ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Scienze
Corso di Laurea in Ingegneria e Scienze Informatiche

SURVEY ON FEW-SHOT SUMMARIZATION

Elaborato in
Programmazione Di Applicazioni Data Intensive

Relatore
Prof. Gianluca Moro

Presentata da
Emanuele Artegiani

Co-relatori
Dott. Luca Ragazzi
Dott. Giacomo Frisoni

Seconda Sessione di Laurea
Anno Accademico 2022 – 2023

KEY WORDS

Natural Language Processing

Low-Resource Summarization

Few-shot Summarization

Data Augmentation

Pre-Training & Fine-Tuning

To everyone who has been there for me,
and helped me reach this goal.

Abstract

Low-Resource Summarization (LRS), chiamato anche *Few-shot Summarization*, si riferisce all'operazione di creare riassunti concisi e coerenti partendo da contenuti testuali quando l'accesso ai dati di addestramento è limitato. Questo argomento di ricerca ha suscitato grande interesse da parte di una vasta comunità di ricercatori ed è attualmente considerato una delle aree di ricerca più utili per le applicazioni del mondo reale, come la sintesi di cartelle cliniche, documenti legali oppure di campi di studio emergenti o molto specializzati. Questo studio offre una panoramica approfondita e attuale dei metodi di LRS esistenti. Include, inoltre, definizioni formali di termini chiave rilevanti per il ramo dell'*apprendimento automatico* (ML) preso in considerazione. In primo luogo, per aiutare i ricercatori ad orientarsi nella moltitudine di lavori relativi a LRS, proponiamo una tassonomia dettagliata per classificare i contributi presentati dalla comunità. In secondo luogo, definiamo chiaramente il termine "few-shot", spesso usato in modo ambiguo. In terzo luogo, abbiamo stilato delle classifiche confrontando ed analizzando 20 proposte per risolvere il compito della sintesi di dialoghi e documenti in contesti con scarsità di dati su 5 dataset differenti ed utilizzando una metrica comune.

Low-Resource Summarization (LRS), also called *Few-shot Summarization*, refers to the task of creating concise and coherent summaries from textual content when there is limited access to training data. This research topic has gained great interest from a sizable research community, and it is currently regarded as one of the most useful research areas for real-world applications, such as medical records and legal and emerging or very specialized fields of study document summarization. This

study offers a thorough and current overview of the existing methods of LRS. It also includes formal definitions of key terms relevant to the branch of *machine learning* (ML) taken into consideration. First, to help researchers navigate the avalanche of LRS works, we propose a detailed taxonomy for classifying the contributions presented by the community. Second, we clearly define the term "few-shot", which is often used ambiguously. Third, we constructed leaderboards after comparing and analyzing 20 solutions for dialogue and document summarization tasks in low-data settings using 5 datasets and one common metric.

Index

List of Figures

List of Tables

Introduction

Nell'era digitale, l'enorme volume di informazioni testuali disponibili su Internet, relativi ad innumerevoli ambiti, ha reso necessario lo sviluppo di sistemi di riassunto automatico dei testi. Questi sistemi, basati su tecniche di elaborazione del linguaggio naturale (NLP) e di ML, distillano lunghi documenti in riassunti concisi, coerenti e informativi. Essi trovano applicazione in diversi campi, dalla sintesi di articoli giornalistici al recupero di informazioni e altro ancora. Tuttavia, nonostante i notevoli progressi compiuti nella sintesi automatica dei testi, persiste una sfida importante — LRS. In questo articolo, LRS si riferisce solo al compito di riassumere l'input quando i dati di addestramento sono pochi. Ma, può anche riferirsi alla sintesi di testi in lingue con risorse linguistiche limitate, caso analizzato nei documenti [18] e [19], come le lingue minoritarie o meno studiate, o in situazioni in cui le risorse computazionali sono limitate [20].

LRS riguarda il difficile compito di generare riassunti di alta qualità quando ci si trova di fronte a limitati dati di addestramento etichettati. Questa sfida è particolarmente pronunciata nei contesti linguistici meno studiati, nelle lingue con poche risorse o quando si ha a che fare con argomenti emergenti e domini specializzati che non dispongono di dati di formazione annotati. Lo sviluppo di modelli di sintesi robusti ed efficaci diventa un'impresa ardua in questi scenari.

L'obiettivo di questa tesi è offrire un'ampia esplorazione dell'intricato panorama del LRS. Esaminando sistematicamente le ricerca esistenti, ci proponiamo di fornire ai lettori una comprensione approfondita delle sfide uniche poste dagli scenari a basse risorse e delle strategie innovative ideate per affrontarle. Questo documento copre un ampio spettro di metodi, tra cui il meta-learning (MTL), il pre-training specifico, il prefix-tuning e gli approcci creativi di aumento dei dati; tutti progettati per superare le limitazioni delle configurazioni a basse risorse.

Consideriamo solo i documenti con analisi quantitative per la comparabilità e la riproducibilità. Ad esempio, sono esclusi lavori come [21] che contengono solo valutazioni umane. Negli ultimi anni l'LRS ha riscosso un notevole interesse da parte dei ricercatori. Utilizzando l'analizzatore dei risultati di ricerca di Scopus, si può notare che l'affermazione precedente è confermata dal numero crescente di pubblicazioni fatte negli ultimi anni in questo contesto (Figura 3.2).

I principali contributi che abbiamo fornito alla comunità con questo articolo sono stati: proporre una tassonomia formale per identificare le varie categorie di approcci LRS, fornire una definizione precisa del termine "few-shot", che viene spesso utilizzato quando si parla di LRS ma con molti significati diversi e organizzare classifiche, sotto un'unica metrica di valutazione, di 20 lavori esistenti in base ai dataset e al contesto di applicazione.

I prossimi capitoli di questa tesi forniscono un'esplorazione approfondita della nostra ricerca, chiariscono le tecniche e spiegano i risultati:

- **Capitolo 1 - Quadro Teorico**: In questo capitolo introduttivo, stabiliamo le basi per comprendere le idee chiave che guidano la nostra ricerca. Si inizia con un'analisi del mondo del NLP. Si esplora poi il mondo della sintesi e si discute brevemente la sua funzione nell'elaborazione linguistica contemporanea, per arrivare infine alla LRS. Questo capitolo introduce il lettore alla mia tesi e gli fornisce le informazioni necessarie per comprendere la ricerca nei capitoli successivi.

- **Capitolo 2 - Opere Correlate**: In questo capitolo esaminiamo il panorama delle ricerche e degli studi recenti sul tema della sintesi, con particolare attenzione alla sintesi astrattiva, che ha suscitato un certo interesse da parte del pubblico.

- **Capitolo 3 - Metodo**: Questo capitolo fornisce un'esplorazione approfondita del lavoro che ha portato alla stesura di questa tesi, illustrando le definizioni proposte, i criteri di selezione dei documenti da considerare e una spiegazione esaustiva di questi ultimi e dei dataset scelti.

- **Capitolo 4 - Risultati e Discussioni**: In questo capitolo vengono presentati i risultati degli esperimenti e delle analisi. Esaminiamo i dati, esponendo

i risultati quantitativi ed evidenziando l'efficacia dei metodi suggeriti. Presentiamo interpretazioni intelligenti di questi risultati attraverso spiegazioni approfondite.

- **Conclusioni e Lavori Futuri**: Questa sezione finale della tesi riassume i contributi e le analisi più importanti della nostro survey. Consideriamo l'importanza del nostro lavoro e i suoi possibili effetti. Descriviamo inoltre i potenziali aggiornamenti che potrebbero essere apportati per mantenere l'utilità del nostro lavoro anche in futuro.

In the digital age, the sheer volume of textual information available across the internet and various domains has necessitated the development of automatic text summarization systems. These systems, powered by natural language processing (NLP) and ML techniques, distillate lengthy documents into concise, coherent, and informative summaries. They find applications in diverse fields, from news article summarization to information retrieval and more. However, while significant progress has been made in text summarization, a significant challenge persists — LRS. In this paper, LRS refers only to the summarization task when there is a scarcity of training data. It can also refer to summarizing text in languages with limited linguistic resources, case analyzed in the papers [18] and [19], such as minority or less-studied languages, or in situations where computational resources are restricted, such as devices with low computational power or memory, as in [20].

LRS pertains to the formidable task of generating high-quality summaries when confronted with labelled training data limitations. This challenge is particularly pronounced in linguistic contexts that are less studied, under-resourced languages, or when dealing with emerging topics and specialized domains that lack annotated training data. Developing robust and effective summarization models becomes a formidable endeavour in such scenarios.

The objective of this thesis is to offer an extensive exploration of the intricate landscape of LRS. By systematically examining existing research, we aim to provide readers with an in-depth understanding of the unique challenges posed by low-

resource environments and the innovative strategies devised to address them. This paper covers a broad spectrum of methods, including meta-learning (MTL), specific pre-training, prefix-tuning and creative data augmentation approaches designed to surmount the limitations of low-resource settings.

We only consider papers with quantitative analysis for comparability and reproducibility. For instance, papers such as [21] containing only human evaluations are excluded. LRS has gained significant appeal from researchers in recent years. By utilizing the Scopus[1] search results analyzer, it can be noted that the previous statement is confirmed by the increasing number of publications made over the past few years in this context (Figure 3.2).

The main contributions we made to the community with this paper were to propose a formal taxonomy to identify the various LRS approach categories, provide a precise definition of the term "few-shot", which is frequently used when discussing LRS but with many different meanings and organize leaderboards, under a single evaluation metric, of 20 existing works according to datasets and context of application.

The next chapters of this thesis provide an in-depth exploration of our research, clarify techniques, and explain findings:

- **Chapter 1 - Theoretical Framework**: In this introductory chapter, we establish the foundations for comprehending the key ideas that guide our research. We start by looking into the world of NLP. The world of summarization is next explored, and its function in contemporary language processing is briefly discussed, finally arriving at LRS. This chapter introduces the reader to my dissertation and gives them the information they need to understand the research in the following chapters.

- **Chapter 2 - Related Work**: In this chapter, we examine the landscape of recent research and studies on the subject of summarizing, with a focus on abstractive summarization, which has attracted some interest from the general public.

[1] `https://www.scopus.com/`

- **Chapter 3 - Method**: This chapter provides a thorough exploration of the work that led to this thesis's writing, explaining the proposed definitions, the criteria for choosing the papers to consider and an exhaustive explanation of all of them and of the datasets selected.

- **Chapter 4 - Results and Discussion**: The results of our experiments and analysis are presented in this chapter. We examine the data, exposing the quantitative findings and highlighting the suggested methods' effectiveness. We present intelligent interpretations of these outcomes through in-depth conversations.

- **Conclusion and Future Work**: This final section of my thesis summarizes our survey's most important contributions and analysis. We consider the importance of our work and its possible effects. We also describe the potential updates that could be made in the future to maintain the usefulness of our work.

Chapter 1

Theoretical Framework

1.1 Natural Language Processing

The field of computer science known as NLP is more particularly the branch of *artificial intelligence* (AI) concerned with providing computers with the ability to understand spoken and written language similarly to humans.

NLP combines computational linguistics — rule-based modelling of human language — with machine learning, deep learning, and statistical models. With these technologies, computers can now process human language in the form of text or audio data and completely "understand" what is being written or spoken, taking into account the thoughts and intents of the speaker or writer.

Computer programs that translate text between languages reply to spoken commands, and quickly summarize vast amounts of text—even in real-time — are all powered by NLP. You've probably used NLP voice-activated GPS units, chatbots for customer support, digital assistants, speech-to-text dictation apps, and other consumer perks. Nonetheless, NLP is being more widely used in corporate solutions to improve worker productivity, streamline mission-critical business processes, and streamline business operations.

It is extremely challenging to develop software that accuratly determines the intended meaning of text or voice data since human language is rife with ambiguity. Homonyms, homophones, sarcasm, idioms, metaphors, exceptions to the rules of grammar and usage, and changes in sentence structure are just a few examples

of the irregularities in human language that take humans years to learn, but that programmers must teach natural language-driven applications to recognize and understand accurately from the beginning if those applications are to be useful. In order to help the computer understand the speech and text data that it is taking in, several NLP activities deconstruct human text and voice data. These are only a few of these tasks: speech recognition (speech-to-text), part of speech tagging, word sense disambiguation, named entity recognition, co-reference resolution, sentiment analysis and natural language generation. In many contemporary real-world applications, machine intelligence is powered by NLP. Here are a few illustrations: spam detection, machine translation, chatbots, social media sentiment analysis and text summarization. You can find detailed explanations of the tasks mentioned earlier in [22].

1.2 Large Language Models

Large language models (LLMs) are large, general-purpose language models (LMs) that can be pre-trained and then fine-tuned for certain tasks. The word *large* has two connotations: immense training data sets (can be up to petabytes in size) and huge number of parameters. To tackle typical NLP tasks like text classification, question-answering, text summarization, and text production, an LLM is simply a Transformer-based neural network [23]. The model's objective is to forecast the text that will probably come next.

With the introduction of transformers and transfer learning, the modification of language models for different tasks initially only required a minor enlargement of the network's final layers (the head), followed by fine-tuning. This strategy, nevertheless, is now out of date. Modern technology has advanced to the point where several different activities may be successfully completed with the same LLM by merely changing the prompt's instructions.

To maximize the potential of LLM, prompts must be effective, and their creation requires talent (*Prompt Engineering*). The number of parameters in a model, which represents the variety of aspects it takes into account while producing output, can be used to assess a model's complexity and effectiveness. LLMs undergo substantial data pre-training to become familiar with the complexities and relationships of

language. Pre-training requires significant computational resources and cutting-edge gear to complete this critical phase. These models can be modified for downstream (particular) tasks by using methods like *fine-tuning* and *contextual learning*.

1.3 Different Types of LMs

Encoder-only, encoder-decoder and decoder-only models are the three main subtypes of transformer models. These varieties go through training with various goals, giving them specific skills for a range of tasks. A comparative overview of these various model architectures and their pre-training objectives can be found in Figure 1.1.

