APG: Automatic Prompt Generation for Improved Document Summarization

Jacob Parnell, Iñigo Jauregi Unanue, Scott Matthews, and Massimo Piccardi, Senior Member, IEEE *†‡§¶

August 6, 2025

Abstract

In recent years, prompting has become a mainstream technique for querying language models in natural language generation tasks such as machine translation, question answering and document summarization. In document summarization in particular, prompts have been used to control aspects of the predicted summary (e.g., style and length) and generally improve its quality. However, all the prompt-based summarization approaches proposed to date rely on significant manual design effort or annotation of additional training data. For this reason, in this paper we propose a novel approach for document summarization – nicknamed automatic prompt generation (APG) – allowing the model to learn to simultaneously generate its own prompts and the predicted summaries without the need for any supplementary annotations. This capability has been achieved by augmenting the training data with sets of keywords automatically extracted from the input documents with an off-the-shelf, unsupervised keyword extractor, and generating the prompts directly from the hidden states of the model's encoder. The proposed approach has been evaluated over five, diverse summarization datasets, showing that its performance has proved higher than that of many baseline models, including a state-of-the-art large language model, with increases of up to +9.50 ROUGE R_1 pp over BART-base, and +4.02 F_{BERT} score pp over GPT-4o mini, as well as evidence of improved generalization.

Summarization, document summarization, prompting, prompt generation, keyword extraction, language models, LLMs.

1 Introduction

Document summarization is a key field of natural language processing (NLP) that aims to convey the salient points of a given text document into a concise summary. At large, summarization is typically achieved in either of two ways: *extractive*, which selects and copies verbatim extracts from the document into the summary [1, 2], and *abstractive*, which is a more "human-like" approach where the summary is allowed to use its own wording and phrasing [3–5]. While in the early days the simpler, extractive approach was the only feasible, the continuous progress in deep learning, datasets and computational resources have made it possible for summarization to become increasingly abstractive and all

*This manuscript was first submitted for review on the 19th of May 2023.

†J. Parnell is with the University of Technology Sydney, Sydney, NSW 2007, Australia, and RoZetta Technology, Sydney, NSW 2000, Australia (e-mail: jacob.parnell@rozettatechnology.com).

‡I. Jauregi Unanue is with University of Technology Sydney, Sydney, NSW 2007, Australia, and RoZetta Technology, Sydney, NSW 2000, Australia (e-mail: inigo.jauregi@rozettatechnology.com).

§S. Matthews is with RoZetta Technology, Sydney, NSW 2000, Australia (e-mail: scott.matthews@rozettatechnology.com).

¶M. Piccardi is with University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: massimo.piccardi@uts.edu.au).

the more effective.

Among the recent, key developments in NLP technology, *prompting* occupies a special place as a core functionality of breakthrough generative models such as GPT [6], LaMDA [7] and PaLM [8]. In the jargon of generative models, the term "prompting" refers to the manual provision of a sequence of tokens to the model to trigger its response: for instance, a question to generate an answer, or a dialogue turn. In the context of abstractive summarization, providing a prompt to a summarization model can, in principle, be used to guide the summary toward certain themes, lexical choices, style, and even to focus on certain parts of the input document [9].

However, while prompting has a remarkable potential for abstractive summarization, it is in no way obvious how the prompts should be constructed and how they should fit within the overall model. This compounds with other, general challenges still affecting the performance of summarization models such as, for instance, the variable amounts of available training data in different domains. In particular, when the training data are limited, the models risk overfitting the provided reference summaries, and consequently impinging on the ability to generalize and faithfully summarize new documents [10]. For these reasons, the main goal of our paper is to explore the use of prompting to improve the performance of summarization models across different domains, without requiring extra annotated data beyond the standard human-annotated summaries provided as references.

The key idea of our approach is to learn to generate functional prompts for the summarization model jointly with the training of the model itself. The learning of the prompts is supported by the additional input of a set of keywords extracted from the input document by a fixed, unsupervised keyword extractor. The intuition behind the provision of these keywords is to "anchor" the generated summary more firmly to the content of the input document, preventing overfitting the reference summaries during training and encouraging generalization. However, rather than utilizing the keywords directly in the generation stage, we allow the model to learn functional prompts from them, concurrently with the generation of the summary. Notably, the keyword extractor does not require training or adaptation to the varying domains, resulting in a versatile, overall approach that can be seamlessly employed for summarization in any domain without the need for additional training data. Overall, the key contributions of our paper include:

- An original model for document summarization that leverages automatically-generated prompts to improve the predicted summaries.
- A novel training approach that uses a set of keywords extracted from the original document to train the model to simultaneously generate the prompts and the summaries.
- Experimental results over five datasets from three different domains (news, headline generation and dialogue) showing that the proposed approach has been able to significantly improve performance with respect to several strong baselines, and provide improvements and better generalization in the vast majority of cases.

The rest of this paper is organized as follows: Section 2 reviews the related work, while Section 3 presents the proposed approach. In turn, Section 4 describes the experimental set-up, while Section 5 presents and discusses the results. Finally, Section 6 summarizes the conclusions from our work.

2 Related Work

In recent years, prompting has become a prominent focus for natural language processing research, with the augmentation of Transformer-based language models [11] with several, different styles of *prompts* for a variety of tasks [12]. In very general terms, a prompt can be any sequence of tokens that is used to induce a response from the model and is kept distinct from its conventional input. Example of uses of prompts include task selection (e.g., prompting a multi-task model to perform a specific task), questions to obtain an answer, sentences to be translated,

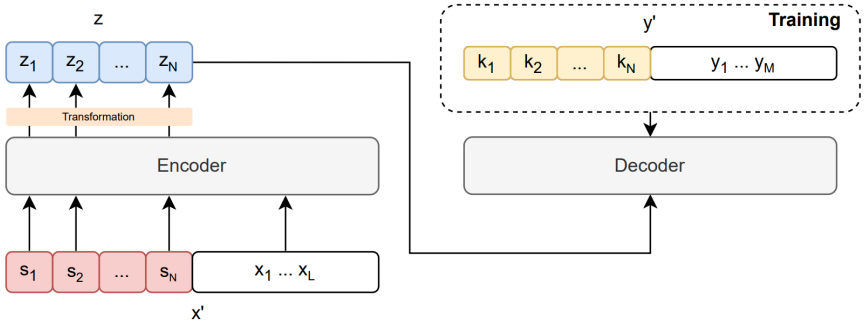


Figure 1: Overall view of the proposed APG model. NB: the model's encoder operates as both document encoder and prompt generator. The prefix vector, $s = (s_1, s_2, \ldots, s_N)$, is prepended to the input document, $x = (x_1, x_2, \ldots, x_L)$, to be encoded jointly, and the decoder's first N hidden states are used to generate the latent prompt, $z = (z_1, z_2, \ldots, z_N)$. The prompt is then passed to the decoder as its first N input tokens. During training, the ground-truth summary, $y = (y_1, y_2, \ldots y_M)$, is prepended with N pre-extracted keywords, $k = (k_1, k_2, \ldots, k_N)$, which encourage the model to learn to generate effective prompts. At inference time, 1) the encoder encodes the prepended input document and generates the latent prompt, 2) the first N tokens output by the decoder are discarded, and 3) those from position $N + 1$ to the end of the prediction are retained as the predicted summary.

topics for generation of essays, and so forth [13, 14]. Prompts have also been used as special control signals to modify or improve the performance of NLP models on individual predictions [15]. In particular, for document summarization, prompting has been utilized in several different manners which we briefly review hereafter [6, 16–21].

Prompt-based approaches differ along several dimensions. The first is the placement of the prompts within the Transformer's architecture: added to the encoder only [22], to the decoder only [9], or instead to both the encoder and the decoder [15, 17, 19]). A second dimension is whether the prompts are generated manually by the user (as in the vastly popular ChatGPT[1]) [12] or automatically by the model, typically after a learning stage (prompt learning) [23]. A last, main dimension is whether the prompts consist of ordinary, discrete tokens ("hard" prompts) [22] or, instead, word embeddings or other, similar numerical vectors ("soft" prompts) [15]. A common conversation around both soft prompts and learned hard prompts revolves around their interpretability [18, 24–26], with the general consensus being that such prompts cannot be interpreted meaningfully by humans. However, despite their lack of explicit interpretability, learned prompts have given ample evidence of their ability to control and improve the predictions made by language models.

