries. While calculating the length of text, we tokenize the text using BART tokenizer and consider the number of tokens output by the BART tokenizer as the length of the text. In Figure 7 and Figure 8 the lengths of the generated summaries are divided into groups of 10 and the average rouge scores for all the summaries in a group is calculated.

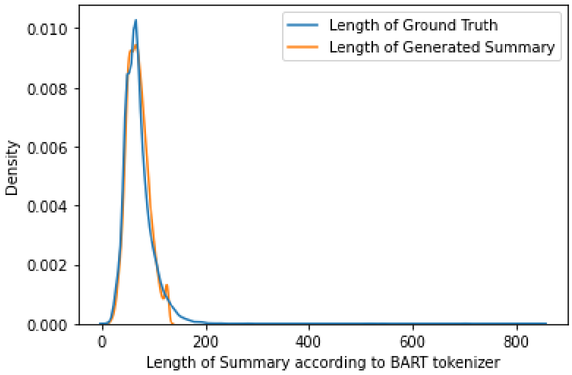

Figure 5: Distribution of summary lengths as calculated by the number of tokens generated by the BART tokenizer for the CNN / Daily Mail dataset.

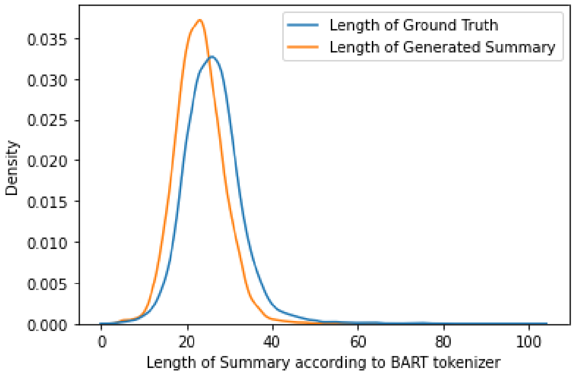

Figure 6: Distribution of summary lengths as calculated by the number of tokens generated by the BART tokenizer for the XSUM dataset.

It can be observed in Figure 7 that Rouge scores increase as the length of the articles increases in the CNN / Daily Mail dataset. However, in Figure 8 we observe that for the XSUM dataset, longer summaries seem to have lower rouge scores.

6.2 Human Evaluation

To capture the subjectiveness and diversity of language generation tasks such as summarization, we conduct human evaluation. Since we have added event input prompting to our model, we aim to utilize human evaluation as a method to understand to what extent summaries generated by our model are influenced by the events. Three evaluators, fluent in the English language were sent summaries generated by BART-Large-CNN, PEGASUS-Large-CNN and Event prompted BART-base, along with their respective source text, extracted events and ground truth summaries. The names of the models that generated the summaries to be evaluated, were not shared with the evaluators. The evaluators were provided with 25 such data points from the test split and a list of metrics to grade the summaries on. The metrics were fluency, event inclusiveness, factual correctness, coherence, and informativeness. Fluency is used to verify if the text generated has the correct grammatical structure and rules. Since our model incorporates input prompting using events, we use event inclusiveness as a metric to capture how much the summaries are influenced by the identified events. Factual correctness is included to confirm if the facts in the summary are consistent with the facts in the source text and ground truth. For understanding to what degree the summary makes sense as a whole, coherence is added as a metric. Informativeness is used to verify if the most important points of the article are present in the summary.

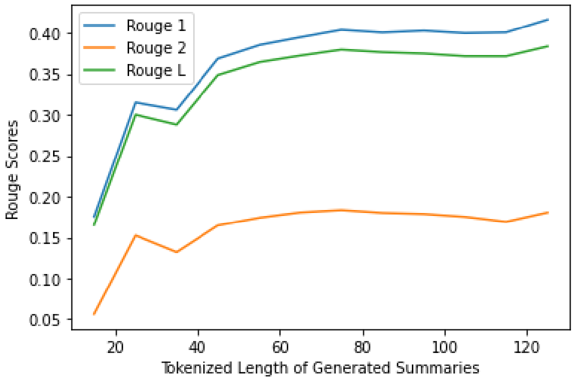

Figure 7: The average rouge scores for all the summaries in a group for CNN / Daily Mail.