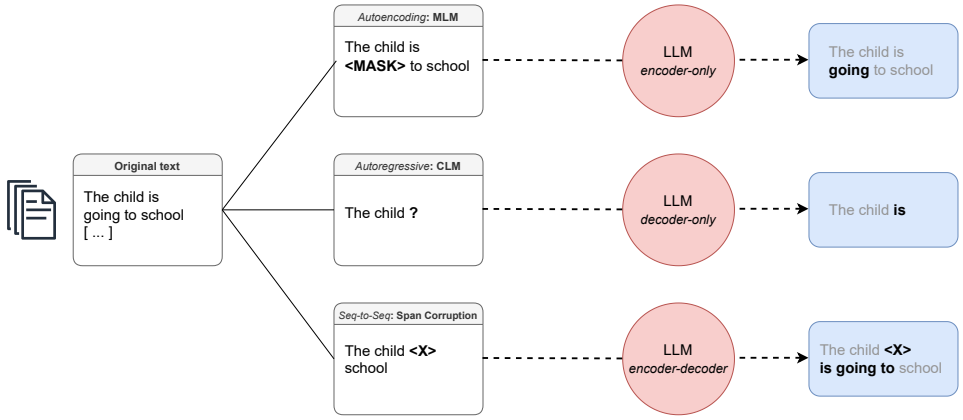


Figure 1.1: Model architectures and pre-training objectives.

Summary of different model architectures and targets of the pre-training objectives. Auto-encoding models use Masked Language Modelling, similar to the encoder in the original Transformer and appropriate for tasks like text categorization. The decoder is used for text production in autoregressive models built on the Causal Language Modeling framework. Sequence-to-sequence models use encoder and decoder components for various pre-training goals, including tasks like translation and summarization.

1.3.1 Autoencoding

Autoencoding models, also known as encoder-only models, go through pre-training using *Masked Language Modeling* (MLM). The main goal of this training

method is to anticipate these masked tokens in order to reconstruct the original sentence. Tokens inside the input sequence are randomly masked by using *mask tokens* (for example, <MASK>). Commonly referred to as a denoising objective, this technique. Autoencoding models excel at capturing bidirectional representations of the input sequence, allowing them to understand the full context of a token rather than just taking into account the words that came before it. These models are useful for a variety of NLP tasks, including token-level tasks like Named Entity Recognition (NER) and word classification, as well as sentence-level ones like sentiment analysis. BERT and RoBERTa [24, 25] are two prominent instances of autoencoding models.

1.3.2 Autoregressive

Autoregressive models are a frequent name for decoder-only models. To predict the following token in a sequence based on the previous tokens, these models undergo a pre-training procedure using *Causal Language Modeling* (CLM). In academic literature, this predictive endeavour is frequently referred to as *full language modeling*. Compared to their encoder-based predecessors, decoder-based autoregressive models are fundamentally different. They conceal the input order and can only access data from tokens before the one being considered. Because they are unaware of the conclusion of the phrase, these models proceed through the input sequence token by token, predicting each character in turn. Decoder-only models work in a unidirectional setting, in contrast to encoder-based designs. These models learn to produce coherent and contextually relevant text by lengthy training on a large corpus of textual data. Larger decoder-only models have powerful zero-shot inference skills, enabling them to succeed across a wide range of NLP tasks, even though they are typically used for text production tasks. GPT and BLOOM [26, 27] are two well-known examples of decoder-based autoregressive models in NLP.

1.3.3 Sequence-to-Sequence

The sequence-to-sequence model, which combines the encoder and decoder elements from the original transformer architecture, is the last variation in the transformer model family. The pre-training goals for these models vary depending on the implementation. For instance, during the pre-training phase of the encoder,

when input token sequences are randomly masked, one well-known sequence-to-sequence model, T5 [28], makes use of *span corruption*. The *Sentinel token* (e.g. <X>) is then used in place of these masking sequences. Sentinel tokens are unique tokens that have been added to the model's vocabulary but do not directly correlate to any words in the input text. The decoder's job is to reconstruct the chains of masked tokens in an auto-regressive fashion, with the predicted tokens coming after the Sentinel token as the output. Sequence-to-sequence models are useful for translating, summarizing, and answering questions. They are also useful for dealing with text input and text output. T5 is not the only notable encoder-decoder model in this class; BART [29] is another.

1.4 Text Summarization

Condensing a lengthy text document into a shorter, more compact version while keeping the key details and meaning is the challenge of text summarization, an NLP technique. The objective is to create a summary that concisely and accurately captures the essence of the original material. Text summarizing can be done in various ways, such as *extractive* methods that extract important sentences or phrases from the text or *abstractive* methods that create new text based on the content of the original one.

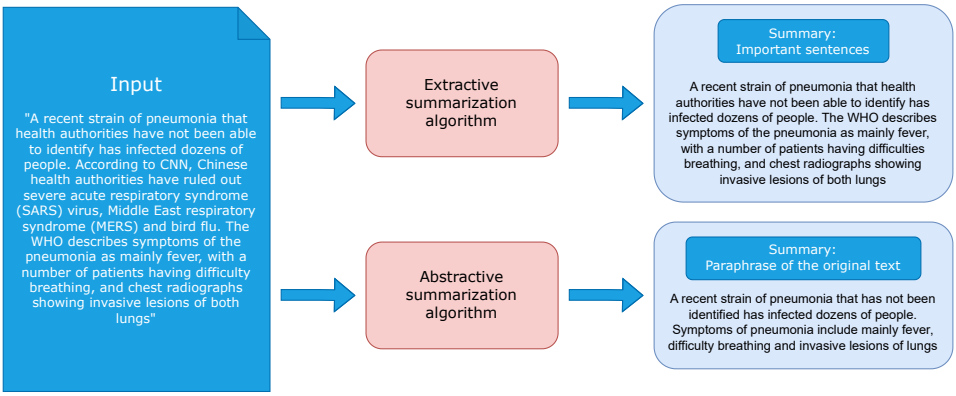


Figure 1.2: Types of text summarization with example [1].

1.4.1 Extractive summarization

In extractive summarization (ES), key sentences or phrases are selected directly from the source text to create a summary. These selected sentences are typically deemed most informative or representative of the original content, preserving the original wording and structure. ES does not involve generating new sentences but instead identifies and extracts the most relevant content for summarization. This approach makes it possible to quickly grasp the main points of a speech unaltered, extrapolating the main arguments from news articles, court documents, academic papers, meetings, emails, and much more.

1.4.2 Abstractive summarization

Abstractive summarization (AS) is a method that goes beyond simply taking phrases out of the original text to provide a logical and succinct summary. AS entails comprehending the meaning of the text and creating new phrases that communicate the essential information, as opposed to merely copying and pasting pre-existing sentences. This technique frequently involves rephrasing and paraphrasing the original content, producing summaries that may be more succinct and human-like. When a more succinct, readable output with strong sentence links and coherence is required, the AS technique might be right choice instead of the ES approach.

1.5 Low-Resource summarization

The task of producing brief and relevant summaries when there is limited access to training data is referred to as few-shot summarizing or LRS. Conventional machine learning scenarios include training a model on large datasets to identify trends and produce precise results. Nevertheless, it can be difficult to gather sizable, annotated datasets in environments with limited resources. This problem is addressed by few-shot summarization, which trains models using a small number of examples — possibly as few as one or a dozen per task.

LRS methods often use *transfer learning*, in which models are fine-tuned using restricted task-specific data after being pre-trained on considerable datasets to learn general language patterns. With this strategy, the model may accomplish targeted

summarization tasks with minimum training instances by generalizing from the more comprehensive information acquired during pre-training. *Prefix tuning*, which improves the model's adaptability and enables it to provide task-specific and contextually relevant responses, and *data augmentation*, which entails inventive methods to increase the number of training samples, are further methodologies employed in LRS.

Chapter 2

Related Work

Complete overviews of the *State-Of-The-Art* (SOTA) are crucial for the research community in the quickly expanding field of NLP as they greatly expedite the search for related works and findings in a particular study area. Surveys significantly improve research efficiency, especially when discussing an area that is constantly and extremely quickly updating, like ML.

2.1 A Survey on Dialogue Summarization: Recent Advances and New Frontiers

One important strategy to lessen dialogue data overload is summarizing, which distils the original dialogue into a shorter version covering only significant information. Lately, there has been a notable shift in the field of natural language production techniques and dialogue systems, resulting in major research interest and a new terrain. Nevertheless, a thorough survey for this task is still lacking. In order to do this, Feng et al. [30] initiate the process by providing an extensive and meticulous review of this field of study. They cover meetings, chat, email threads, customer support, and medical dialogues, methodically grouping the existing works in detail based on the traits of each domain. They also arrange two leaderboards under common criteria and offer an overview of research datasets that are accessible to the general public. They also share their opinions and discuss potential future directions, such as faithfulness, multi-modal, multi-domain, and multi-lingual dialogue summarization.

The authors of the paper analyzed different dialogue summarization scenarios:

- Meetings.
  Feng et al. discover that extractive approaches are inappropriate for meeting summaries because of the multi-participant nature of the meeting, which causes information to be dispersed and incoherent. Despite the powerful modeling capabilities of deep learning-based techniques, literal information alone is insufficient. This is because meeting utterances contain a variety of interactive signals. As a result, several research have been highlighted that focus on including auxiliary information for improved meeting modeling.

- Chats.
  Generally speaking, two aspects are heavily stressed in most current works: *dialogue interaction modeling* and *dialogue participant modeling*. These aspects align with the salient features of conversational data. Current dialogue summarization systems typically encode the text with additional information, which is typically obtained via open-domain toolkits that are inappropriate for dialogues, as demonstrated by the works analyzed. Therefore, Feng et al. present an unsupervised DialoGPT [31] annotator capable of performing three tasks related to dialogue-specific annotation: keyword extraction, redundancy detection, and topic segmentation. The results show that the existing techniques, using pre-trained language models, are adept at turning the original conversation into a straightforward summary. Still, they are not very good at determining crucial details and frequently cause hallucinations. In the future, this work should investigate further low-resource environments and powerful chat modeling methodologies and reasoning abilities.

- Email Threads.
  Email is a particular type of discourse that tries to streamline workflow. Consequently, an email often contains action items, demands, and commitments, which makes interpreting the communication's aim crucial. Future research should focus more on comprehending the email's coarse-grained intent and its fine-grained action items. Additionally, making better use of quotations can have a significant impact. Many excellent annotated samples are available

for email summaries, and the authors discover that both ES and AS work well alone and in tandem.

- Customer Service.
  Customer service aims to respond to inquiries made by representatives. As a result, it naturally has strong motivations, which gives the conversation a particular progression as it involves two parties with unique qualities—the consumer and the agent. Therefore, it is crucial for this endeavour to model participant roles, evolution chains, and inherent issues. To this end, some works investigate *topic modeling* for this purpose, using pre-established or dynamic themes; other works create a unique summary for every member. In addition, some fine-grained data, including slots, states, and intents, should be considered to guarantee faithfulness.

- Medical.
  Medical discourse summaries should be more faithful than innovative, with the primary goal being to assist physicians in finishing electronic health records as soon as possible. Thus, it is preferable to combine extractive approaches with straightforward abstractive ones. Semistructured summaries can be created using the topic information as a guide. Furthermore, handling negations and terms carefully in medical discourse is crucial. Many studies focused on improving the copy mechanism to make copying from the input easier.

2.2 Pretrained Language Models for Text Generation: A Survey

Text generation has emerged as a crucial and demanding problem in NLP. The discipline of deep learning has made significant advancements thanks to the resurrection of neural generation models, particularly the pretrained language models (PLMs). Li et al. [32] summarise the key developments made in the field of PLMs for text generation. They provide the overall task definition as an initial step and briefly overview the common PLM architectures for text generation. The main topic of discussion is how to modify current PLMs to accommodate various input data

types and fulfil unique requirements in the produced text. They also provide an overview of several crucial text production fine-tuning techniques. They wrap up this study by outlining a few potential directions for the future. Their survey attempts to offer a summary and references to related studies for text generation researchers.

The following presents the studies and conclusions the survey's writers found.

- Model Extension.
  Despite several extensions being suggested, differences still persist between tasks related to pre-training and downstream generation. To further exacerbate the gap, the "[MASK]" token from the pre-training stage will not be used in the fine-tuning stage. It also wants to create a suitable pre-training paradigm for text production. Additionally, it has been demonstrated that adding outside knowledge to PLMs before training is beneficial [33], and exploring methods to incorporate relevant information for text generation is intriguing.

- Controllable Generation.
  Though it's still extremely early, controlled text production with PLMs is an intriguing direction. One of the many helpful applications of controlling various properties of the output text is to provide positive responses for patients suffering from depression in dialogue systems. Nevertheless, PLMs are typically pretrained in universal corpora, making it challenging to regulate the multi-grained characteristics of the text that is produced (such as sentiment, topic, and coherence). Text production using control codes that regulate style, content, and task-specific behavior has been studied by [34]. These control codes, however, are coarse-grained and preset. Future research can investigate multi-grained control and create suitably steerable PLMs.

- Model Compression.
  While large-scale parameter PLMs have proven successful in text production, their deployment in resource-constrained applications remains difficult. Therefore, researching ways to get competitive performance with less parameters makes sense. Parameter sharing [35] and knowledge distillation [36] are two techniques that have been proposed to compress PLMs; however, the majority of these techniques concentrate on BERT-based models, with little emphasis on compressing PLMs for text generation.

- Fine-tuning Exploration.
  Pre-training's primary goal is to apply the linguistic skills acquired in PLMs to downstream generation activities. Furthermore, the most common transfer technique used today is fine-tuning. Transferring knowledge from PLMs to downstream models could take several forms.

- Language-agnostic PLMs.
  Nowadays, English is the primary language used in almost all PLMs for text production. These PLMs will face difficulties while handling jobs involving non-English generating. Consequently, it is worthwhile to research language-agnostic PLMs, which must capture linguistic and semantic characteristics that are common to all languages. Reusing current English-based PLMs for text production in non-English languages is an intriguing direction.

- Ethical Concern.
  PLMs are currently pretrained on massive corpora retrieved from the internet without fine-grained filtering, which raises ethical concerns like creating user-specific private content. As a result, researchers ought to make every effort to stop PLM misuse. We can use [37] essential steps, including determining risks and possible effects and estimating likelihood. Furthermore, the text produced by PLMs may exhibit bias, consistent with the bias shown in training data related to gender, race, and religion. Therefore, to avoid these biases, we should intervene in PLMs. For PLMs, much study has been done, but it is still in its early stages.