In summarization, prompts have been used as an additional training signal to improve certain aspects of the generated summary [9, 22, 27, 28]. In particular, [9] has leveraged *entity chains* extracted from the input document by a separate entity extractor for guiding the decoder to generate summaries with larger numbers of relevant entities. In turn, [27] has proposed controlling other aspects of the generated summaries, including their length, style and the portion of the input documents to pay attention to. The length and style controls are obtained by prepending special tokens to the input document, while the portion control is obtained by inputting both the original document and the attention part separated by a special marker. To make the prompt generation more flexible, CTRLSum [22] has proposed utilizing an ad-hoc keyword extraction step

[1] https://openai.com/blog/chatgpt/

to obtain discrete keywords to prepend to the input document. In this way, the generation of keywords can be specialized for the particular task at hand. To strike a balance between the influence of the keywords and that of the input document on the generated summaries, CTRLSum uses "keyword dropout" that consists of randomly removing keywords during training to correspondingly reduce their influence. Eventually, LOTUS [28] has proposed learning latent prompts during training by minimizing their divergence with manually-annotated control signals. At inference time, the trained model is able to autogenerate suitable prompts for the given input documents.

More recently, large language models have become a staple in many NLP tasks, with users becoming adept at a field known as "prompt engineering". This allows the model to maximize the most out of a user's well-defined prompt to solve, as effectively as possible, the task at hand [20, 21].

However, all these approaches require considerable extra annotation effort compared to conventional summarization, either for supplying the model's training with "gold" prompts, or to separately train dedicated entity and keyword extractors. By contrast, the rationale for the proposed approach is to provide a more versatile model that can feed the keywords from an off-the-shelf, statistical-based keyword extractor that requires neither training nor supervision. In addition, the keywords are only needed at training time while the model learns to generate its own prompts. At inference time, our summarization model operates without the need for any external module (trained or untrained) or manual prompts, and is therefore completely equivalent to, and interchangeable with, a conventional summarization model.

3 The Proposed Approach: Automatic Prompt Generation

Let $x = (x_1, x_2, \ldots, x_L)$ represent the sequence of tokens of an input document, and $y = (y_1, y_2, \ldots, y_M)$ be the sequence of tokens of its ground-truth summary, typically with $M \ll L$. The summarization model is a conventional Transformer comprising of standard encoder and decoder modules [11]. Let us now introduce a *prefix vector* of chosen length N, $s = (s_1, s_2, \ldots, s_N)$, just consisting of a sequence of constant, notional tokens (in our implementation, we have set them all to token `<s>`). This prefix vector is prepended to the input document:

$$x' = s \oplus x \tag{1}$$

and jointly processed by the model's encoder to obtain a vector, h, of N corresponding hidden states. The encoder is then augmented with a "head" over the given vocabulary to generate N tokens, $z = (z_1, z_2, \ldots, z_N)$, one per hidden state. The generation can be described as follows: let us assume D to be the size of a hidden state and V that of the vocabulary, and refer to the model's token embedding matrix of size $V \times D$ as E. We then form the logit matrix, Eh, of size $V \times K$ and choose the K tokens that maximize its columns:

$$z = \underset{(1\ldots V)^K}{\operatorname{argmax}}(Eh) \tag{2}$$

The generated tokens are used to *prompt* the model's decoder, i.e to form the sequence of its first N input tokens. Given their placement at the beginning of the decoder's input, such prompt tokens are able to influence the generation of all the decoder's output tokens, including those that will form the actual, predicted summary. The question is then: how can the encoder learn to generate a useful vector of prompt tokens? To this aim, we propose leveraging a precomputed vector of N keywords, $k = (k_1, k_2, \ldots, k_N)$, extracted from the input document by an external, unsupervised keyword extractor. During training, the k keywords are prepended to the ground-truth summary, y:

$$y' = k \oplus y \tag{3}$$

to become the target of the model's training objective:

$$\underset{\theta}{\operatorname{argmax}} \Big(\sum_{t}^{N} \log p_\theta(k_t|z_{<t}, x') + \sum_{t}^{M} \log p_\theta(y_t|y_{<t}, z, x') \Big) \tag{4}$$

In this way, the model is able to learn to simultaneously generate the keywords (first N tokens) and the actual summary (from the $N+1$-th token to the $N+M$-th token) based on the prepended input document and the prompt. In turn, this training objective influences the encoder by backpropagation and its updated parameters will lead to the generation of different prompts. Empirically, we have observed that this training objective converges stably irrespective of both the initialization and the training data. At inference time, the first N tokens output by the decoder are simply discarded, and the subsequent tokens retained as the predicted summary. Fig. 1 shows an overall view of our model.

3.1 Automatic Keyword Extraction

To obtain the target keywords for each input document, we utilize an unsupervised, lightweight keyword extractor, YAKE!, which is able to select the most relevant keywords of a document based on the word statistics of the document itself [29]. The extraction is performed with a five-step approach including: preprocessing and candidate term identification, feature extraction, term scoring, n-gram generation and candidate keyword scoring, and, finally, de-duplication and ranking. Unlike other statistical methods (e.g., TF-IDF), YAKE! does not require a reference document corpus and claims to be able to support texts of different sizes, domains or languages. We refer the reader to [29] for more details.

Other works have used various keyword or entity extractors to control summarization [22, 28, 30, 31]. However, all those approaches expect the extractors to be trained with specific training data. Conversely, YAKE! is an unsupervised and untrained approach which can be used on any document. This dispenses with the need to train the extractor for any new domain or dataset, and also removes a potential source of overfitting. More so, it removes the need for any extra manual annotation (keywords, entities, gold signals), making the proposed approach applicable off-the-shelf to any existing summarization dataset.

In summary, during the training stage our model aims to learn to generate the prompt based on 1) the input document and 2) the training signal provided by the reference summary and a set of keywords extracted by an unsupervised keyword extractor. At inference time, the model is able to generate both the prompt and the summary automatically. Given the pivotal role played by the prompt, we refer to the proposed approach as *automatic prompt generation (APG)* for document summarization.

4 Experiments

4.1 Datasets

We have carried out a set of experiments over five datasets across three different domains: two news summarization datasets, CNN/DailyMail [4] and XSum [32]; two dialogue summarization datasets, SamSum [33] and DialogSum [34]; and a headline generation dataset, AESLC [35]. The main statistics of these datasets are provided in Table 14 in Appendix .8, including the number of samples and the average token lengths of the input documents and the reference summaries. We have reported these statistics directly from the original papers where available, and otherwise computed them ourselves using the NLTK tokenizer. Based on these values, we have adjusted the maximum output length for the predicted summaries to correspond approximately with the average length of the reference summaries. In addition, since the majority of the datasets have an average input document length below 512 tokens, we have limited the input length for all datasets to 512 tokens. Lastly, since keyword extraction can be performed prior to summarization, we have pre-extracted and tokenized all the keywords once and for all using the keyword extractor described in Section 3.1. The datasets are identical to those that are publicly available, and our experiments can be easily reproduced using the scripts provided in our GitHub repository[2].

[2]We release our code to permit complete reproducibility of our experiments at: `https://github.com/`

4.2 Models and Training

To limit the computational load and energy consumption of our experiments, we have selected a Transformer of limited size, BART-base [36], as our baseline. We have trained this baseline with a conventional negative log-likelihood objective, stopping training when a standard convergence criterion based on the validation performance was met, or the maximum number of training epochs was reached. We have built our model over this baseline and compared its performance with it. In addition, we have compared the performance with an off-the-shelf, state-of-the-art summarizer based on T5-base [37] which is a comparable, yet slightly larger, pre-trained language model often used in summarization tasks. We have also included in the performance comparison a replicated version of CTRLSum [22] which can be regarded as a state-of-the-art representative of the category of the prompt-based summarizers. For its training, we have used the configuration described in the original paper with the exception of a) the number of keywords extracted in the preprocessing step, b) the training time, and c) the underlying Transformer model. To strike an approximate parity and make the results more immediately comparable, we have set the maximum number of extracted keywords to 10 as in our APG model, set the same training time, and used BART-base as the underlying Transformer. As the last fine-tuned summarizer, we have included *prompt tuning* [18], re-implemented using the same BART-base model and number of prompt tokens as our APG approach. The original paper [18] suggested re-using the pre-trained model's parameters and fine-tuning the prompt embeddings alone. However, for a fairer comparisons with the other models, we have allowed updating all of the model's parameters. Our final baseline is a contemporary, 8B-parameter large language model, GPT-4o mini[3], which offers marked performance improvements compared to its predecessor GPT-3.5, at a significantly cheaper price point[4]. We have used this LLM with four manual prompting variations to explore its performance potential; in specific, our prompting methods include:

- **Direct**: directly asking the LLM to summarize the input document.
- **YAKE!**: an approach which adds the same YAKE! keywords used by APG as part of the prompt to help inform the summary generation.
- **Self-Extract**: an approach which expects the LLM to generate the keywords before generating the summary.
- **In-Context**: in-context learning with a document-summary pair from the training set in the prompt to provide context, before the request to generate the summary.