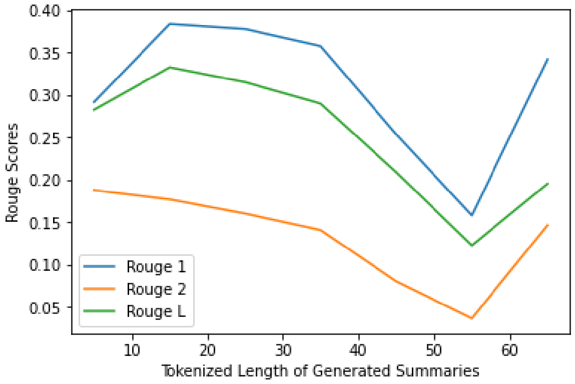

Figure 8: The average rouge scores for all the summaries in a group for XSUM.

The average of the scores was taken across all the data points for different models and their metrics. The average fluency, event inclusiveness, factual correctness, coherence, and informativeness scores for BART-Large-CNN, BART-base prompted by

Metrics		Event prompted BART-base	BART Large-CNN	Pegasus Large-CNN
Fluency	P1	4.12	**4.32**	3.76
	P2	4.88	**4.92**	4.72
	P3	4.64	**4.76**	4.40
Event inclusivness	P1	3.92	**4.16**	3.60
	P2	**3.84**	2.56	2.04
	P3	**4.16**	3.72	3.68
Factual Correctness	P1	**4.08**	4.04	3.80
	P2	**4.96**	4.88	4.56
	P3	**4.76**	4.64	4.44
Coherence	P1	**3.80**	3.60	3.28
	P2	**4.24**	3.84	3.88
	P3	**4.44**	4.40	3.64
Informativness	P1	3.52	**3.56**	3.16
	P2	**3.44**	**3.44**	2.68
	P3	3.80	**3.84**	3.20

Table 5: Human Evaluation results for the CNN / Daily Mail dataset.

events, and PEGASUS-Large CNN are shown in Table 5, where P1, P2, and P3 refer to the the three evaluators.

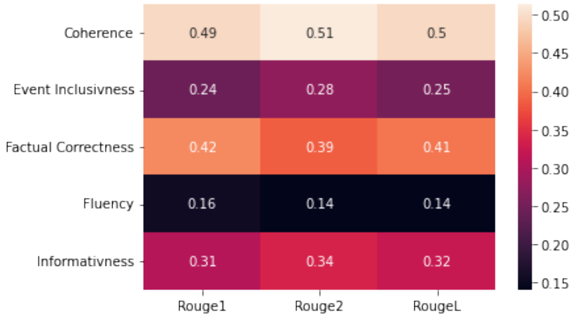

Figure 9: Correlation matrix between the rouge scores and human evaluation metrics. Scores assigned by reviewers are averaged out.

We can observe in that our scores for fluency are comparable to the scores for BART Large-CNN. We can see that 2 among 3 reviewers rated our generated summaries as being the most event inclusive. The first reviewer gave our summaries a score very close to the BART-Large score. For factual correctness and coherence the scores are highest for our generated summaries. For informativness, we observe that our scores are very similar to BART-Large scores, which rank the highest in the informativeness category.

7 Conclusion and Future Work

We introduce event prompting and an additional event attention block in the existing BART-base architecture to enable the model to generate summaries related to the events identified in the source text. Our model achieves comparable Rouge and BERT scores as the larger versions of BART and PEGASUS (Zhang et al., 2020). We also carry out human evaluation for our trained model, and achieve higher scores for event information inclusiveness as compared to the other transformer based models.

8 Acknowledgements

We thank Mr. Mayank Chowdhary, Mr. Jasdeep Singh Grover, Mr. Tim Fischer and the reviewers for their invaluable feedback.

References

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.

Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing sentence embedding with generalized pooling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1815–1826, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

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

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

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. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Faik Kerem Örs, Süveyda Yeniterzi, and Reyyan Yeniterzi. 2020. Event clustering within news articles. In *Proceedings of the Workshop on Automated Extraction of Socio-political Events from News 2020*, pages 63–68.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Alan Ritter, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1104–1112.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Si Sun, Zhenghao Liu, Chenyan Xiong, Zhiyuan Liu, and Jie Bao. 2021. Capturing global informativeness in open domain keyphrase extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 275–287. Springer.

Marco A Valenzuela-Escárcega, Gus Hahn-Powell, Mihai Surdeanu, and Thomas Hicks. 2015. A domain-independent rule-based framework for event extraction. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 127–132.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Runxin Xu, Tianyu Liu, Lei Li, and Baobao Chang. 2021. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In *Proceedings of the 59th Annual Meeting of the*

Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 3533–3546, Online. Association for Computational Linguistics.

Haoyu Zhang, Jingjing Cai, Jianjun Xu, and Ji Wang. 2019. Pretraining-based natural language generation for text summarization. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 789–797, Hong Kong, China. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.