2.3 From Standard Summarization to New Tasks and Beyond: Summarization with Manifold Information

In their paper, Gao et al. [2] focus on investigating new summarizing tasks and techniques in real-world applications where data are typically not in a plain text format. Gao et al. conducted a literature study on new summarizing tasks and the methodologies that go along with them in their survey paper. This paper introduces

eight new summarization tasks (listed in Figure 2.1): Stream document, Timeline document, Extreme long document, Dialog, Query-based document, Incorporating reader comment, Template based and Multi-media. The first five of these jobs fall under the category of summarization *incorporating document structure*. The final three assignments fall into the category of summarizing *incorporating additional knowledge*.

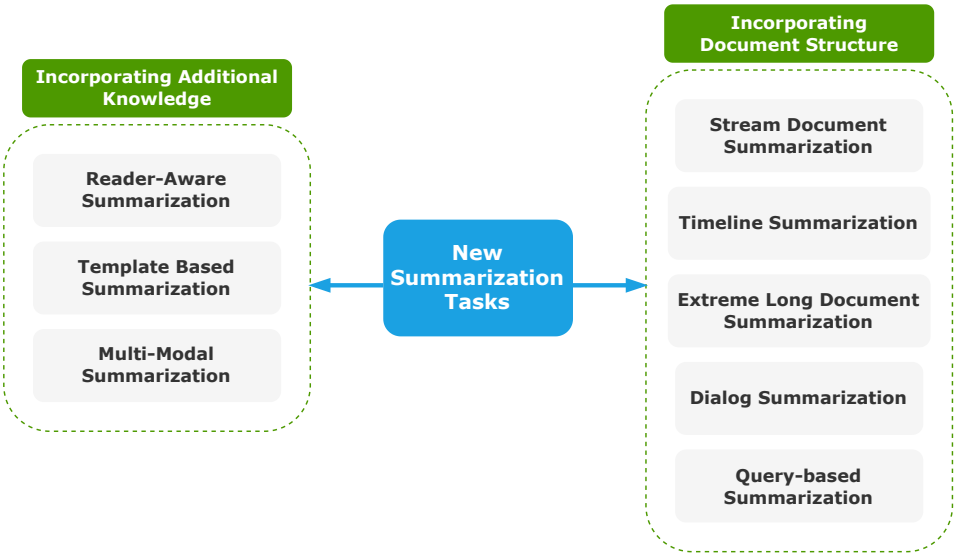


Figure 2.1: New summarization tasks introduced in the paper [2].

The following is a list of the key conclusions from the survey that was taken into account regarding summarization incorporating additional structure:

- Stream Document Summarization.
  When a new document is received in a text stream, the stream summary must be updated concurrently, considering any previously received information. The state-of-the-art techniques used for this task today are all derived from extractive and human-engineered techniques.

- Timeline Summarization.
  Timeline summarization is crucial for understanding the evolution of a topic. Previous research has used extractive methods, but Chen et al. [38] propose a key-value memory network-based architecture to store events in a timeline.

The network uses event time representation as the key and splits values into *global* and *local* slots. Finally, an RNN-based decoder is used to generate an abstract summary.

- Extreme Long Document Summarization.
  Even with current techniques, summarising an extremely long document is still a significant challenge. The ES and AS approaches are both used to tackle this difficult task.

- Dialog Summarization.
  In contrast to summarizing a paper, the important details are dispersed over the entire conversation history. For this job, Gliwa and et al. [39] offer the first large-scale dataset *SAMSum* to take advantage of the neural-based text-generating method. Making a summary of conference transcriptions is the meeting summarization task, a subtask of the dialogue summarization task. Thus, other visual cues, such as a participant's head position and eye contact, might be utilized to summarize a meeting.

- Query-based Summarization.
  Researchers are increasingly focusing on query-based summarization tasks to generate a summary that highlights relevant points in the context of a given query. Most methods are based on conventional machine learning methods, such as semi-supervised graph-based models, unsupervised multi-document query-based summarization, and sentence compression methods. To avoid generating repeated phrases and increase summary diversity, neural-based Seq2Seq frameworks are proposed. These frameworks ensure that context vectors in the attention mechanism are orthogonal to each other and use a modified long-short term memory (LSTM) cell to compute the new state at each decoding time step. The attention mechanism also focuses on different portions of the query at different time steps.

Below is an overview of incorporating additional knowledge summarization extrapolated from the paper:

- Reader-aware Summarization.
  News websites often allow readers to post comments on articles, which can

help summarize the main idea of the news. Two methods are introduced: conventional learning methods and neural networks. Gao et al. [40] propose a large-scale dataset and a neural generative method *RASG* for sentence extraction. The dataset contains 863,826 data samples, each with multiple documents, summaries, and reader comments. The RASG method is a generative-adversarial [41] learning method that captures reader attention distribution on the article and uses reader-focused article information to guide the summary generation process. This approach differs from previous methods that use sentence extraction on small-scale datasets.

- Template Based Summarization.
  The template-based summarizing approach retrieves a summary template and modifies it into a new one of the current material to provide a fluid textual result. The currently used approaches can be divided into two groups: *hard-editing* and *soft-editing*. To be more precise, hard-editing methods compel the machine to produce a summary using the same linguistic structure as the template. On the other hand, summaries produced using soft-editing techniques are more adaptable and can make use of partial words in the template. The latter are the most used nowadays.

- Multi-Modal Summarization.
  In the multi-modal summarization task, as opposed to the typical text summarization task setting, the visual information is integrated into the text summarizing process alongside the input material to enhance the quality of the output summary. The authors present existing multi-modal summarisation methods, such as using a Seq2Seq-based abstractive model for image-based summarization and an RNN-based decoder for summary generation. They also propose an image attention and context filter to avoid summarization noise. The video-based summarization uses a ResNeXt-101 3D [42] convolutional neural network to model video frames and fuse this information into Seq2Seq using a hierarchical attention mechanism.

2.4 A Survey on Cross-Lingual Summarization

The process of creating an overview in one language (e.g., English) for the provided document(s) in another language (e.g., Chinese) is known as *cross-lingual summarization* (XLS). In light of globalization, the computational linguistics community began paying more attention to this job. However, there is currently not a thorough review available for this task. As a result, Wang et al. [43] offer the first comprehensive critical analysis of the datasets, methodologies, and difficulties in this area. In particular, they meticulously arrange current datasets and methodologies according to various construction techniques and solution paradigms, respectively. They provide and summarize prior work for each type of dataset and methodology in detail and compare them with each other to provide more in-depth assessments. Finally, the authors also highlight promising directions and provide insights to help future studies.

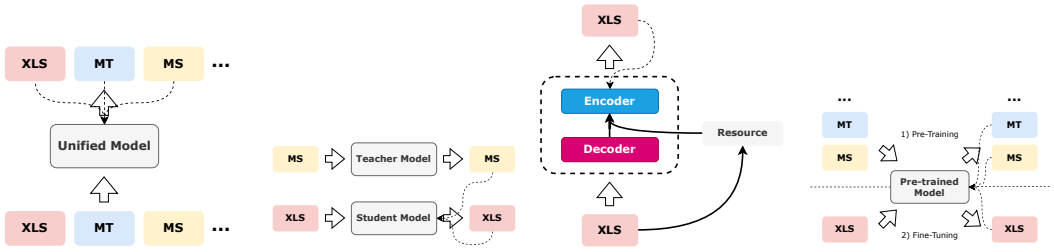
Wang et al. examine existing extensive XLS datasets and further classify them into two groups:

- Synthetic datasets.
  They are generated directly by translating the summaries of a monolingual summarization (MS) dataset from their original language to different target languages. The trade-off between quality and scale exists. Consistent with MS, because news articles are easy to gather, XLS datasets in the news domain have a substantially more significant scale than others. It is costly and often unfeasible to manually translate or post-edit every summary in such massive databases. Because of this, these datasets typically use automatic translation techniques, which results in poor quality. The discourse domain's XLS datasets present significant challenges. In addition to its restricted scope, a dialogue's essential information is frequently dispersed among several utterances, resulting in low information density [30], which combines with complicated dialogue phenomena.

- Multi-lingual website datasets.
  Multi-lingual online resources are expanding dramatically as a result of global-

ization. One explanation is the growing trend of websites offering multi-lingual versions of their material to cater to users worldwide. As a result, there may be a lot of parallel documents in several languages on these websites. Some academics attempt to create XLS datasets by utilizing these services. The authors discover that while working with low-resource linguistic scenarios, the automatic creation of this kind of datasets is unsuccessful.

The pipeline methods, whose basic notion is breaking down XLS into MS and MT sub-tasks and then completing them step by step, are typically the focus of early XLS works. Based on the final sequence of the subtasks, these approaches can be further separated into summarize-then-translate (*Sum-Trans*) and translate-then-summarize (*Trans-Sum*) categories.



(a) Multi-Task (b) Knowledge-Distillation (c) Resource-Enhanced (d) Pre-Training

Figure 2.2: Four end-to-end frameworks are summarized here; color viewing is recommended. Dashed arrows indicate the supervised signals. The associated tasks' input or output sequences are represented by colored blocks with no edges. It should be noted that the knowledge distillation framework may have many instructor models and that the pre-training activities utilized in the multi-task/pre-training framework are not exclusive to MT and MS; for the sake of simplicity, we have excluded them here.

Despite its ease of use, the pipeline technique has some drawbacks, including error propagation, latency during inference, the requirement for a sizable corpus to train MT models or the financial burden of paying for MT services. Many end-to-end XLS models have been developed to address the problems mentioned above, primarily due to the rapid growth of neural networks. Below is the evaluation of the prior end-to-end XLS models and further categorization into four frameworks

(Figure 2.2): *pre-training*, *resource-enhanced*, *knowledge-distillation*, and *multi-task* frameworks. Wang et al. will go through each framework's central concept and related models in detail.

Chapter 3

Method

We offer a survey of current LRS methodologies. We start by searching through every new LRS technique that the research community has put forth. After that, we have to filter them based on the paper's goal and the filters that have been considered. Second, we examined the articles that had been filtered in order to provide a precise definition of the phrase "*few-shot*", select a single evaluation metric, identify relevant datasets, and describe the authors' selection process for the training samples. Following the publishing analysis, we offer a formal taxonomy for LRS approaches, compile prior research based on various contexts, and arrange leaderboards using the previously selected unified metric.

3.1 Paper retrieval

Searching top ML conferences (IJCAI[1], EMNLP[2], ACL[3], NAACL[4], COLING[5], EACL[6]), Google Scholar[7] and Scopus[8] using relevant keywords, we discovered papers related to *low-resource/few-shot summarization*. After that, we filtered

[1] https://www.ijcai.org/
[2] https://2023.emnlp.org/
[3] https://aclanthology.org/
[4] https://naacl.org/
[5] https://coling2022.org/
[6] https://eacl.org/
[7] https://scholar.google.com/
[8] https://www.scopus.com/

the papers based on their relevance to the interest in LRS[9] and the availability of reproducibility results; for instance, papers such as [21] containing only human evaluations are excluded. The resulting LRS methods obtained had to be tested on commonly used datasets or be open-source to enable testing on the latter, according to the evaluation metric chosen. One useful tool utilized is the Scopus search results analyzer, which, in combination with the Scopus query research (3.1), permits the retrieval of some data, such as the number of publications made over the past few years that match the query (Figure 3.2).

```
TITLE-ABS-KEY ((low-resource OR few-shot OR (few AND
   shot)) AND abstractive AND summarization) AND
   (LIMIT-TO(SUBJAREA, "COMP"))
```

Figure 3.1: Scopus query used to retrieve LRS publications.

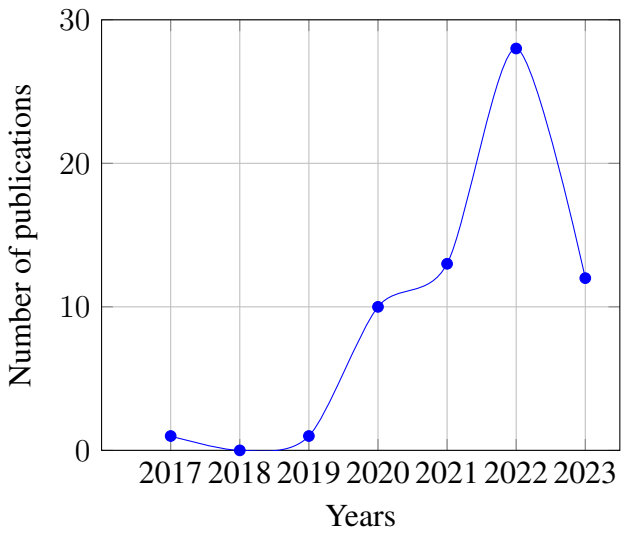


Figure 3.2: Plot of the frequency of LRS document publications over the years relative to the Scopus query (3.1).

[9] Only regarding training-data scarcity.

3.2 Training samples choice

The first choice that needs to be made before an ML model can begin to be trained is which training samples to use. Examining how the training samples for LRS are selected we discovered that out of 20 papers, 15 employ random selection from the dataset, two use similarity rankings ([15], [44]), only one uses human-annotated samples ([3]), one did not specify it ([17]), and the remaining two take the first *n* samples ([9], [10]). Through some experimentation, we discovered that, in this instance, employing random or first-*n* sampling did not affect final performance [10].

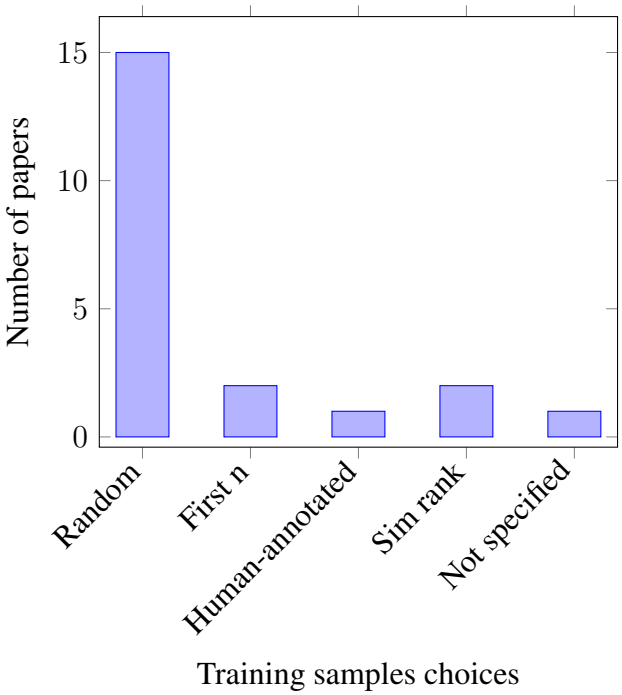


Figure 3.3: The histogram shows the frequency of occurrences for the various methods of training samples choice. "Sim rank" stands for similarity ranking.