All details are provided in Appendix .1.

To evaluate our results, we report the F1 variants of the ROUGE score [38] (noted as R_1, R_2, and R_L hereafter) and the F1 variant of BERTScore [39] (F_{BERT}) averaged over three independently-initialized training runs. The ROUGE scores have been used almost universally in literature to evaluate the performance of dialogue summarization. However, they are not able to properly score desirable linguistic features of the summaries such as, for instance, the use of synonymy and paraphrasing, and capture shortcomings such as factual inconsistency [24]. For these reasons, in the evaluation we have included BERTScore, which is based on word embeddings and is able to better assess the semantic content of the summaries.

When we use these metrics to score the predicted summaries, we exclude the first N generated tokens that correspond to the prompt slots, and use the remaining for scoring. In addition, all the performance differences between the proposed approach and the compared models have been tested for statistical significance using a non-parametric bootstrap test [40]. All other hyperparameters used for the training configurations are presented in Appendix .1.

jacob-parnell-rozetta/apg

[3] https://openai.com/index/GPT-4omini-advancing-cost-efficient-intelligence/

[4] https://openai.com/api/pricing/

5 Results

Table 1 show the main results grouped by domain: news summarization comprising of CNN/DailyMail and XSum, dialogue summarization comprising of DialogSum and SamSum, and headline-generation comprising of AESLC, in order. The notations used in the tables are as follows: $(*)$ refers to our replicated CTRLSum model that uses BART-base for comparability, and therefore not values reported from the original paper; $(**)$ denotes the T5 model. This model always deteriorated in validation performance after the first epoch, hence we report its best results prior to deterioration; (†) refers to statistically significant differences with respect to the BART-base baseline with a p-value < 0.01 [40]; and (‡) with a p-value < 0.05. The best scores of each configuration are highlighted in boldface. For brevity, in this section, we only report the In-Context and YAKE! variants of our GPT-4o mini baselines as these are the most relevant to our approach, while the results for the other two prompting methods are presented in Table 13 in Appendix .6.

News summarization. The proposed approach has been able to achieve the best ROUGE and F_{BERT} scores for CNN/DailyMail, and the best F_{BERT} score for XSum. In contrast, the replicated CTRLSum model has achieved the best ROUGE R_1 and R_L scores on XSum, while prompt tuning has achieved very strong ROUGE scores across both datasets. However, the proposed approach has clearly outperformed all the others in terms of F_{BERT} score. Such a disagreement between ROUGE and F_{BERT} scores is remarkable, as it seems to suggest that the proposed approach has been capable of producing the most semantically-relevant summaries. In turn, the results for GPT-4o mini have been interesting (with a clear edge for the In-Context prompting), yet markedly lower than the fine-tuned baselines and the proposed approach.

Dialogue summarization and headline-generation. In dialogue summarization, the proposed approach has achieved the best scores with the SamSum dataset, while the T5 model has achieved the best scores for DialogSum (except ROUGE R_1). However, the performance improvements obtained by the proposed approach for SamSum have been much more substantial, with an increase of +5.30 pp in ROUGE R_1 over the BART baseline, and marked increases also in the other scores. In turn, prompt tuning has been able to achieve the second-best scores with both SamSum and AESLC. Yet, its results have been below those of the proposed APG approach for all datasets and metrics. In the case of GPT-4o mini with In-Context prompting, the results have been roughly comparable with those of the CTRLSum baseline (even surpassing it in some metrics), yet still lower than those of all the other fine-tuned approaches. The proposed approach has been able to achieve by far the best scores also for AESLC, with an improvement of +9.50 pp in ROUGE R_1 over the BART baseline and more than +10 pp in ROUGE R_1 over all prompting variations of GPT-4o mini.

Overall, the proposed approach has obtained the best scores in 14 cases out of 20. While the BART baseline, the T5 model and the prompt tuning approach have performed competitively on the whole, the replicated CTRLSum model has reported disappointing results over the DialogSum, SamSum, and AESLC datasets, suggesting that more adaptations would be required to make it perform well also in these domains. For fairness, we acknowledge that the original CTRLSum paper used a larger underlying model (BART-large, with 406M parameters vs the 139M of BART-base), and was able to obtain substantially higher accuracies compared to its BART-large model baseline when given enough training time. However, its modest performance with BART-base in all our experiments is somehow surprising. Its scores not only have been worse than those of the proposed model for four datasets out of five, but also of those of the BART baseline. However, it is possible that CTRLSum may require much longer training times to reach an effective parametrization, and to this aim we report further experiments in Appendix .2. Overall, it seems that the proposed APG model presents a smaller optimization problem and is able to train effectively in approximately the same number of training iterations as its BART baseline. For prompt tuning [18], we have also repeated the

experiments with a different initialization technique proposed in the original paper, where the embeddings of the prefix tokens are initialized with the embeddings of actual tokens sampled from the top-5000 tokens in the vocabulary. However, the model performed generally worse, as shown by Table 11 in the Appendix. In addition, we have carried out an experiment over the SamSum dataset increasing the prompt length to 50 and 100 tokens. The results in Table 10 show that this has not been able to improve performance. Lastly, we certainly cannot exclude that the performance of GPT-4o mini could be improved with a range of other techniques such as larger numbers of examples, chain-of-thought guidance, or retrieval-augmented generation. However, the difference in its parameter size with the proposed APG model is drastic (139M parameters vs. estimated 8B parameters; more than 57x), highlighting its remarkable performance.

5.1 Qualitative Analysis

To further explore the performance of the proposed approach, Table 2 shows a qualitative example for each of the datasets from BART, GPT-4o mini and APG. Correct predictions, incorrect predictions/incomplete sentences, and entities retrieved from the input document have been highlighted in green, red, and blue, respectively (the comments in the accompanying text and captions allow for color-blind comprehension). In addition, we report the F_{BERT} scores of the predictions and the prompts generated by APG. These examples show quite conspicuously that the APG approach has generally led to better summaries.

News datasets. In the case of the CNN/DailyMail dataset, the qualitative differences are slight, which would justify the modest score improvements of Table 1; however, a noticeable change is the inclusion of the token "Spaniard" in the summary predicted by APG, which is a key piece of information present in the input document, yet absent from the reference summary. In the case of the XSum dataset, the BART baseline has produced a relatively short summary that is factually incorrect with respect to the reference, while the summary generated by APG has been much more faithful (as highlighted by the entities in green/light font) and has retrieved an extra entity from the input document ("one-bedroom flats"), showing that the prompt has had a positive impact on summary generation. In turn, GPT-4o mini has produced a grammatically-poor summary, with several incorrect predictions and/or incomplete sentences (highlighted in red). Of all the compared approaches, prompt tuning appears to be the most comparable with APG, given the frequency of faithful terms relative to incorrect predictions.

Dialogue and headline-generation datasets. Similar comments can be addressed to the example from the DialogSum dataset, where the anonymized tokens generated by the BART, prompt tuning, and GPT-4o mini baselines ("#Person 1#") have instead been recovered as correct named entities by APG. In addition, APG has been able to include another informative entity ("Saturday") that is present in the input document, but missing from the reference, giving further evidence to the information transfer permitted by the prompting mechanism. In a similar manner, GPT-4o mini has improved information quality in this summary by including the statement regarding concerns around "work-life balance". Lastly, APG has included the invitation's acceptance mentioned in the reference. Also for the last two datasets, SamSum and AESLC, APG has been able to produce informative summaries as opposed to the wholly incorrect summaries of the BART baseline. In turn, prompt tuning has generated some unfaithful terms for SamSum, while being somewhat aligned to the APG-generated summary for AESLC. GPT-4o mini has also tended to over-generalize, particularly for AESLC, where it has failed to keep the generated summary to a length comparable to that of the reference summary.

For completeness, all the examples also show the prompt for the decoder that was automatically generated by the proposed APG model. Patently, none of the prompts carry any meaning for humans. This is in line with the recent literature which has reported the lack of interpretability of automatically generated prompts [26]. However, despite the lack of meaning for humans, it appears that the prompts, together with their hidden states and self-attention weights in the relevant slots of the decoder, have been able to

provide a useful amount of information toward the prediction of the actual summary [25]. This is testified by both the qualitative improvements in the summary's content and the noticeable increases in F_{BERT} score.

5.2 Human Evaluation

Following [41, 42], we have also conducted a small-scale human evaluation over 20 randomly-extracted samples from the CNN/DailyMail dataset, for both the BART baseline and our proposed APG approach. Three annotators have blindly assessed the generated summaries based on the two dimensions of fluency and conciseness, with a scoring over a 5 point Likert scale, from 1 (worst) to 5 (best). The results in Table 3, averaged across the three annotators for both evaluation dimensions, show that APG has been the preferred method, mildly outperforming BART by as much as +0.14 pp.

In addition, to evaluate the inter-annotator agreement, we have counted how many times each annotator preferred the prediction of the proposed model over the baseline's. For the conciseness dimension, all three annotators preferred the proposed model in 15 out of 20 cases. For the fluency dimension, two annotators preferred the proposed model 17 times out of 20, while the other preferred it 11 times out of 20. Overall, we believe that there is sufficient consensus to state that the proposed model outperformed the BART baseline in this evaluation.

5.3 Factuality

Another interesting performance dimension for a summarizer is *factuality*, which quantifies the ability of a summary to reflect the facts in the input document [43]. To evaluate the factuality of the proposed approach, in this section we carry out an analysis with two dedicated metrics. The first is FactCC [44], a news-based factuality metric that is able to assess the factuality of summaries in the news domain. The second is FactKB [45], a general-purpose factuality metric that leverages a language model pre-trained with a bespoke factuality-based objective. For these reasons, we have measured FactCC only over the CNN/DailyMail and XSum datasets, while we have measured FactKB for all of them; the results over the test sets are reported in Table 4. The results seem to confirm that the use of keywords in the proposed APG approach allows it to remain more factually grounded to the input documents.

5.4 Keyword Density Analysis

In this section, we attempt to explore possible reasons for the noticeable difference in performance across the different domains, and in particular why the improvement with the proposed approach has been much more pronounced for certain datasets. To this aim, we have set to measure the "density" of the keywords present in the predicted summaries compared to that of the ground-truth references. Previous work has shown that documents tend to display an "inverted pyramid structure", with a higher keyword density in the beginning portions of the document, and decreasing thereafter [46]. Our empirical observations with the keywords extracted by YAKE! have confirmed this trend for both the documents and their ground-truth summaries, and we expect that a good predicted summary should reflect it to a large extent.

The results are reported in Fig. 2, which plots the keyword density produced by the BART and APG models with the CNN/DailyMail and SamSum datasets, along with the keyword density of the ground-truth summaries, split along the successive parts of the summary (0-10%, 10-20% etc). As a preliminary observation, the keyword density for the ground-truth summaries in Fig. 2 shows the decreasing trend that we had expected. For the CNN/DailyMail dataset (Fig. 2a), the behaviors of the BART baseline and APG have both been very similar to that of the ground truth, and this might be an underlying justification for their limited performance differences in Table 1. Conversely, for the SamSum dataset (Fig. 2b) the BART baseline has not been able to include a large number of keywords in the critical first 10% of its summary, while APG has included an amount that is very comparable to that of the ground truth. This alignment between the ground truth and our APG model could be a possible explanation for

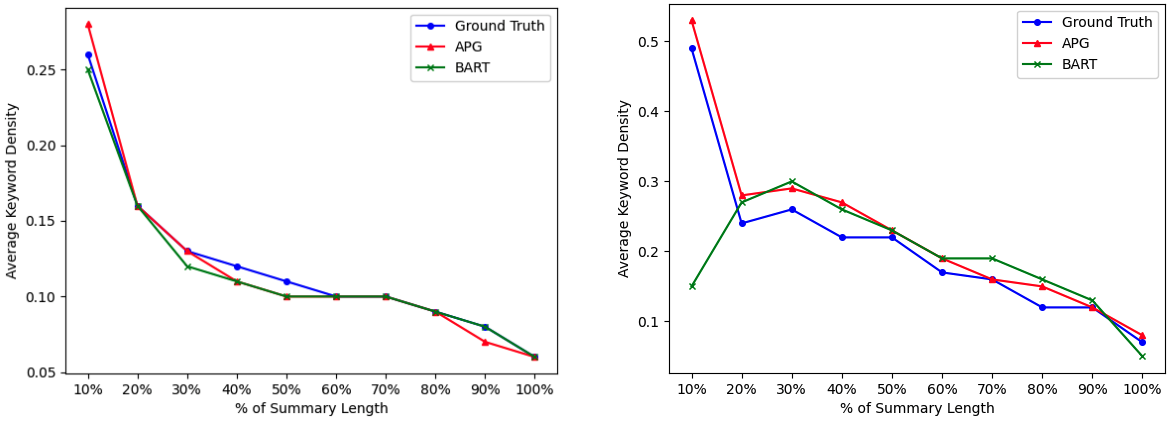


Figure 2: Keyword density for the ground-truth, BART and APG summaries along the various parts of the summary for the CNN/DailyMail (left) and SamSum (right) test sets. The summaries have been split into bins of 10% of their length, and the keyword density averaged across the entire test set.

its much larger performance gap over the BART baseline on this dataset. At the same time, it can also be read as an overall validation for our intuition of using keywords to guide the training of our model.

5.5 Comparison With Direct Keyword Prompting

The proposed approach uses the keywords extracted from the training documents to train the model to generate effective summarization prompts. A logical question could be whether the keywords themselves could perform as an effective prompt if directly used as input for the decoder; an approach that we refer to as "direct keyword prompting" hereafter. To probe this, we have carried out an experiment where the model's decoder has been prompted with the keywords extracted from the input document during both training and inference, and everything else has been kept identical to the BART baseline. Table 5 reports the results over the validation split of the CNN/DailyMail dataset, showing that the direct keyword prompting approach has performed poorly not only compared to the proposed approach, but also to the plain BART baseline. Results over other datasets and splits have shown a very comparable trend, and for this reason we have not pursued this approach further.

5.6 Cross-Dataset Analysis

In this analysis, we explore the ability of the proposed APG approach to generalize to other datasets from within the same domain (e.g., news to news, dialogue to dialogue). We regard this as a very desirable property of a summarization model which can give evidence to its learning of the specific semantic and syntactic properties of a given domain and robustness beyond a specific dataset. To this aim, Table 6 reports the average of the ROUGE scores (R_1, R_2, R_L) along with the F_{BERT} score for cross-dataset experiments with APG and the BART baseline. The results show that the proposed APG approach has been able to obtain a higher average ROUGE score in three out of four configurations and higher F_{BERT} scores in all tested combinations. In addition, while the scores are obviously lower than those from the within-dataset experiments reported in Tables 1 and 1, there is no evidence of performance collapse.

In particular, the DialogSum-trained APG model

has reported very high ROUGE and F_{BERT} scores when tested on the SamSum dataset. This may be partially explained by the fact that DialogSum is composed of real-life spoken dialogues [34], which are arguably complex and should generalize well to other dialogue datasets. However, the APG approach has generalized much better than the BART baseline (+4.53 pp ROUGE and +1.32 pp F_{BERT}), thanks to the extra information provided by the prompt. Conversely, the XSum-trained models are the only case where the ROUGE score of the BART baseline has been higher than that of APG. We attribute this to the abstractive nature of the XSum dataset coupled with the influence of the prompt, which may have increased the use of synonyms and paraphrases in the APG's predictions, and, as a consequence, slightly decreased their literal matches with the reference summaries. This attribution is confirmed by the fact that the F_{BERT} scores are comparable, showing that there has been no drop in the APG's semantic summarization capabilities also in this case.

6 Conclusion

This paper has proposed a novel, prompt-based approach for document summarization – automatic prompt generation, or APG — which is able to automatically generate a suitable prompt from the model's encoder and use it to influence the decoder's predicted summary. The prompt generation is learned concurrently with the summary prediction during the training stage by augmenting the reference summaries with a set of keywords extracted from the input document with an off-the-shelf, unsupervised keyword extractor. Experimental results over five, diverse summarization datasets in the news, dialogue and headline-generation domains have showed that the proposed APG approach has, in most cases, achieved higher scores than strong baselines (BART, T5, CTRLSum and GPT-4o mini), with increases of up to +9.50 ROUGE R_1 pp over BART-base and +4.02 F_{BERT} score pp over GPT-4o mini. A qualitative analysis, a small-scale human evaluation, and a factuality analysis have all confirmed that APG has been able to generate more informative and factual summaries than the baselines, and experiments carried out over cross-dataset configurations have given evidence of its improved within-domain generalization ability. Given that the performance differences across domains remain significant, as an area of possible, future research we aim to explore adaptations of the prompt generation mechanism that can explicitly target reductions of such a performance gap.