3.3 "Few-shot" meaning

The term "few-shot" is used with different arbitrary meanings, and our objective is to offer a clear definition for it. In the analyzed papers, "few-shot" typically refers

to either 10-shot or 100-shot. Of 20 papers, 13 use 10-shot[10] and 14 use 100-shot, while the rest define "few-shot" as a value between 147 and 1000 samples. The analysis results can be seen in Figure 3.4; some publications refer to "few-shot" in multiple ways, like the 10-shot/100-shot case. In conclusion, we suggest that 10/100-shot be included in the official definition of "few-shot" since it appears in LRS studies far more frequently than the other concepts.

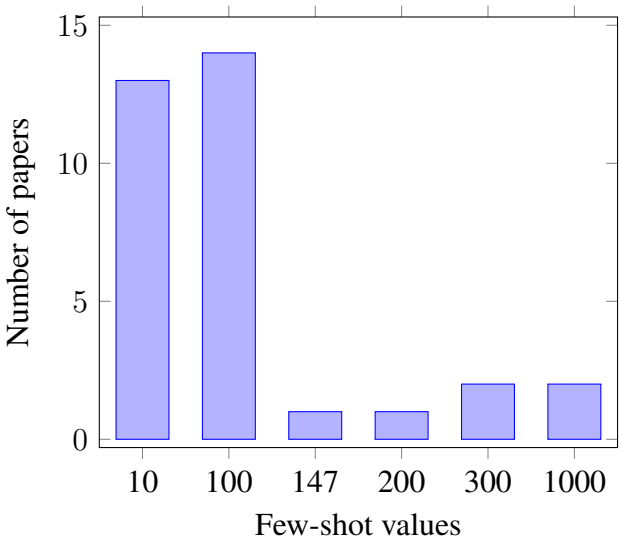


Figure 3.4: The term "few-shot" can have multiple meanings in the papers. The histogram shows the frequency of occurrences.

3.4 Evaluation metrics

When evaluating summarization tasks, the standard metric used is ROUGE [45]. This metric measures the word overlap (ROUGE-1), bi-gram overlap (ROUGE-2), and longest common sequence between the ground truth and the generated summary (ROUGE-L). The F1 scores are calculated for ROUGE-1, ROUGE-2, and ROUGE-L. Generated summaries evaluation is based only on ROUGE scores because they are the only metric used to evaluate every method.

Other metrics, such as BLEU [46], BERTScore [47], and METEOR [48], are not commonly used in analyzed papers and are therefore not considered. In the 20

[10] If a paper mentions 10-shot, it also refers to 100-shot, but not vice versa.

considered papers, BLEU was used in only three papers, BERTScore in only four, and METEOR in only one.

3.5 Datasets

To provide meaningful leaderboards, we selected the five most used dataset to evaluate the analyzed methods which are: *SAMSum* [39] (every dialogue summarization approach tested on it), *CNN/DailyMail* [49], *XSum* [50], *BillSum* [51] and *Multi-News* [52].

- The SAMSum dataset contains 16k messenger-like conversations written by fluent English linguists, reflecting their daily writings. Conversations can be informal, semi-formal, formal, or slang-filled and are annotated with summaries, providing a concise summary of the conversation in the third person.
- The Extreme Summarization (XSum) dataset evaluates abstractive single-document summarization systems by creating short, one-sentence summaries from BBC articles collected from 2010 to 2017. The dataset covers various domains.
- The CNN/DailyMail dataset is an English-language dataset with over 300k unique news articles, supporting both extractive and abstractive summarization, originally designed for machine reading and comprehension.
- BillSum is a dataset for summarizing US Congressional and California state bills. The dataset focuses on mid-length legislation from 5,000 to 20,000 characters, using characters instead of words or sentences due to their complex structure. Short bills don't need summaries and just make little adjustments, while long legislation often consists of large sections.
- Multi-News is a collection of news articles from `https://www.newser.com/` along with summaries of those articles written by humans. Editors write each synopsis professionally and include links to the original, cited articles.

3.6 Taxonomy

In this section, we propose a taxonomy for the LRS methods to organize them, helping future works in this field. All the techniques are for *abstractive* summarization; only one method focuses on *extractive* summarization. Figure 3.5 shows the sets and sub-sets of currently existing approaches.

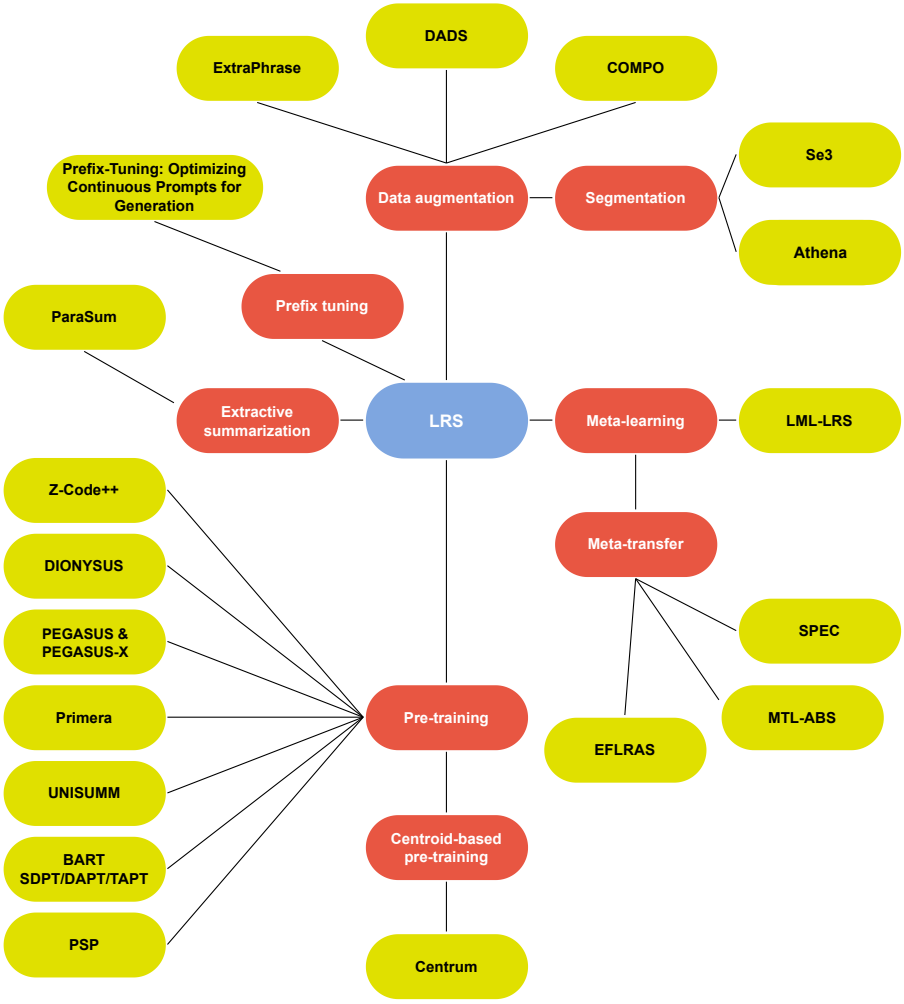


Figure 3.5: Currently existing LRS methods organized according to the proposed taxonomy.

3.6.1 Data Augmentation

Data augmentation (DA) in low-resource summarization scenarios involves the creation of additional training data by applying various techniques to the limited available data. The goal is to expand the dataset to improve the performance and generalization capabilities of summarization models when faced with data scarcity.

DA includes a wide range of techniques used individually or in combination; however, a subset of methods uses a specific type of DA called *segmentation*. The latter consists of splitting the original text into multiple coherent chunks to improve the amount of data.

Several techniques can be used to generate pseudo-training data for natural language processing tasks. Here the list of them: **ExtraPhrase** [8], **DADS** [3], **COMPO** [53], **Se3** [9] and **Athena** [10]. It's worth noting that the last two techniques are based on segmentation, unlike the others. Each technique is covered in detail in section 3.7.

3.6.2 Extractive summarization

ES is a text summarization technique that entails selecting and extracting key words or phrases from the source material to create a concise summary. Instead of generating new sentences, ES identifies the most informative and relevant content from the source text, typically based on criteria like sentence importance, coherence, or saliency. This approach is commonly used in applications where the goal is to preserve the original wording and structure of the text while condensing it for brevity. There is only one LRS method that utilizes ES, known as **Parasum** [17], detailed in section 3.7.19.

3.6.3 Prefix tuning

The standard method for utilizing huge PLMs for activities that come after is fine-tuning. However, fine-tuning requires keeping a complete copy for every task and changing every language model parameter. For natural language generation tasks, prefix-tuning is a lightweight alternative to fine-tuning. It maintains language model parameters frozen and instead optimizes a series of continuous task-specific

vectors known as the *prefix*.

Prefix-tuning allows future tokens to attend to this prefix as if they were "virtual tokens," taking inspiration from prompting for language models. It is demonstrated in [54] that prefix-tuning achieves comparable performance in the full data setting, outperforms fine-tuning in low-data settings, and more accurately extrapolates to scenarios involving patients not covered in training by changing only 0.1% of the parameters.

3.6.4 Meta-learning

MTL, also known as "learning to learn", is a technique in ML that focuses on improving the learning process. It involves training models to know new tasks better or quickly adapt to new data. With MTL, a model learns from various tasks or datasets and applies that knowledge to perform well on new, unseen ones. This approach aims to create more versatile models that can adapt swiftly to different tasks or domains, making them valuable in situations with limited training data or the need for rapid learning. The solution proposed is a method called **Lightweight Meta-Learning for Low-Resource Abstractive Summarization** (LML-LRS) [14].

Meta-transfer learning is an extension of MTL that focuses on transferring knowledge and adapting models across different tasks, domains, or environments. This approach trains models to quickly learn from different tasks and transfer knowledge to new ones, even across domains. Meta-transfer learning is a useful technique for improving performance in new and different contexts, especially when the source and target tasks or domains are significantly different, allowing models to leverage prior knowledge for improved performance in new and diverse contexts. Below are various techniques for addressing the LRS issue, including the **MTL-ABS** [15], **SPEC** [44] and **Efficient Framework for Low-Resource Abstractive Summarization by Meta-Transfer Learning and Pointer-Generator Networks** (EFLRAS) [16] approaches. All MTL solutions are described in section 3.7.

3.6.5 Pre-training

Pre-training in the context of ML refers to the first stage of training a model on a large dataset or a pre-existing model before fine-tuning it for a specific task.

During pre-training, the model learns general language or domain knowledge, such as grammar, vocabulary, and world facts, which can be a foundation for various downstream tasks.

The methods under analysis implement various pre-training techniques, often exclusive to the method itself, but also other approaches, like fine-tuning or prefix tuning, to maximize their effectiveness in the objective task.

Below are the proposed methods: **Z-Code++** [6], **DIONYSUS** [4], **PEGASUS** [11] and **PEGASUS-X** [55], **PRIMERA** [12], **UNISUMM** [7], **PSP** [13] and **Centrum** [56]. In [5], it is looked at how to adapt abstractive summarization models (in this case BART) to six different target domains, including dialogues, in a low-resource environment. The study focused on the second phase of pre-training, using large-scale generative models in three settings: *source domain pre-training* (**SDPT**), *domain-adaptive pre-training* (**DAPT**), and *task-adaptive pre-training* (**TAPT**). Also, these approaches are shown in detail in section 3.7.

3.7 Methods overview

Complete explanations of the methods that are previously categorized under the suggested taxonomy are provided in this section. The steps taken to get the end result are described for each technique.

3.7.1 DADS

By substituting text parts from the input dialogue and summary, *Data Augmentation for Low-Resource Dialogue Summarization* (DADS) [3] creates synthetic examples while maintaining the augmented summary to match a workable summary for the augmented dialogue. DADS uses pre-trained language models to generate highly plausible dialogue alternatives yet allow for the generation of various alternatives.

The data augmentation process is divided into the following three phases:

1. Utterances-to-summary alignment.
   First, matching input utterances with summary spans. By dividing each sentence into clauses using the commercial NLP pipeline annotator spaCy [57], the granularity of augmentations to the sub-sentence level increased. Next, the universal sentence encoder [58] is used to encode the collection of all summary clauses and dialogue utterances into a shared space, and then the cosine similarity is calculated. Liu et al. choose the top 20% of utterances for each summary clause that have the highest similarity scores as input pairs for augmentation. One augmented example will be produced by one (utterances, clause) pair.

2. Dialogue utterance replacement.
   Inspired by Meena [59] and DialogGPT [60], an auto-regressive encoder-decoder model that is initialized from T5-11B [61] and fine-tuned using a dialog reconstruction loss is employed. An utterance from an input example is randomly masked to train the model. For fine-tuning, the authors leverage the conversational dataset (SocialMedia), a high-quality, large-scale dialogue dataset suggested by Meena [59]. This refined model is known as DIAL-REPL. For the chosen utterances, they create synthetic alternatives using

DIAL-REPL. As illustrated in step 2 of Figure 3.6, DIAL-REPL is requested to predict the masked utterance given the input dialogue, the summary, and a prompt. Given the original dialogue, the appropriate point of the selected utterance is substituted by a [MASK] token. The standard prompt they used was, "The following conversation is about:" followed by the discussion and the summary. One by one, in an auto-regressive fashion, all of the chosen utterances are substituted; utterances that have already been generated are incorporated into the input for the subsequent masked position.

3. Summary FillUp.
   The paired clause in the summary is finally changed to a new one consistent with the expanded discourse. The authors' technique aims to achieve two goals: it will produce a more varied set of summaries and rectify semantic deviations expected to occur during dialogue utterance substitution, hence avoiding downstream summarization models to memorize repeating targets. For this specific challenge, they optimized a large pretrained PEGASUS [11] model to predict a masked sentence in summary given the input and the summary as context. The CNN/DailyMail [49] dataset was transformed into training data for this model by masking a sentence in the gold summary, preceding the masked summary with the input document, separating them with a separator token, and assigning the model the task of predicting the masked sentence — a process that is similar to the Gap Sentence Generation [11] procedure. In order to perform summary augmentation, we conceal the summary clause, prepend the enhanced conversation as input, and utilize the Summary FillUp model to forecast a new replacement clause. Liu et al. remove duplicate outputs, augment each annotated dialogue-summary (d, s) pair multiple times, and retain the remaining pairs as augmented instances.