References

[1] R. Nallapati, F. Zhai, and B. Zhou, "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, p. 3075–3081, AAAI Press, 2017.

[2] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 1747–1759, Association for Computational Linguistics, June 2018.

[3] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 379–389, Association for Computational Linguistics, Sept. 2015.

[4] A. See, P. J. Liu, and C. D. Manning, "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)*, (Vancouver, Canada), pp. 1073–1083, Association for Computational Linguistics, July 2017.

[5] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization,"

in *International Conference on Learning Representations*, 2018.

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, vol. abs/2005.14165, 2020.

[7] R. Thoppilan, D. D. Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, Y. Li, H. Lee, H. S. Zheng, A. Ghafouri, M. Menegali, Y. Huang, M. Krikun, D. Lepikhin, J. Qin, D. Chen, Y. Xu, Z. Chen, A. Roberts, M. Bosma, V. Zhao, Y. Zhou, C.-C. Chang, I. Krivokon, W. Rusch, M. Pickett, P. Srinivasan, L. Man, K. Meier-Hellstern, M. R. Morris, T. Doshi, R. D. Santos, T. Duke, J. Soraker, B. Zevenbergen, V. Prabhakaran, M. Diaz, B. Hutchinson, K. Olson, A. Molina, E. Hoffman-John, J. Lee, L. Aroyo, R. Rajakumar, A. Butryna, M. Lamm, V. Kuzmina, J. Fenton, A. Cohen, R. Bernstein, R. Kurzweil, B. Aguera-Arcas, C. Cui, M. Croak, E. Chi, and Q. Le, "Lamda: Language models for dialog applications," 2022.

[8] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," 2022.

[9] S. Narayan, Y. Zhao, J. Maynez, G. Simões, V. Nikolaev, and R. McDonald, "Planning with learned entity prompts for abstractive summarization," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 1475–1492, 2021.

[10] J. Parnell, I. Jauregi Unanue, and M. Piccardi, "A multi-document coverage reward for RELAXed multi-document summarization," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Dublin, Ireland), pp. 5112–5128, Association for Computational Linguistics, May 2022.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, pp. 1–11, 2017.

[12] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *CoRR*, vol. abs/2107.13586, 2021.

[13] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, T. Bers, S. Biderman, L. Gao, T. Wolf, and A. M. Rush, "Multitask prompted training enables zero-shot task generalization," 2021.

[14] Y. He, H. S. Zheng, Y. Tay, J. Gupta, Y. Du, V. Aribandi, Z. Zhao, Y. Li, Z. Chen, D. Metzler,

H.-T. Cheng, and E. H. Chi, "Hyperprompt: Prompt-based task-conditioning of transformers," 2022.

[15] X. Liu, Y. Gao, Y. Bai, J. Li, Y. Hu, H. Huang, and B. Chen, "PSP: Pre-trained soft prompts for few-shot abstractive summarization," in *Proceedings of the 29th International Conference on Computational Linguistics*, (Gyeongju, Republic of Korea), pp. 6355–6368, International Committee on Computational Linguistics, Oct. 2022.

[16] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[17] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," 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)*, (Online), pp. 4582–4597, Association for Computational Linguistics, Aug. 2021.

[18] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, (Online and Punta Cana, Dominican Republic), pp. 3045–3059, Association for Computational Linguistics, Nov. 2021.

[19] X. Liu, K. Ji, Y. Fu, W. L. Tam, Z. Du, Z. Yang, and J. Tang, "P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks," 2021.

[20] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large language models are human-level prompt engineers," 2023.

[21] Y. Wang, Z. Zhang, and R. Wang, "Element-aware summarization with large language models: Expert-aligned evaluation and chain-of-thought method," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 8640–8665, Association for Computational Linguistics, July 2023.

[22] J. He, W. Kryscinski, B. McCann, N. F. Rajani, and C. Xiong, "Ctrlsum: Towards generic controllable text summarization," *CoRR*, vol. abs/2012.04281, 2020.

[23] K. Qi, H. Wan, J. Du, and H. Chen, "Enhancing cross-lingual natural language inference by prompt-learning from cross-lingual templates," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Dublin, Ireland), pp. 1910–1923, Association for Computational Linguistics, May 2022.

[24] M. Gao and X. Wan, "DialSummEval: Revisiting summarization evaluation for dialogues," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 5693–5709, Association for Computational Linguistics, July 2022.

[25] M. Deng, J. Wang, C.-P. Hsieh, Y. Wang, H. Guo, T. Shu, M. Song, E. P. Xing, and Z. Hu, "Rlprompt: Optimizing discrete text prompts with reinforcement learning," 2022.

[26] A. Webson and E. Pavlick, "Do prompt-based models really understand the meaning of their prompts?," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 2300–2344, Association for Computational Linguistics, July 2022.

[27] A. Fan, D. Grangier, and M. Auli, "Controllable abstractive summarization," in *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, (Melbourne, Australia), pp. 45–54, Association for Computational Linguistics, July 2018.

[28] Y. Zhang, X. Zhang, X. Wang, S.-q. Chen, and F. Wei, "Latent prompt tuning for text summarization," 2022.

[29] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "Yake! keyword extraction from single documents using multiple local features," *Information Sciences*, vol. 509, pp. 257–289, 2020.

[30] E. Çano and O. Bojar, "Keyphrase generation: A text summarization struggle," 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)*, (Minneapolis, Minnesota), pp. 666–672, Association for Computational Linguistics, June 2019.

[31] H. Li, J. Zhu, J. Zhang, C. Zong, and X. He, "Keywords-guided abstractive sentence summarization," in *AAAI Conference on Artificial Intelligence*, 2020.

[32] S. Narayan, S. B. Cohen, and M. Lapata, "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*, (Brussels, Belgium), pp. 1797–1807, Association for Computational Linguistics, Oct.-Nov. 2018.

[33] B. Gliwa, I. Mochol, M. Biesek, and A. Wawer, "SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization," in *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, (Hong Kong, China), pp. 70–79, Association for Computational Linguistics, Nov. 2019.

[34] Y. Chen, Y. Liu, L. Chen, and Y. Zhang, "DialogSum: A real-life scenario dialogue summarization dataset," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, (Online), pp. 5062–5074, Association for Computational Linguistics, Aug. 2021.

[35] R. Zhang and J. Tetreault, "This email could save your life: Introducing the task of email subject line generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 446–456, Association for Computational Linguistics, July 2019.

[36] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "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*, (Online), pp. 7871–7880, Association for Computational Linguistics, July 2020.

[37] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2020.

[38] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, (Barcelona, Spain), pp. 74–81, Association for Computational Linguistics, July 2004.

[39] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020.

[40] R. Dror, G. Baumer, S. Shlomov, and R. Reichart, "The hitchhiker's guide to testing statistical significance in natural language processing," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Melbourne, Australia), pp. 1383–1392, Association for Computational Linguistics, July 2018.

[41] J. Zhu, Q. Wang, Y. Wang, Y. Zhou, J. Zhang, S. Wang, and C. Zong, "NCLS: Neural cross-lingual summarization," 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), (Hong Kong, China), pp. 3054–3064, Association for Computational Linguistics, Nov. 2019.

[42] J. Wang, F. Meng, D. Zheng, Y. Liang, Z. Li, J. Qu, and J. Zhou, "Towards unifying multilingual and cross-lingual summarization," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 15127–15143, Association for Computational Linguistics, July 2023.

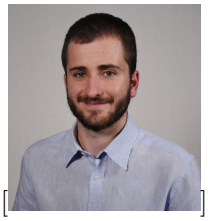
[43] A. Pagnoni, V. Balachandran, and Y. Tsvetkov, "Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Online), pp. 4812–4829, Association for Computational Linguistics, June 2021.

[44] W. Kryscinski, B. McCann, C. Xiong, and R. Socher, "Evaluating the factual consistency of abstractive text summarization," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 9332–9346, Association for Computational Linguistics, Nov. 2020.