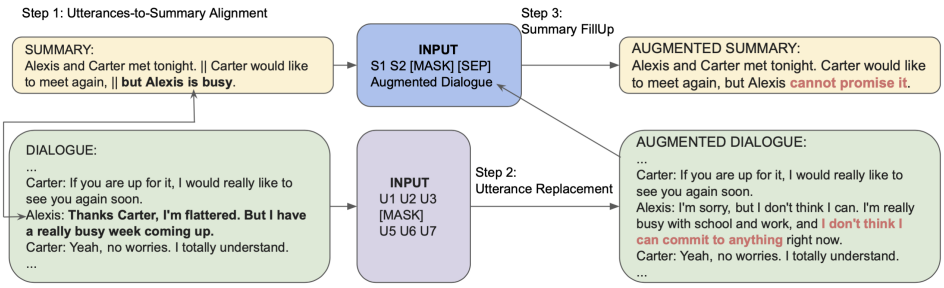


Figure 3.6: Example of DADS data augmentation process. (Source: [3]).

3.7.2 COMPO

COMPOsitional Data Augmentation for Abstractive Conversation Summarization first extracts conversation structures like topic splits and action triples as basic units. Then, it organizes these semantically meaningful conversation snippets compositionally to create new training instances. Additionally, it is important to explore noise-tolerant settings in both *self-training* and *joint-training* paradigms to make the most of the augmented samples.

COMPO is applied to the Bart [29] pre-trained model and consists of the following phases:

1. Topical Split.
2. Action Extraction.
3. Action-based Composition.

A more complete and detailed description can be consulted in paper [53].

3.7.3 DIONYSUS

Dynamic input optimization in pre-training for dialogue summarization (DIONYSUS) is a pre-trained encoder-decoder model for summarizing dialogues in any new domain, especially in 0-shot scenarios. To pre-train DIONYSUS (a diagram in Figure 3.7), two pseudo summaries for each dialogue example are created: one from a fine-tuned summarization model and the other from meaningful dialogue turns. Then, one of these pseudo summaries is chosen based on information distribution

differences in different types of dialogues. This selected pseudo summary is the objective for pre-training DIONYSUS using a self-supervised approach on a large dialogue corpus. The full implementation is described in [4].

Li et al. consider three ways to produce the final pseudo summary S for every individual dialogue training example. These tactics are predicated on the extracted "Principal" P and the generated pseudo summary G. The three DIONYSUS pre-training objective strategies are:

- All G.
  S = G: The pre-training objective is always the summary that the helper generates.

- All P.
  S = P: The "Principal" is always chosen as the pre-training goal.

- Better ROUGE.
  The pre-training target is determined by the authors using either G or P depending on the recollection of material from the discourse. Using Algorithm 1, the pre-training objective is obtained by computing the ROUGE1-F1 score for the dialogue and pseudo-summaries.

Li et al. incorporated a *copy mechanism* to enhance DIONYSUS performance. This mechanism is noteworthy because it allows the conversation to be summarized over numerous turns, which is helpful in conversations like meetings and medical discussions.

Algorithm 1 DIONYSUS Better ROUGE

$S \leftarrow \varnothing$
sg := $rouge(G, D\ P)$
sp := $rouge(P, D\ P)$
if sg > sp **then**
    S := G
else
    S := P
end if

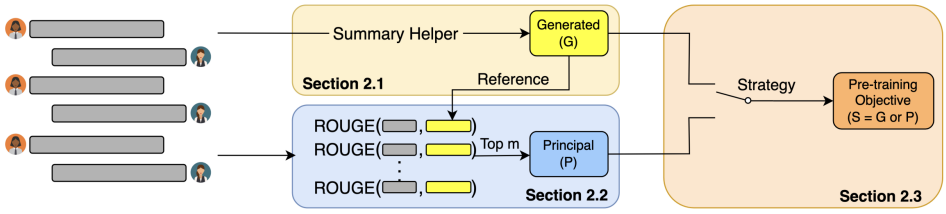


Figure 3.7: A DIONYSUS pre-training diagram. In order to choose dialogue turns as the "Principal" (P), the summary assistant creates a pseudo-summary (G). Then, using a variety of techniques, it selects between the created summary and the principal as the pre-training goal. (Source: [4]).

3.7.4 BART SDPT/DAPT/TAPT

The paper [5] presents a study of domain adaptation for the AS task across six diverse target domains, including dialogues, in a low-resource setting. Specifically, the investigation focuses on the second phase of pre-training on large-scale generative models (in the paper - BART [29]) under three different settings: *source domain pre-training* (SDPT), *domain-adaptive pre-training* (DAPT) and *task-adaptive pre-training* (TAPT). Research indicates a correlation between the pre-training data's similarity to the target domain task and the pre-training efficacy. Furthermore, it has been discovered that if pre-training is continued, the model may have *catastrophic forgetting*; however, this problem can be mitigated by using a learning strategy that causes less forgetting. Additionally, the results show that there is still a significant difference between the low- and high-resource situations, emphasizing the necessity for more sophisticated domain adaptation techniques for the AS task.

The description of the three approaches is below:

- SDPT.
  To facilitate quick adaptation in target domains, the study employs training samples from a News domain as a source domain because of its richness. In order to infuse task information into the pre-trained language model and enable it to swiftly adapt to the same job in target domains, despite low domain similarity, the method pre-trains BART using the summary data from the source domain.

- DAPT.
  We continue pre-training BART using its initial pre-training goal function (corrupting texts and then optimizing a reconstruction loss — the cross-entropy between the original document and the decoder's output) using an unlabeled domain-related corpus. The idea behind this approach is to quickly adapt the pre-trained language model to the target domains by incorporating domain information.

- TAPT.
  There are two possible problems with the size of the domain-related corpus for DAPT, which is often vast. First, a corpus this size could not always be accessible, particularly for domains with limited resources. Secondly, pre-training on a corpus this size takes much time and processing power. Pretraining on a smaller unlabeled corpus is, therefore, a valuable and useful research direction. TAPT stands for pre-training on a subset of the unlabeled texts in the summarization task of the target domain. As it uses the input documents from the summarizing job directly, TAPT uses a significantly smaller but far more task-relevant pre-training corpus than DAPT. In this configuration, TAPT runs significantly more cheaply and is not dependent on gathering a sizable corpus of data related to the topic.

As cited before, there is the significant problem of catastrophic forgetting that the authors overcome through RecAdam [62]; an example is shown in Figure 3.8.

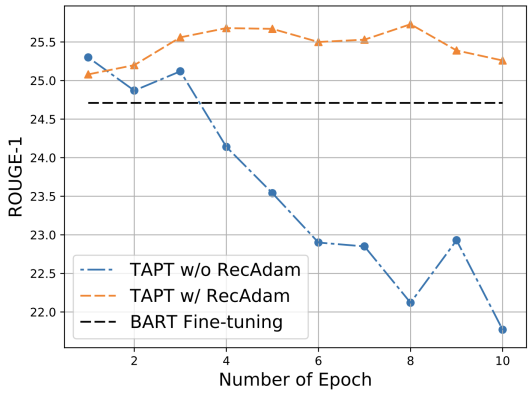


Figure 3.8: TAPT performance in the email domain when employing and not RecAdam across various pre-training epoch numbers. (Source: [5]).

3.7.5 Z-Code++

The paper [6] presents Z-Code++, a new pretrained language model optimized for abstractive text summarization. The model extends the state-of-the-art encoder-decoder model using three techniques. First, a *two-phase pre-training* (Figure 3.9) process improves the model's performance on low-resource summarization tasks. This step consists of pre-train Z-Code++ using two language modeling tasks, *replaced token detection* (RTD) [63] and *corrupted span prediction* (CSP) [64] [65]. In synthesis, the model is first pre-trained using text corpora for language understanding and then is continually pre-trained on summarization corpora for grounded text generation.

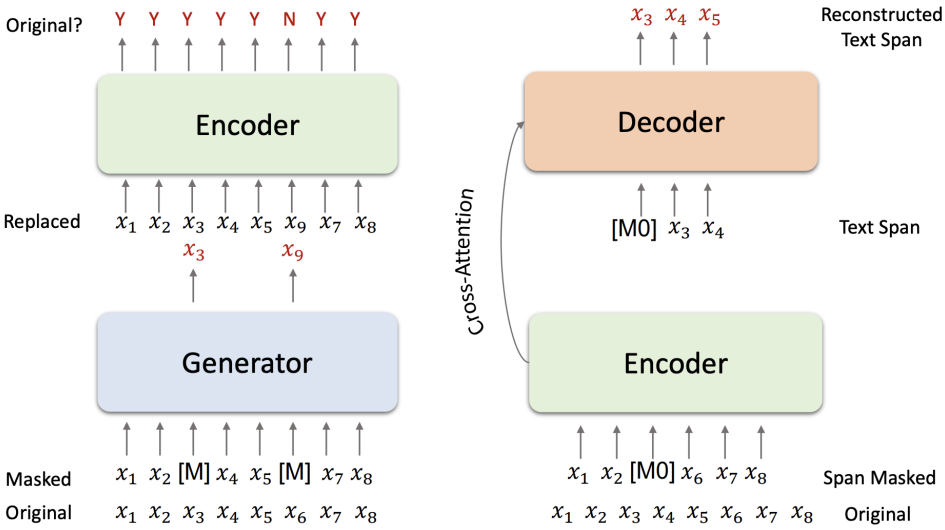


Figure 3.9: Token detection (RTD) and corrupted span prediction (CSP), which were employed in Z-Code++'s language model pre-training phase, were substituted by the two pre-training tasks. The encoder must be optimized using RTD, and the encoder-decoder must be optimized by CSP. During training, encoders with the same color share parameters. (Source: [6]).

Second, self-attention layers in the encoder are replaced with *disentangled attention* layers, where each word is represented using two vectors that encode its content and position, respectively. Third, a fusion-in-encoder (FiE), a simple yet effective method of hierarchical *long sequence encoding*, is used. Thanks to the FiE, the complexity of the encoder is reduced from $O(LN^2)$ to $O(mNl + nN^2)$.

3.7.6 UNISUMM

The current training paradigm for few-shot summarizing algorithms ignores potentially shareable knowledge in heterogeneous datasets despite the proliferation of summarization tasks and datasets. Chen et al. [7] present UNISUMM, a unified few-shot summarization model that can be prefix-tuned to perform well on any few-shot summarization task. The model has been pre-trained with numerous summarizing tasks. The process involves doing multiple steps: *multi-task pre-training with Prefix* (Figure 3.10 (a)), *prefix-tuning* (Figure 3.10 (b)), *universal prefix* and *asymmetrical weight decay*.

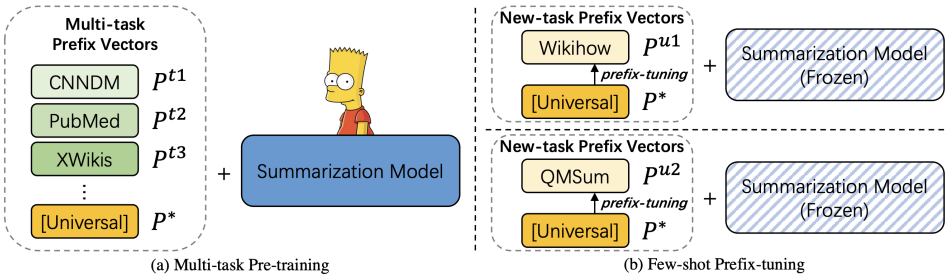


Figure 3.10: The UNISUMM two-phase framework. (a) The pre-training multi-tasking stage. The parameters and prefixes of the summarization model are jointly optimized on several pre-training tasks, such as CNN/DailyMail, PubMed, etc. (b) The phase of few-shot tuning. Adjusting the prefix settings while maintaining the same parameters for the summarization model for a new assignment, such as WikiHow. (Source: [7]).

3.7.7 ExtraPhrase

ExtraPhrase is a low-cost and effective strategy to augment training data for abstractive summarization tasks presented in [8]. This method constructs pseudo training data in two steps: *ES* and *paraphrasing*. The ES step extracts significant portions of an input text, and the paraphrasing step generates a variety of expressions. This is achieved by translating a sentence into a target language and back again into the original language, a process known as *round-trip translation*. Additionally, ExtraPhrase works better than current techniques like self-training and back-translation. Furthermore, ExtraPhrase is less expensive than the current methods.

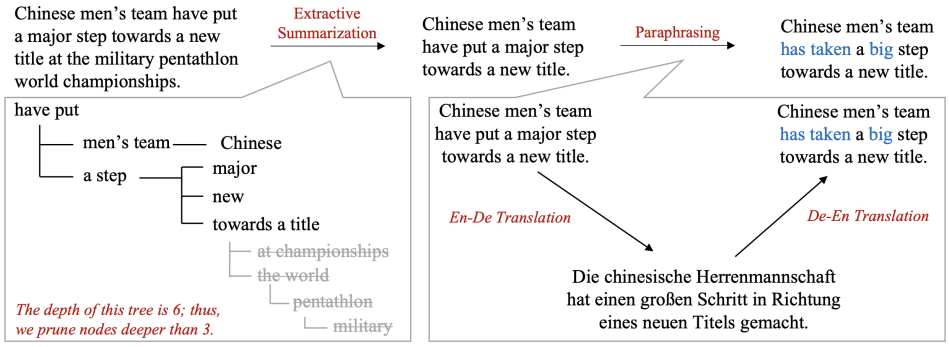


Figure 3.11: An instance of a pseudo-summary produced with ExtraPhrase. The output sentences for each step are displayed in the upper section. Blue highlights are applied to paraphrased words following round-trip translation in step 2. (Source: [8]).

3.7.8 Se3

In low computing resource circumstances, long-document summarization (LDS) is not possible due to transformers' quadratic memory complexity. The use of input truncation by state-of-the-art models results in the removal and disregard of potentially summary-relevant elements, which negatively impacts performance. Moreover, this loss generally has detrimental effects for semantic text analytics in high-impact sectors like law. A novel semantic self-segmentation (Se3) approach for LDS is proposed in the paper [9] to address the critical issues of low-resource regimes: processing inputs longer than the GPU memory capacity and producing accurate summaries even when there are only a few dozen training instances available. By summarizing each chunk and concatenating the results, Se3 divides a lengthy input into semantically coherent chunks, enabling transformers to summarize very large documents without truncating them.

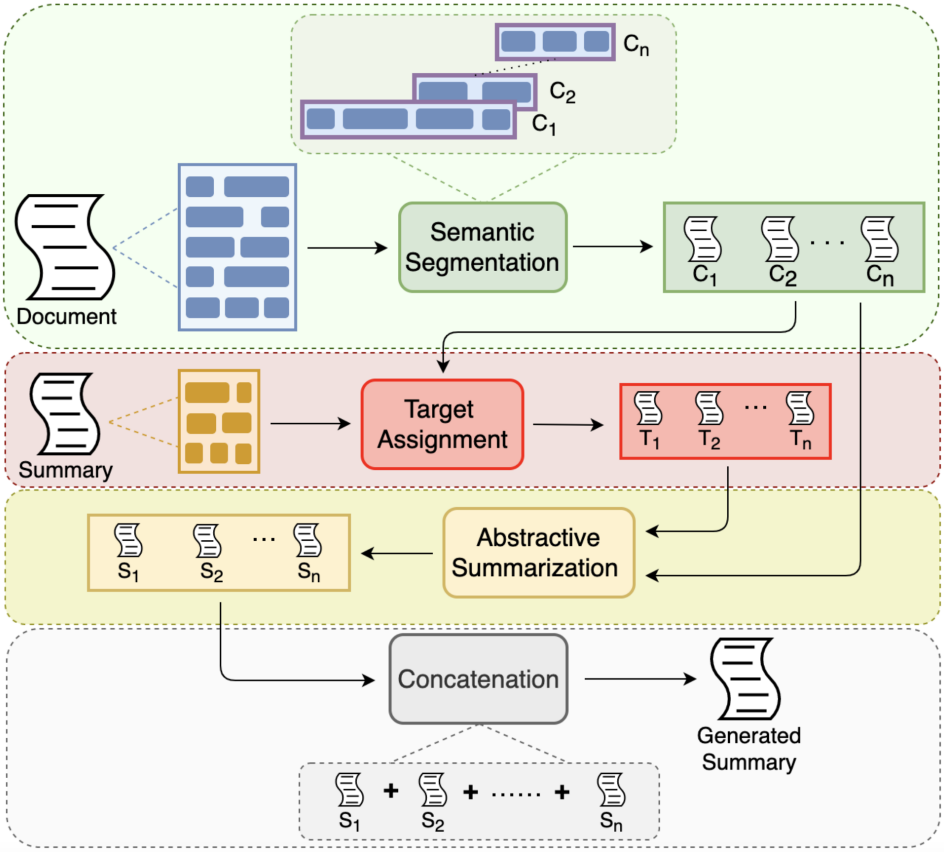


Figure 3.12: An overview of Se3 for AS of an extensive input. Initially, a document with many sentences, represented by *blue* rectangles, is divided into content-related sections (*green* phase). Following that, the most similar chunk is allocated to each summary sentence (*orange* rectangles), resulting in new high-correlated source-target pairs (*red* phase) that are utilized to train summarization models (*yellow* phase). Concatenating the chunk summaries yields the final summary at inference time (*gray* phase). (Source: [9]).

3.7.9 Athena

In text summarization, generative transformer-based models have demonstrated state-of-the-art performance. However, they continue to have difficulties with lengthy documents in LRS scenarios in real-world situations. In order to close the gap, the study [10] addresses two important research issues related to summarizing long documents: *long-input processing* and *document representation learning*, all of which are

combined into a single coherent model that has been trained for LRS. More specifically, by optimizing the alignment of chunk-target pairs in the text segmentation output, the innovative align-then-abstract representation learning approach (Athena) jointly trains a *summarizer* and a *segmenter*. Figure 3.13 provides an overview of Athena's architecture.

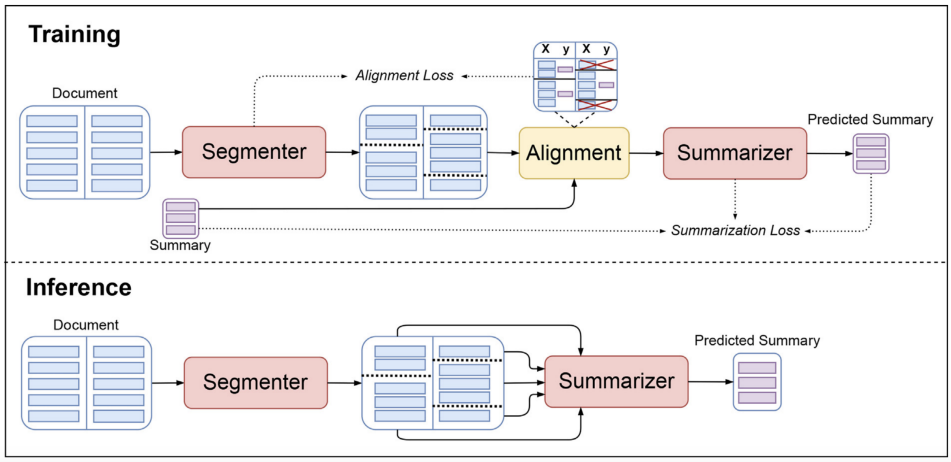


Figure 3.13: ATHENA's training and inference time depiction. A concise summary is the result, whereas a lengthy document is the input. (Source: [10]).

The tasks Athena does during the summarization process are briefly described below:

- Long-input processing.
  The model uses the Se3 algorithm [9], an unsupervised method, but introduces a novel loss to train the segmenter end-to-end to improve summarization accuracy. The model also aligns each sentence to the chunk that can better summarize it, yielding new high-correlated instances.

- Document representation learning.
  The method is trained to maximize the conditional probability of generating $\overline{Y_i}$ from $\overline{X_i}$, where $(\overline{X}, \overline{Y})$ is the set of chunk-target pairs produced by the segmentation and alignment modules. The segmenter is trained to maximize the alignment between each chunk-target pair in terms of semantic content coverage, encouraging the model to locate the best text segmentation that

improves the summarization. The summarization module is taught to produce the next output token for each target by reducing the negative log-likelihood after receiving input from the chunk-target pairings. The weights of the model updated during this learning are the same used for segmenting document sentences into chunks.

To sum up, the model is trained using an align-then-abstract approach to segment and summarize content-wise chunks. The segmenter is optimized with alignment loss, while the summarizer is optimized with summarization loss. An update step is involved with dynamic mini-batch gradient descent, where the gradients are computed for each document chunk, averaged per document, and descended after each instance. The summarizer reads more labeled samples, and the final summary is obtained by concatenating predicted chunk-level summaries.

3.7.10 PEGASUS

In the work [11], it is proposed pre-training large Transformer-based encoder-decoder models on massive text corpora with a new self-supervised objective. In PEGASUS, important sentences are removed/masked from an input document and they are generated together as one output sequence from the remaining sentences, similar to an extractive summary, achieving interesting performance. Additionally, PEGASUS has a *large* version that performs better; the details on the implementation can be found in Zhang et al. publication [11].

PEGASUS use two different pre-training objectives (example in Figure 3.14), in isolation and in conjunction. The first one is *Masked Language Model* (MLM), the study uses BERT to select 15% tokens in input text, replacing 80% with [MASK2], 10% with random tokens, and 10% are left unchanged. MLM is applied to train Transformer encoder as the sole pre-training objective or along with GSG. However, MLM does not improve downstream tasks at large pre-training steps, so it is not included in the final model PEGASUS$_{large}$. The second one is *Gap Sentences Generation* (GSG), the authors propose a new pre-training objective for AS that closely resembles the downstream task. The objective involves generating summary-like text from an input document, using a sequence-to-sequence self-supervised objective. The model is inspired by masking words and contiguous spans, selecting and

masking, with [MASK1], whole sentences from documents, and concatenating the gap-sentences into a pseudo-summary. The *gap sentences ratio* (GSR), similar to *mask rate* in other works, is calculated to inform the model. The objective also includes selecting sentences that appear significant/principal to the document. Then three primary strategies for selecting *m* gap sentences are considered: *random*, *lead*, and *principal*.
Random: Uniformly select *m* sentences at random.
Lead: Select the first *m* sentences.
Pricipal: Choose the top m statements based on their relative relevance. ROUGE1-F1 score is calculated between the sentence and the remainder of the document as a proxy for importance.

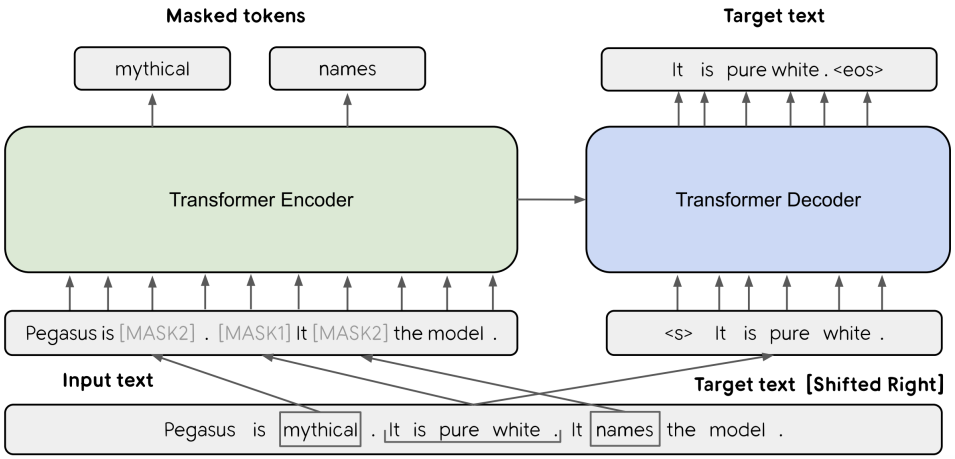


Figure 3.14: PEGASUS's basic design is a Transformer encoder-decoder. This example uses both GSG and MLM concurrently as pre-training goals. Three sentences are present at first. [MASK1] masks a single sentence that is utilized as target generation text (GSG). The other two phrases are still there in the input, but [MASK2] (MLM) randomly masks some tokens. (Source: [11]).

After the pre-training on C4 [66] or HugeNews datasets, PEGASUS is fine-tuned on downstream tasks to achieve better performance on them.

3.7.11 PEGASUS-X

The paper [55] introduces PEGASUS-X, an extension of the PEGASUS model with additional long input pre-training to handle inputs of up to 16K tokens. PEGASUS-X achieves strong performance on long input summarization tasks comparable with much larger models while adding few additional parameters and not requiring model parallelism to train. As a general rule, Phang et al. pre-train PEGASUS$_{base}$-sized models according to the [11] recipe. Unless otherwise noted, every trial in this ablation study used C4 for 500k steps of pretraining with 512 input tokens, 256 output tokens, and a 45% masking ratio. In order to accommodate the 8x increase in input sequence length, the authors reduce the masking ratio by a factor of 8 and extend the input length to 4096 tokens for lengthy input pre-training. They also modify the masking ratio from 45% to 5.625% and limit their search to documents that are longer than 10,000 characters.

3.7.12 PRIMERA

With an emphasis on summarization, PRIMERA is a pre-trained model for multi-document representation that minimizes the need for dataset-specific architectures and a significant quantity of fine-tuning labeled data. A recently developed pre-training objective is used by PRIMERA (structure depicted in Figure 3.15) to educate the model to connect and aggregate information across documents. It also streamlines the processing of concatenated input documents by utilizing effective encoder-decoder transformers. Complete information can be found in [12].

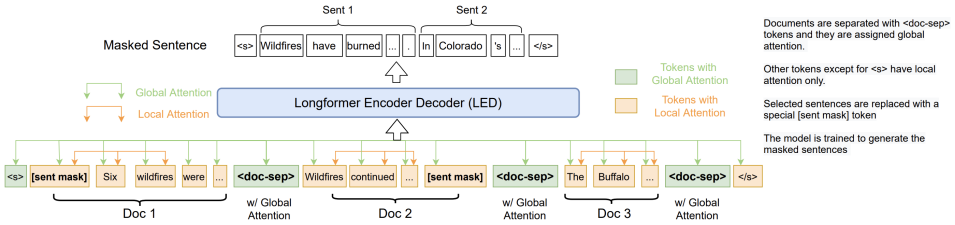


Figure 3.15: PRIMERA's model structure. (Source: [12]).

Xiao et al. employ the GSG objective, following PEGASUS [11], but they also propose a method for identifying salient sentences for masking in multi-document

summarization, called Entity Pyramid Masking (EPM), based on the Pyramid Evaluation's concept [67] of content saliency. The more documents an entity appears in, the more salient the information should be. The method uses entity frequency as a proxy for saliency, addressing the limitations of human-annotated *Summary Content Units* (SCUs). PRIMERA implements the next three procedures (Figure 3.16) to choose key phrases for EPM: *entity extraction* (i), *entity pyramid estimation* (ii) and *sentence selection* (iii).

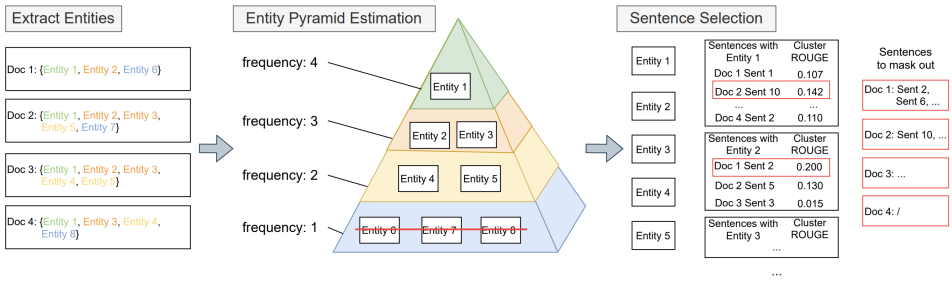


Figure 3.16: Using the Entity Pyramid Strategy, choose words that are important to disguise. The frequency of entities in the papers is the basis of the pyramid entity. For every entity with frequency > 1, the most representative sentences are selected using Cluster ROUGE, for example. Sentence 10 for Entity 1 in Document 2. (Source: [12]).