[45] S. Feng, V. Balachandran, Y. Bai, and Y. Tsvetkov, "FactKB: Generalizable factuality evaluation using language models enhanced with factual knowledge," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (H. Bouamor, J. Pino, and K. Bali, eds.), (Singapore), pp. 933–952, Association for Computational Linguistics, Dec. 2023.

[46] W. Kryscinski, N. S. Keskar, B. McCann, C. Xiong, and R. Socher, "Neural text summarization: A critical evaluation," 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)*, (Hong Kong, China), pp. 540–551, Association for Computational Linguistics, Nov. 2019.

Jacob Parnell received a B.Sc degree in Astrophysics and a B.Phil degree in Planetary Science from Macquarie University, Sydney, in 2018 and 2019 respectively. At the end of 2023, he completed his PhD in Natural Language Processing from the University of Technology Sydney. Currently, he is a data scientist and NLP researcher at RoZetta Technology in Sydney, Australia, and an Adjunct Research Fellow at the University of Technology Sydney. His research interests include document summarisation and NLP more broadly, and the intersection of quantum mechanics with machine learning.



Inigo Jauregi Unanue received the BEng degree in telecommunication systems from University of Navarra, Donostia-San Sebastian, Spain, in 2016, and the PhD degree in natural language processing from University of Technology Sydney in 2020. From 2014 to 2016, he was a research assistant at Centro de Estudio e Investigaciones Tecnicas (CEIT). Currently, he is a natural language processing and machine learning researcher at RoZetta Technology in Sydney, Australia. His research interests include machine translation and low-resource natural language processing.

Scott Matthews has more than 25 years' industry experience in data science across

the property, insurance, marketing and market research sectors. He leads the Data Science team at RoZetta Technology which develops innovative analytical solutions across a diverse range of industry problems. Prior to joining RoZetta Technology, Scott was the Head Architect of Analytics at CoreLogic Australia. Scott is interested in the application of NLP, in particular entity identification & linking, and document summarisation, to large scale text processing problems faced in industry. He is a passionate supporter of RoZetta Technology's PhD program and has co-supervised a number of candidates as an industry partner with Prof. Massimo Piccardi from UTS. Scott received BAppSci (Hons 1) degree from CSU in 1995 majoring in Mathematics and Statistics.

Massimo Piccardi (SM'05) received the MEng and PhD degrees from the University of Bologna, Bologna, Italy, in 1991 and 1995, respectively. He is currently a Full Professor of computer systems with University of Technology Sydney, Australia. His research interests include natural language processing, computer vision and pattern recognition and he has co-authored over 180 papers in these areas. Prof. Piccardi is a Senior Member of the IEEE, a member of its Computer and Systems, Man, and Cybernetics Societies, and a member of the International Association for Pattern Recognition. He presently serves as an Associate Editor for the IEEE Transactions on Big Data.

Table 1: Quantitative results over five summarization datasets

Model	CNN/DM			
	R_1	R_2	R_L	F_{BERT}
CTRLSum*	38.71	16.54	26.79	88.01
T5**	31.14	13.21	23.57	87.71
BART	39.92	17.91	27.25	88.95
Prompt tuning [18]	40.12	18.11	27.52	88.70
GPT-4o mini (In-Context)	34.00	11.65	20.64	87.05
GPT-4o mini (YAKE!)	35.39	12.81	21.60	87.27
APG	**40.29**‡	**18.14**	**28.02**†	**89.05**†
Model	XSum			
	R_1	R_2	R_L	F_{BERT}
CTRLSum*	**40.80**	15.88	**30.85**	89.66
T5**	30.34	9.74	24.18	89.87
BART	37.62	15.44	29.88	90.69
Prompt tuning [18]	38.26	**16.08**	30.80	90.51
GPT-4o mini (In-Context)	21.92	4.74	14.91	86.96
GPT-4o mini (YAKE!)	21.29	4.28	14.31	86.73
APG	37.61	15.50	30.32†	**90.98**†
Model	DialogSum			
	R_1	R_2	R_L	F_{BERT}
CTRLSum*	33.76	7.95	26.18	89.10
T5	41.89	**16.61**	**34.31**	**91.36**
BART	41.19	16.05	33.58	90.72
Prompt tuning [18]	41.69	16.24	33.75	90.31
GPT-4o mini (In-Context)	32.65	10.82	24.83	89.42
GPT-4o mini (YAKE!)	26.83	8.18	19.85	87.15
APG	**42.05**†	16.38†	34.04†	91.16†
Model	SamSum			
	R_1	R_2	R_L	F_{BERT}
CTRLSum*	35.44	11.32	27.91	87.48
T5	40.39	18.78	33.78	90.65
BART	41.79	19.61	33.65	90.71
Prompt tuning [18]	43.19	20.68	34.25	90.57
GPT-4o mini (In-Context)	39.09	15.70	30.38	90.53
GPT-4o mini (YAKE!)	32.55	12.01	24.59	89.39
APG	**47.09**†	**22.19**†	**38.54**†	**91.92**†
Model	AESLC			
	R_1	R_2	R_L	F_{BERT}
CTRLSum*	3.91	0.98	3.87	82.99
T5	18.87	8.26	18.72	86.80
BART	20.09	9.02	19.87	74.24
Prompt tuning [18]	22.47	10.37	22.04	78.61
GPT-4o mini (In-Context)	18.26	7.36	16.69	84.59
GPT-4o mini (YAKE!)	12.28	4.73	11.08	84.25
APG	**29.59**†	**15.75**†	**28.96**†	**88.29**†

Table 2: Qualitative examples over the five summarization datasets

.

CNN/DM
Reference: Chelsea beat Stoke 2-1 in Premier League at Stamford Bridge on Saturday. Cesc Fabregas' nose was left bloodied after tussle with Charlie Adam. Midfielder showed off injury as he posted picture to Instagram after game. Fabregas joked that he might finally be able to get his nose fixed. **BART:** Cesc Fabregas was caught in the face by Charlie Adam's flailing arm. The Chelsea midfielder was left with a bloody nose and requiring treatment. Fabregas also took a whack on his leg during the clash on Saturday. Chelsea extended their lead at the top of the Premier League to seven points. *(F_{BERT}: 89.56)* **Prompt tuning:** Cesc Fabregas was caught in the face by Charlie Adam's flailing arm. The Spain international was left bloodied after being caught by a flailing leg. Fabregas took to Instagram to show off the nasty-looking gash on his nose. Chelsea extended their lead at the top of the Premier League to seven points. *(F_{BERT}: 89.36)* **GPT-4o mini (In-Context):** Cesc Fabregas showcased a significant gash on his nose after a clash with Charlie Adam during Chelsea's 2-1 victory over Stoke City. The win allowed Chelsea to extend their lead at the top of the Premier League to seven points, but came at a cost for Fabregas, who was left bloodied from the incident. He shared a photo of his injury on Instagram, humorously suggesting he might finally get his nose fixed. This is not the first time Fabregas has dealt with a nose injury, as he previously suffered a broken nose while playing for Spain in 2008. *(F_{BERT}: 89.94)* **APG:** Cesc Fabregas was caught by Charlie Adam's flailing arm during Chelsea's 2-1 win against Stoke City on Saturday. The Spaniard was left bloodied after being caught by Adam's arm. Fabregas also took a whack on his leg during the clash on Saturday night. Chelsea extended their lead at the top of the Premier League to seven points. *(F_{BERT}: 90.21)* (prompt: *ributedributedributedributedributedributedributed*)
XSum
Reference: There is a "chronic" need for more housing for prison leavers in Wales, according to a charity. **BART:** housing for ex-offenders who are homeless is "chronic", according to a charity which helps people find accommodation. *(F_{BERT}: 90.95)* **Prompt tuning:** housing for ex-offenders who are homeless in Wales has been a "desperate need" for more than 1,000 new one-bedroom flats, a charity has said. *(F_{BERT}: 91.36)* **GPT-4o mini (In-Context)**: got a roof over your head," he said. Prison Link Cymru reported a significant number of referrals in 2015-16, with many ex-offenders struggling to find housing for extended periods. The charity argues that investing in housing solutions would be more cost-effective than incarcerating homeless repeat offenders. Despite the Welsh *(F_{BERT}: 86.23)* **APG:** Prison leavers are being asked to consider moving to one-bedroom flats in Wales to help them deal with homelessness. *(F_{BERT}: 91.45)* (prompt: *jugjugjugjugjugche*)
DialogSum
Reference: Frank invites Betsy to the party to celebrate his big promotion. Betsy couldn't wait for the party. **BART:** #Person1# tells Frank that #Person1# got a promotion and invites him to a party. *(F_{BERT}: 89.33)* **Prompt tuning:** #Person1# tells Frank #Person1 #'ll throw a big party for all of #Person2#'s friends on Saturday *(F_{BERT}: 85.40)* **GPT-4o mini (In-Context)**: #Person1# has received a promotion and is planning a big party for 150 friends, inviting #Person2# to attend. #Person1# expresses excitement about the promotion, which allows them to afford a new house, but worries about work-life balance. #Person2# looks forward to the celebration. *(F_{BERT}: 87.62)*