3.7.13 PSP

Effective and tuning only very light parameters, PSP [13] is a novel soft prompts architecture combined with a prompt pre-training plus prompt fine-tuning paradigm (a comparison with earlier approaches is presented in Figure 3.17). The soft prompts are composed of continuous input embeddings across an encoder and a decoder in order to comply with the structure of the generating models. Interestingly, a new inner prompt to record document-level data is included into the text. The goal is to concentrate on comprehending the document so that the model can produce document-related information more effectively. The model is first taught the fundamentals of summarizing by the prompt pre-training with self-supervised pseudo-data during the training phase. Then, only the lightweight, planned prompts are refined using few-shot samples.

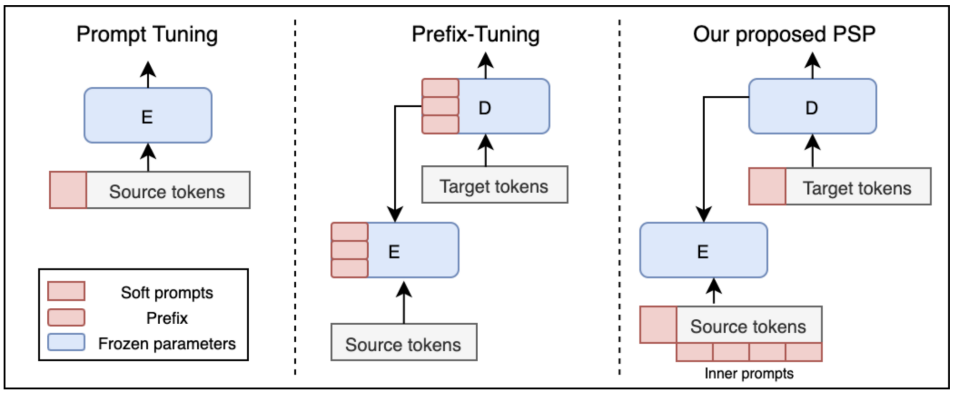


Figure 3.17: The comparison between earlier techniques and PSP. The encoder and decoder are denoted by the letters "E" and "D," respectively. (Source: [13]).

Following the descriptions of the various types of prompts present in PSP:

- Encoder-Decoder Basic Prompts.
  In the training phase of current architectures, P_{en} extracts knowledge from the encoder's frozen language model, while P_{de} copies P_{en}'s behaviour and guides the model to generate fluent summary text.

- Inner-Prompts for Document Understanding.
  The authors propose adding inner prompts to the source document, corresponding to sentences, to enhance understanding of the document's discourse. These prompts are added to the token embedding. Three strategies are proposed for incorporating different inner prompts, enhancing the model's ability to quickly interpret the document by strengthening associations between outputs and documents: *interval*, *sequential* and *fixed-length*.

- Self-supervised Prompt Pre-training.
  Soft prompts are pretrained using summarization-oriented self-supervised objectives to improve their understanding of documents and adapt to summarization tasks. Two strategies for constructing self-supervised data were tested: *lead* and *GSG*, each designed to suit a specific writing bias in the document.

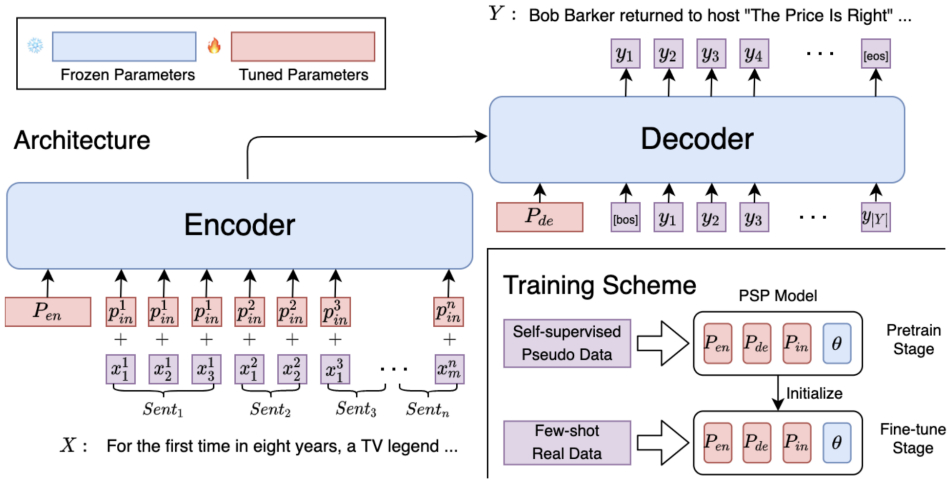


Figure 3.18: PSP's architecture and training program. Red and blue squares, respectively, represent tweaked and frozen settings. (Source: [13]).

The paper [13] displays all of the PSP information and outcomes. Figure 3.18 illustrates the overall architecture and training scheme.

3.7.14 Centrum

In Multi-Document Summarization (MDS), the input can be modelled as a set of documents, and the output is its summary. In the paper [56], the focus is on pre-training objectives for MDS. Specifically, a novel pre-training objective is introduced, which involves selecting the ROUGE-based centroid[11] of each document cluster as a proxy for its summary. To reach his objective, Centrum thus does not require human written summaries. It can be utilized for pre-training on a dataset consisting solely of document sets (containing at least three documents each). Additionally, it handles possible noise, such as a document mistakenly linked to a cluster, that may be present in datasets of multi-document clusters that are automatically generated [69]. Such noise is eliminated by the Centrum pre-training objective since a mismatched document would not be selected as the cluster centroid. Puduppully et al. approach uses [70] concept for MDS task-specific pre-training, which sets it apart from Vogler

[11] As an alternative to other cluster quality metrics, centroid distance is calculated as the average distance between each cluster item and the cluster centroid, which represents the "average object" or average point in space for the cluster. [68].

et al.'s. Furthermore, the authors use the LED [71] design to handle the extended document context in the input, in accordance with Xiao et al. [12].

3.7.15 Lightweight Meta-Learning for Low-Resource Abstractive Summarization

Lightweight Meta-Learning for Low-Resource Abstractive Summarization (LML-LRS) involves incorporating a lightweight module (Figure 3.19) into the attention mechanism of a pre-existing language model. This module is initially trained with high-resource task-related datasets through MTL and then refined with the low-resource target dataset.

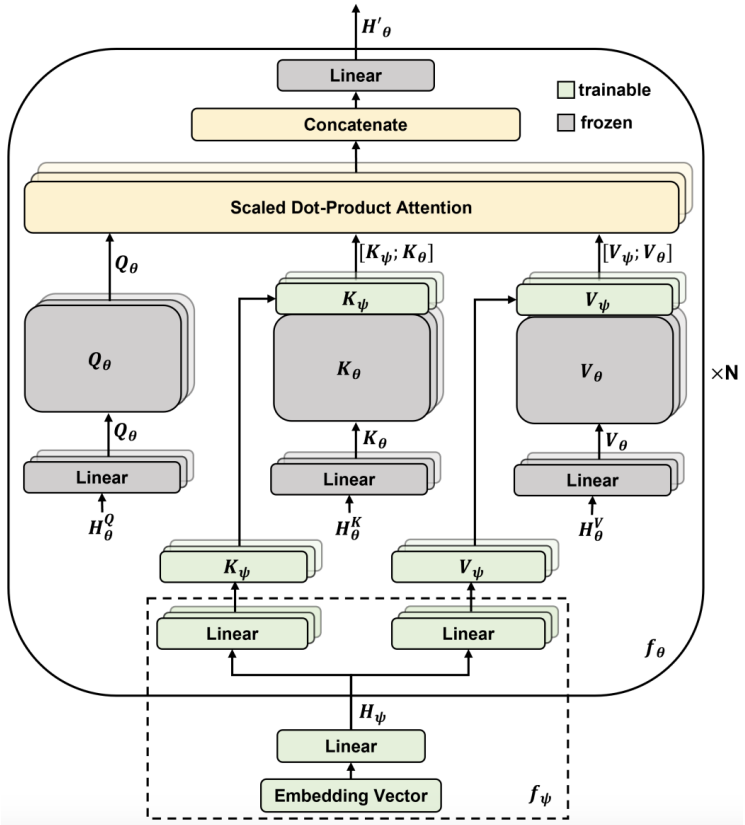


Figure 3.19: An overview of the lightweight module's attention mechanism. The dotted line box indicates the lightweight module. Merely, the module can be trained during the meta-learning and fine-tuning process. (Source: [14]).

The training framework involves meta-learning a lightweight module with high-resource datasets and fine-tuning it on a low-resource dataset. The Model-Agnostic Meta-Learning (MAML)[finn2017modelagnostic] algorithm initialises the model parameters for quick adaptation to new domains with small training data. The pre-trained language model is frozen, and only the lightweight module is meta-learn. Existing high-resource datasets are used in the meta-learning step, except for one target dataset. Three datasets similar to the low-resource target dataset are selected using document similarity. The meta-learning consists of two optimization loops: *inner loop* (IL) and *outer loop* (OL). The datasets are divided into domain-specific datasets for the IL and domain-specific datasets for the OL. The module learns domain-specific knowledge through the IL, while the parameters are initialized to adapt to various domains using losses of domain-adapted modules. The parameters are then repeated in both loops to quickly adapt to the target domain. Then, the module is fine-tuned with a low-resource target dataset, setting the number of training target data to 10 and 100 and optimizing to minimize negative log-likelihood.

3.7.16 MTL-ABS

Large pre-trained models and a variety of existing corpora are two knowledge-rich sources that the paper [15] suggests using to address the LRS challenge. The former can offer the fundamental skills needed to complete summarization tasks, while the latter can assist in identifying shared syntactic or semantic knowledge to enhance the ability to generalize through meta-transfer learning.

For Meta-Transfer Learning for low-resource ABStractive summarization (MTL-ABS), Chen et al. chose the transformer-based encoder-decoder model [72]. Token [SEP] is added as a boundary at the end of each sentence, and a special token [CLS] is added at the start of each phrase to aggregate information. The Multi-Headed Attention (MHA) layer, Feed-Forward (FF) layer, and Layer Normalization (LN) layer are the three sub-layers that make up the primary self-attention (SA) layer. $SA(h) = LN(FF(MHA(h)) + h)$ is the expression for the self-attention layer, where h is the intermediate hidden representation. SA layers are piled on top of the transformer (TF) layer. BERT [24], which is trained on the general domain, is used to initialize the encoder of the basic model. As suggested by earlier research [72], they

refine the encoder using an extractive objective on the selected pre-training corpus before the meta-transfer learning. The study suggests limiting the number of meta-trainable parameters and layers in a large pre-trained model to prevent overfitting and gradient instability, and this is overcome using adapter (ADA) modules as a bottlenecked feed-forward network with a skip-connection from input to output. To use previously learned information when doing meta-learning, we place adapters into each layer of the encoder and decoder. In the transformer layer, the adaptor is introduced specifically after each feed-forward layer. Consequently, layers with adapted self-attention (ADA-SA) and adapted transformer (ADA-TF). Figure 3.20 shows the proposed summarisation framework.

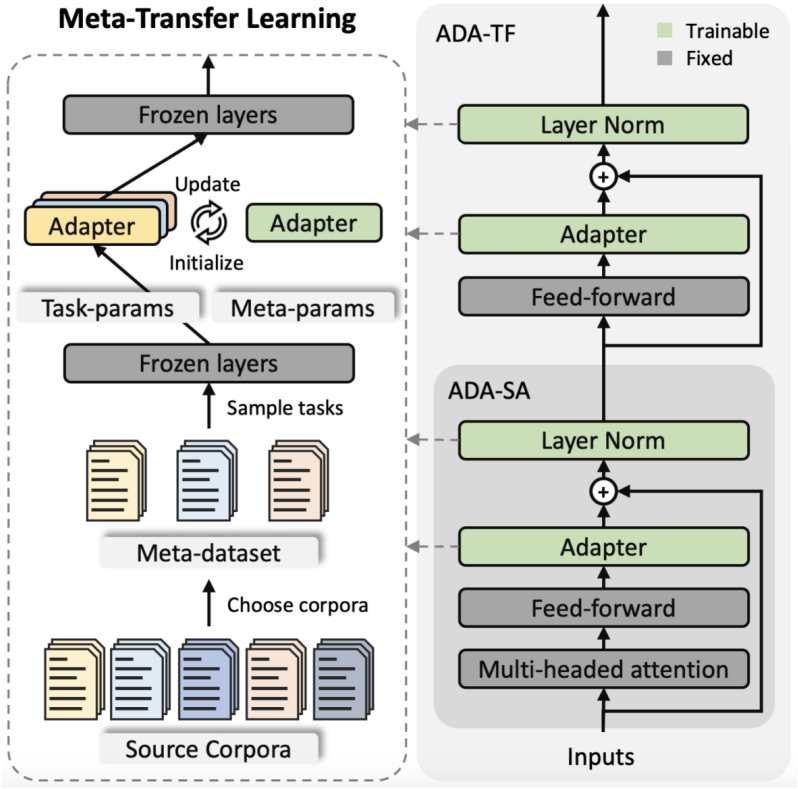


Figure 3.20: A framework for summarization with metatransfer learning is proposed. After each feed-forward layer, the adapter modules are placed into the encoder and decoder. Only the adapters and layer normalization layers are learnable during the meta-transfer learning process. The learning example of the layer normalizing layers is removed for clarity. (Source: [15]).

In MTL-ABS, source corpora choice is guided by similarity rankings: *semantics*, *word overlapping*, *coverage*, *information density* and *length*. Based on the findings of previous similarity metrics, Chen et al. select source corpora for the meta-dataset by averaging the following three criteria: *cosine similarity*, *ROUGE precision* and *article length*.

3.7.17 SPEC

The data features and learning objectives in the pretext tasks have the potential to impact conventional transfer learning techniques. Consequently, Chen et al. provide a meta-learning framework to transfer few-shot learning processes from source corpora to the target corpus, based on pre-trained language models. Earlier techniques use training examples to learn without breaking down the *content* and *preference*. Therefore, the preference bias in the training set may limit the generated summaries, particularly in low-resource environments. To control preferences during inference, the authors of the paper [44] suggest using parameter modulation to break down the contents and preferences during learning. Given a target application, it could be difficult to determine the necessary preferences because they might be difficult to ascertain through observations. To automatically estimate appropriate preferences and produce related summary candidates from the limited number of training instances, SPEC is therefore suggested as a novel decoding technique.