Table 3: Human evaluation results

Model	Conciseness	Fluency
BART	3.02	3.93
APG	**3.15**	**4.07**

Table 4: Factuality evaluation with FactCC and FactKB

Model	**CNN/DM**	**XSum**	**DialogSum**	**SamSum**	**AESLC**
			FactCC		
BART	0.43	0.23	-	-	-
APG	**0.47**	**0.24**	-	-	-
			FactKB		
BART	**0.97**	0.24	0.54	0.41	0.29
APG	**0.97**	**0.26**	**0.58**	**0.71**	**0.45**

Table 5: Comparison with direct keyword prompting over the CNN/DailyMail dataset

Model	R_1	R_2	R_L	F_{BERT}
BART	40.84	18.60	27.92	88.97
Direct keyword prompting	39.71	17.54	27.25	87.97
APG	**41.30**	**18.91**	**28.25**	**89.05**

Table 6: Cross-dataset analysis over news and dialogue
dataset pairs

Dataset Pairs	Avg. ROUGE	F_{BERT}
	BART	
CNN/DM — XSum	11.35	86.11
XSum — CNN/DM	**13.57**	86.00
SamSum — DialogSum	20.76	87.65
DialogSum — SamSum	21.36	88.73
	APG	
CNN/DM — XSum	**11.64**$_{(+0.29)}$	**86.28**$_{(+0.17)}$
XSum — CNN/DM	12.69$_{(-0.88)}$	**86.09**$_{(+0.09)}$
SamSum — DialogSum	**21.72**$_{(+0.96)}$	**88.42**$_{(+0.77)}$
DialogSum — SamSum	**25.89**$_{(+4.53)}$	**90.05**$_{(+1.32)}$

.1 Model Hyperparameters

The baseline Transformer used in this work is BART-base [36]. This model has approximately 139M parameters, and the addition of the linear layer for expanding from the hidden state size, D, to the vocabulary size, V, to generate the prompt tokens (Eq. 2) adds an additional 38.6M parameters. As optimizer, we have used Adam with its default hyperparameters and a learning rate of 3×10^{-5}.

For the T5 model, we have used T5-base [37]. This model is larger than BART-base, with approximately 247M parameters. For training, we have used the AdamW optimizer with its default hyperparameters and a learning rate of 3×10^{-4}.

Both the BART and T5 model implementations have been built on top of PyTorch Lightning[5]. The full lists of the hyperparameters is shown in Table 7. Where applicable, (*) denotes early stopping which is enforced if the monitored metric does not improve over two consecutive epochs. By default, the monitored metric is the validation loss.

For the replicated CTRLSum model we have used its original Fairseq[6] implementation and the hyperparameters shown in Table 8, with the aim to compare it with the other models in the fairest way possible. All other hyperparameters have been kept to their default values from the original paper [22] and GitHub repository[7]. In addition, we report the maximum number of training steps used for the various datasets, which correspond to exactly the same training times used for training the proposed APG model in PyTorch Lightning.

For the implementation of GPT-4o mini, we have used the OpenAI platform between the 30th of July and the 2nd of August 2024. Below, we detail more formally the structure of the four prompting variations we have utilized in our GPT-4o mini baselines. In the case of the In-Context approach, we have used the same document-summary pair (as a representative of the training set) in the prompt for each example generated in the test set.

[5] https://github.com/PyTorchLightning/pytorch-lightning

[6] https://github.com/facebookresearch/fairseq

[7] https://github.com/salesforce/ctrl-sum

Table 7: Training and evaluation hyperparameters.

Hyperparameter	**Value**	
	BART	T5
Learning Rate	3×10^{-5}	3×10^{-4}
Training Epochs*	20	20
Batch Size	8	8
Beam Size	1	8
Optimizer	Adam	AdamW
β_1	0.9	0.9
β_2	0.999	0.999
Warm-up	1000	1000
Weight Decay	0.00	0.01
	GPT-4o mini	
Temperature	0.7	
Max Tokens	Dataset dependent	
Frequency Penalty	0.0	
Logit Bias	null	
Log Probs	False	
N	1	
Presence Penalty	0.0	

Table 8: CTRLSum replication hyperparameters

Hyperparameter	**Value**
Pre-Processing	
Max Source Length	1024 tokens
Max Target Length	*
Keyword Selection Threshold	0.1
Max # Keywords	10
Max # Sentences	3
Training	
Max Steps	Dataset dependent
Summarization Model	BART-base
Inference	
Sequence Tagger Model	BERT-large-cased
Summarization Beam Size	1
Max Generation Length	Dataset dependent
Maximum Number of Training Steps	
CNN/DM	107,670 steps
XSum	127,510 steps
DialogSum	9,726 steps
SamSum	7,780 steps
AESLC	10,250 steps

1. **Direct**

```
Prompt: Please summarize the
    following text.
```

2. **YAKE!**

```
Prompt: Here is a list of
    relevant keywords: {keyword_1,
     keyword_2, ..., keyword_n}.
    Please use them to generate a
    summary of the following text.
```

3. **Self-Extract**

```
Prompt: Please extract keywords
    from the following text and
    use them to generate a summary
    . Do not return the keywords
    in your response.
```

4. **In-Context**

```
Prompt: Here is an example of a
    document-summary pair,
    enclosed in triple quotes:
    """[DOCUMENT] {document} [
    SUMMARY] {summary}""". Please
    summarize the following text.
```

.2 CTRLSum: Performance vs. Training Time

In this Appendix we explore the performance of the replicated CTRLSum model as a function of the amount of training time. To this aim, Fig. 3 shows the ROUGE scores of the replicated CTRLSum model for an increasing number of training steps for a) CNN/DailyMail and b) SamSum. For comparison, the horizontal dashed lines show the scores obtained by APG, and the vertical black lines show the number of training steps at which they were obtained.

For CNN/DailyMail, the original paper [22] had capped the training time of CTRLSum to 30,000 training steps, which for our replicated model with BART-base has obtained modest evaluation scores, as shown by the first data points in Fig. 3a. We can see that also at a parity of training time the scores of the replicated CTRLSum model have been worse than those of APG. However, with increasing training steps they have eventually surpassed them (at approximately 1.5x the training time).

For SamSum, in contrast, the scores of the replicated CTRLSum model have not been able to surpass those of the APG model even with an uncapped number of training steps, as shown in Fig. 3b. The performance has plateaued after approximately 2x the training time, showing no appreciable, further performance gains. This shows that the performance of the replicated CTRLSum model cannot always be increased by extending its training beyond equivalent parity. Therefore, the remarkably higher performance of the proposed APG model has to be ascribed to its different mechanism for generating the prompt.

Table 9: Examining the effect of prompt length on validation
ROUGE and F_{BERT}

N	CNN/DM		SamSum	
	Avg. ROUGE	F_{BERT}	Avg. ROUGE	F_{BERT}
5	29.54	89.15	36.86	92.00
10	**29.71**	**89.19**	**37.07**	**92.07**
15	29.38	89.09	36.68	92.01

.3 Impact of the Prompt Length

To explore the impact of the number of training keywords and the consequent prompt length, Table 9 shows the average ROUGE scores and F_{BERT} scores for the proposed APG model over the validation splits of the CNN/DailyMail and SamSum datasets (which have been used for selecting all the hyperparameters). The differences in scores are rather small, showing a desirable insensitivity to this hyperparameter. However, 10 tokens have led to the best scores in all cases and therefore have been used across all the experiments.

In addition, we have explored the impact of increasing the prompt length in the prompt tuning approach (the original paper [18] reported a best per-

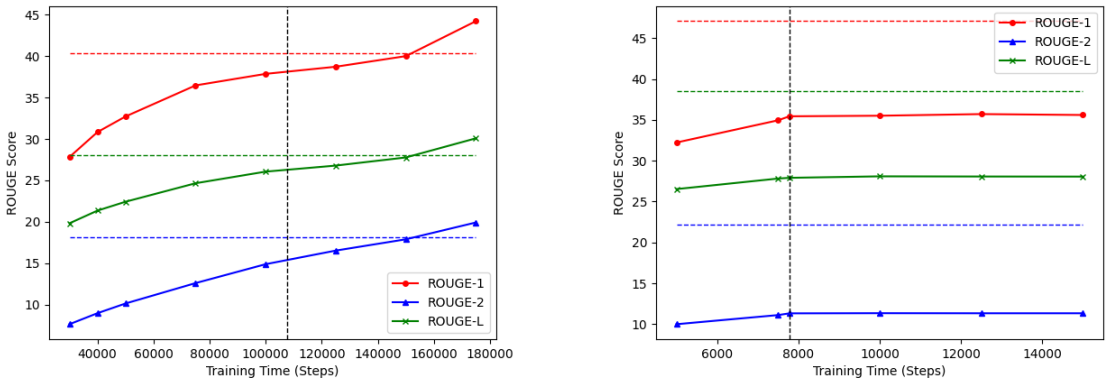


Figure 3: ROUGE scores for CTRLSum for varying numbers of training steps for CNN/DailyMail (left) and SamSum (right). ROUGE R_1 is shown in red, R_2 in blue, and R_L in green. For comparison, the horizontal dashed lines show the APG scores, and the vertical black lines are the number of training steps at which they were achieved.

formance for 100). To this aim, we have iterated the experiment on the SamSum dataset with a prompt length of 10 (same as APG), 50, and 100. Table 10 shows that increasing the prompt length has actually decreased the ROUGE scores, while the F_{BERT} score has only minimally improved.

Table 10: Prompt tuning with increasing prompt length

Model	R1	R2	RL	F_{BERT}
BART	41.79	19.61	33.65	90.71
Prompt tuning ($N = 10$)	43.19	20.68	34.25	90.57
Prompt tuning ($N = 50$)	42.68	20.37	34.49	90.62
Prompt tuning ($N = 100$)	41.98	20.32	34.09	90.64
APG	**47.09**	**22.19**	**38.54**	**91.92**

.4 Prompt Tuning: Random vs. Sampled Initialization

Lester et al. in [18] have proposed two ways to initialize the embeddings of their prompt tokens: 1) randomly or 2) by assigning them with the embeddings of actual tokens uniformly sampled from the 5000 most-frequent tokens in the vocabulary. For comparison, we have repeated all experiments with both initializations and reported the results in Table 11. While the differences are rather small, it is clear that the random initialization has performed better across all five datasets, and for this reason we have reported its results in the main paper. We also note that BART's vocabulary is only approximately sorted in descending frequency order, so we have approximated the N most-frequent tokens with the top N non-special tokens.

.5 Larger Model Experiments

All the results for the fine-tuned approaches presented in this paper have leveraged base-sized Transformer models, and it is impossible to extrapolate them to larger models without a dedicated investigation. As a small experiment, we have carried out a run of APG and the BART baseline on the SamSum dataset by using the BART-large-CNN model[8], a BART-large model pre-trained on the CNN/Daily-

[8] https://huggingface.co/facebook/bart-large-cnn

Table 11: Random vs. sampled embedding initialization

Model	CNN/DM			
	R_1	R_2	R_L	F_{BERT}
Random	40.12	18.11	27.52	88.70
Top-5000	40.14	18.06	27.51	88.70
Model	XSum			
	R_1	R_2	R_L	F_{BERT}
Random	38.26	16.08	30.80	90.51
Top-5000	37.99	15.80	30.43	90.37
Model	DialogSum			
	R_1	R_2	R_L	F_{BERT}
Random	41.69	16.24	33.75	90.31
Top-5000	41.52	16.14	33.74	90.32
Model	SamSum			
	R_1	R_2	R_L	F_{BERT}
Random	43.19	20.68	34.25	90.57
Top-5000	42.80	20.36	34.16	90.56
Model	AESLC			
	R_1	R_2	R_L	F_{BERT}
Random	22.47	10.37	22.04	78.61
Top-5000	21.62	9.61	21.19	78.65

Mail dataset. The results, reported in Table 12, show that, while the performance gap has significantly reduced, the proposed approach has still achieved overall better performance metrics. In addition, the dedicated pre-training with the CNN/DailyMail dataset has proved counterproductive for performance, rather than improving it, possibly due to the different characteristics of these datasets.

Table 12: An experiment with a larger model

Model	R1	R2	RL	F_{BERT}
BART (large)	40.49	19.97	31.96	89.19
APG (large)	**42.60**	**20.71**	**33.54**	**89.41**

.6 Additional Results for GPT-4o mini

In Table 13 we provide the results for the other two prompting variations of GPT-4o mini. The trends are similar to those of the other two prompts, yet less competitive in most cases.

Table 13: Additional results for GPT-4o mini

Model	CNN/DM			
	R_1	R_2	R_L	F_{BERT}
GPT-4o mini (Direct)	34.95	12.37	21.60	87.39
GPT-4o mini (Self-Extract)	33.38	11.12	20.34	87.15
Model	XSum			
	R_1	R_2	R_L	F_{BERT}
GPT-4o mini (Direct)	22.01	4.79	14.91	87.02
GPT-4o mini (Self-Extract)	21.53	4.49	14.52	86.99
Model	DialogSum			
	R_1	R_2	R_L	F_{BERT}
GPT-4o mini (Direct)	28.95	8.72	21.66	87.72
GPT-4o mini (Self-Extract)	23.78	6.78	17.55	86.94
Model	SamSum			
	R_1	R_2	R_L	F_{BERT}
GPT-4o mini (Direct)	34.71	12.71	26.30	89.86
GPT-4o mini (Self-Extract)	32.05	10.83	23.96	89.50
Model	AESLC			
	R_1	R_2	R_L	F_{BERT}
GPT-4o mini (Direct)	12.14	4.56	10.81	84.32
GPT-4o mini (Self-Extract)	11.77	4.16	10.49	84.41

.7 Behavior of the Training Loss

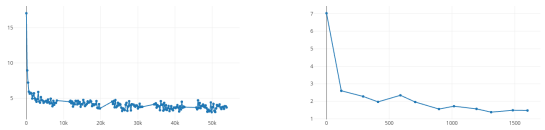
The training loss presented in Eq. 4 is not strictly monotonic since it is not possible to backpropagate through the generated, discrete prompt tokens. For this reason, we have recorded the values of the loss using a logger (MLFlow[9]) and explored its behavior. The results for CNN/DailyMail and SamSum are plotted in Fig. 4, showing that the behavior has been desirably smooth and decreasing for both datasets. For clarity, the frequency of the logs has been set to be approximately proportional to the maximum

[9] `https://mlflow.org/`

Table 14: Main information of the datasets used in the experiments
(no mark: original paper; *: NLTK tokenizer)

Dataset	Train	Test	Val	Input	Target	Input Tokens	Output Tokens	Link
CNN/DM	287k	13.4k	11.5k	781 tokens	56 tokens	512	128	https://huggingface.co/datasets/ccdv/cnn_dailymail
XSum	204k	11.3k	11.3k	431 tokens	23 tokens	512	64	https://huggingface.co/datasets/xsum
DialogSum*	12.4k	1.5k	500	177 tokens	30 tokens	512	64	https://huggingface.co/datasets/knkarthick/dialogsum
SamSum	14.7k	819	818	121 tokens*	22 tokens*	512	128	https://huggingface.co/datasets/samsum
AESLC	14.4k	1.9k	1.9k	75 tokens	4 tokens	512	32	https://huggingface.co/datasets/aeslc

number of training steps, and we have skipped the logging during the validation iterations. Hence, for CNN/DailyMail the logging frequency has been much higher than for SamSum and shows the plateauing of the loss more neatly. However, in both cases the empirical convergence is clearly visible.



(a) (b)

Figure 4: Training loss along the training iterations for CNN/DailyMail (left) and SamSum (right).

.8 Dataset Links and Details

For reproducibility, Table 14 reports the statistics pertinent to the specific datasets used in the paper, the input and output tokens we cap the data to based on those statistics which somehow reflects the average lengths of their documents and reference summaries, and the links to all the datasets which are available via Hugging Face. For CNN/DailyMail, the version is slightly edited from the original dataset due to errors widely reported by the research community.