SPEC is similar to MTL-ABS (previous work of Chen et al.), except for the addition of *Summary Preference Decomposition*. This study focuses on estimating preferences between articles and summaries using statistical relations. Preferences are constructed using metrics like ROUGE-1, ROUGE-2, ROUGE-L, extractive diversity/coverage, novel word ratio, and compression ratio. Other perceptive metrics like sentiment or popularity can also be used, but these metrics are used for generality. The goal is to create a module to add more preference data to the summarizer, given a pre-trained base model and a preference vector. Given the preferences characterizing the relationships between articles and summaries and the preference-aware summarizer, we suggest two approaches that make use of various external information sources for learning in low-resource situations: *Intra-Preference Learning* (IPL) and *Inter- and Intra-Preference Learning* (IIPL). The preference-aware summarizer

allows users to provide desired preferences for summaries but may face challenges in defining preferences due to conflicts like extractive diversity and novel word ratio. To address this, *Preference-Match Decoding* (PMD) is proposed, which uses a few annotated examples to generate suitable candidate summaries for user selection automatically. The PMD hypothesizes that a corpus can contain few representative preferences, and the summaries generated can be suitable under the data distribution.

3.7.18 Efficient framework for low-resource abstractive summarization by meta-transfer learning and pointer-generator networks

An efficient framework for low-resource abstractive summarization (EFLRAS) using a pointer-generator network and a meta-learning technique to address the LRS problems is proposed in [16]. Meta-learning using existing high-resource datasets enables the proposed model to rapidly adapt to a new domain using limited data to solve the domain-shifting problem. In addition, the paper cited explored the copy mechanism using a pointer-generator network that can copy words from a source document when generating a summary.

The method proposed by Huh et al. architecture (Figure 3.21) includes:

- Prompt module.
  For a wide range of jobs involving enormous volumes of data, a LLM with many parameters works well. On the other hand, the model is prone to overfitting if a new domain lacks labeled samples. To solve this, the language model is frozen, and a *prompt module* is introduced with few parameters. The module comprises layers that produce value and important metrics to complement the attention mechanism of the model. A new attention mechanism is built by concatenating extra key and value metrics from hidden states, in contrast to the original matrix multiplication attention method.

- Pointer-generator network.
  The *pointer-generator network* is an extension of the sequence-to-sequence model for generating abstractive and extractive summaries using a copy mechanism. The degree of extraction and abstraction is determined by a generation

probability calculated for each decoder timestep. The generation probability is determined from the hidden state of the decoder's last layer, which determines the abstractiveness of a word and the probability of how much it reflects the output of the generative model, a vocabulary distribution. The cross-attention score matrix of the last decoder layer is used to create a copy distribution, with the probability of the token being generated being the attention value at each decoding step.

- Meta-transfer learning.
  Meta-transfer learning is a sequential process of MTL and transfer-learning, with a training framework consisting of MTL with high-resource datasets and transfer-learning with a low-resource dataset. The prompt module is integrated into a frozen language model, but its parameters are randomly initialized. The MAML algorithm [73] is adopted to initialize parameters for rapid adaptation to new domains with small training data. One dataset with randomly sampled data is used for transfer learning, while the remaining are used for meta-learning. The top three target-related datasets are used for fast adaptation using document similarity, based on the average score of similarity of document length, ROUGE-2 precision, and cosine similarity, referring to [15]. This process is repeated as many times as the number of datasets.

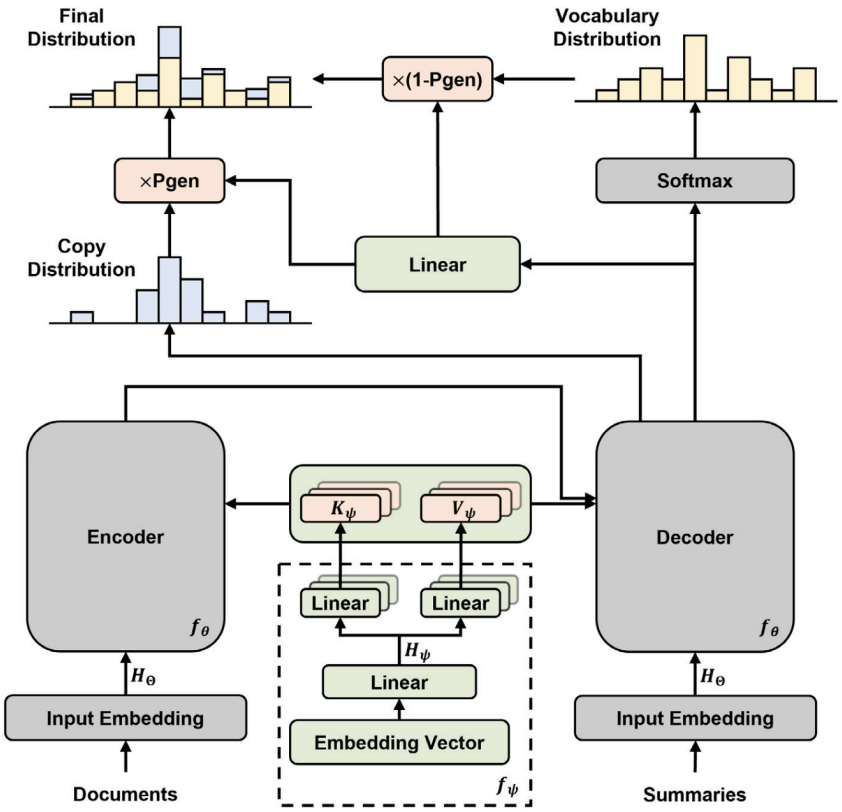


Figure 3.21: The general architecture of the model we have suggested. The dotted line box indicates the prompt module. We integrate the prompt module with the attention mechanism and freeze the language model to restrict the trainable parameters. The generation probability, which determines the weights of copying from an input document or creating tokens from the generation model, is also calculated by adding a linear layer. (Source: [16]).

The authors explore the *effect of copy mechanism*. It is less effective when a few training samples are used, indicating that more data is needed for effective training. In addition, it was found to be more effective on news and conversational datasets. When studying the *effect of attention aggregation* for copy mechanism, it was found that large numbers of duplicate tokens in long documents make the mean attention score closer to the average score. The study proposes using the maximum score instead of the average score for aggregating duplicate token scores.

3.7.19 Parasum

The paper [17] proposes a novel paradigm for low-resource extractive summarization called ParaSum. This paradigm reformulates text summarization as textual paraphrasing, aligning the text summarization task with the self-supervised *Next Sentence Prediction* (NSP) task of PLMs. This approach minimizes the training gap between the summarization model and PLMs, enabling a more effective probing of the knowledge encoded within PLMs and enhancing the summarization performance. Furthermore, to relax the requirement for large amounts of training data, it has been introduced a simple yet efficient model and aligned the training paradigm of summarization to textual paraphrasing to facilitate network-based transfer learning. An overview of the architecture is shown in Figure 3.22.

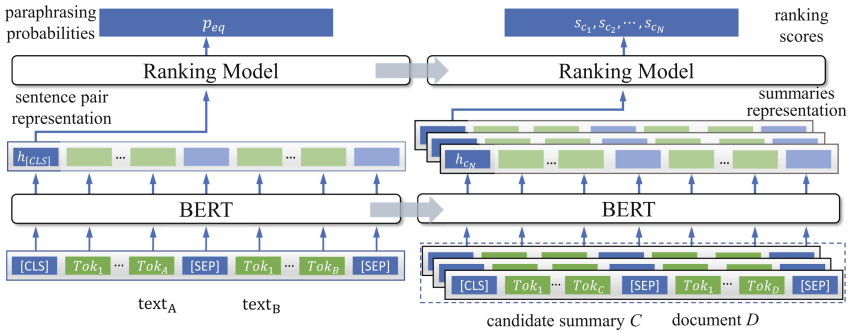


Figure 3.22: The ParaSum model architecture and transfer learning process. (Source: [17]).

3.8 Methods testing

3.8.1 Experimental setup

The following hardware was used to run the tests we completed: GPU GeForce RTX 3090, CPU Intel(R) Core(TM) i9-10900X, 32GB RAM. The Python [74] version is the *3.8*, and the execution environment is a *Docker* [75] container created with the Dockerfile shown in Figure 3.23.

```
FROM huggingface/transformers-pytorch-latest-gpu
# Set work directory
WORKDIR /workspace
RUN pip3 install torch==1.10.2+cu113
   torchvision==0.11.3+cu113 torchaudio==0.10.2+cu113 -f
   https:
   //download.pytorch.org/whl/cu113/torch_stable.html
RUN pip3 install
   git+https://github.com/huggingface/transformers
RUN pip3 install --upgrade nvidia-ml-py3==7.352.0
RUN pip3 install --upgrade sentence-transformers==2.2.2
RUN pip3 install --upgrade accelerate==0.14.0
RUN pip3 install --upgrade codecarbon==2.1.4
RUN pip3 install --upgrade streamlit==1.14.0
RUN pip3 install --upgrade wget==3.2
RUN pip3 install --upgrade tensorflow==2.10.0
RUN pip3 install --upgrade tensorflow-datasets==4.7.0
RUN pip3 install --upgrade scikit-learn==1.1.3
RUN pip3 install --upgrade nltk==3.7
RUN pip3 install --upgrade stqdm==0.0.4
RUN pip3 install --upgrade datasets==2.6.1
RUN pip3 install --upgrade wandb==0.13.5
```

Figure 3.23: The Dockerfile used to generate the container.

3.8.2 PEGASUS-X testing

Except for PEGASUS-X [3.7.11], an extension of PEGASUS [3.7.10] intended for LDS, all previously discussed techniques have been evaluated on LRS scenarios. We tested this strategy on CNN/DailyMail and BillSum datasets, using the same training conditions as PEGASUS (0-shot, 10-shot, and 100-shot), because it's fascinating to examine how the modifications made to PEGASUS affect the performance in LRS tasks.

Setup training

Before starting the training, it is essential to do some preliminary steps.

1. Clone the PEGASUS-X repository[12].

2. Install some extra packages, not included in the base Dockerfile (Figure 3.23): `absl-py, mock, numpy, rouge-score, sacrebleu, sentencepiece, tensorflow-text==1.15.0rc0, tensor2tensor==1.15.0, tensorflow-datasets==2.1.0`.

3. Download the tokenizer and a PEGASUS-X checkpoint (in our case, a non-fine-tuned one); links are provided in the README file in the `pegasus/flax/` directory.

4. Download the dataset[13] you want to fine-tune the model on using the following command:
   `python -m tensorflow_datasets.scripts.download_and_prepare -datasets=<dataset_name>`.

5. In `/pegasus/flax/configs/default.py` change `config.dataset_name`, `config.train_split`, and `config.eval_split` values according to the desired ones. In the file mentioned above, other model configuration settings can be modified.

[12] `https://github.com/google-research/pegasus`

[13] The list of the dataset names is consultable at `https://www.tensorflow.org/datasets/catalog/overview?hl=en`.

6. Create a finetune and an eval configuration file. Figure 3.24 shows the ones we made for the fine-tuning and Figure 3.25 evaluation on the CNN/DailyMail dataset.

```
"""Base-sized Model Hyperparameter configuration."""
from pegasus.flax.configs import pegasus_x_large as
   pegasus_x_large_config

def get_config():
  """Get large-sized PEGASUS-X hyperparameter
     configuration."""

  # Load base config
  config = pegasus_x_large_config.get_config()
  config.run_mode = "train"
  config.dataset_name = "cnn_dailymail"
  config.per_device_batch_size = 1
  config.overwrite_train_steps = 0
  config.learning_rate = 0.003
  config.max_input_length = 6144

  # Replace this:
  config.eval_load_checkpoint_dir = "./"

  # Replace this:
  config.tokenizer_path =
     "./model_data/c4.unigram.newline.10pct.96000.model"

  return config
```

Figure 3.24: PEGASUS-X fine-tune configuration file for CNN/DailyMail dataset.

```
"""Base-sized Model Hyperparameter configuration."""
from pegasus.flax.configs import pegasus_x_large as
   pegasus_x_large_config

def get_config():
  """Get large-sized PEGASUS-X hyperparameter
     configuration."""

  # Load base config
  config = pegasus_x_large_config.get_config()
  config.run_mode = "eval_only"
  config.dataset_name = "cnn_dailymail"
  config.per_device_batch_size = 8
  config.beam_size = 2
  config.beam_alpha = 1.0
  config.max_input_length = 6144

  # Replace these:
  config.eval_load_checkpoint_dir = "./"
  config.eval_step = 9

  # Replace this:
  config.tokenizer_path =
     "./model_data/c4.unigram.newline.10pct.96000.model"

  return config
```

Figure 3.25: PEGASUS-X evaluation configuration file for CNN/DailyMail dataset using 10-shot fine-tuned checkpoint.

Once the previous steps are completed, it is possible to start the training/fine-tuning or evaluation phase using the following command:

```
python -m pegasus.flax.main -config
pegasus/flax/configs/<config_file> -workdir ./<output_dir>
```

where `config_file` is the name of the configuration file created in step 6 and `output_dir` is the path of the directory where output files will be saved. A model checkpoint is required to begin the evaluation process. As a result, the only situation in which training is not needed is when testing 0-shot summarization, as step 3 suffices.

3.8.3 UNISUMM testing

Since many approaches have been tested in a 0-shot context, we choose to assess UNISUMM [3.7.6] on these scenarios as well in order to offer better leaderboards. The fact that UNISUMM is a pretty new open-source project with direct Microsoft support was a deciding factor in the decision.

Setup evaluation

In this specific case, a training phase is not required because the goal is to test UNISUMM on 0-shot summarization. Before the evaluation phase a brief preliminary setup is needed.

1. Download the model public checkpoint and SummZoo (major information about this benchmark are in [7]) data from UNISUMM repository[14].

2. Install `tensorboardX, transformers==4.2` and `nvidia-apex`[15] packages.

3. Set the environment variables that will be used from the evaluation script (e.g. SAMSum:
   ```
   export MODEL_PATH=../unisumm_model/ckpt-300000
   export SAVE_PATH=../unisumm_outs/samsum0
   export TASK_MAP_FILE=../unisumm_model/task_map.json
   export INPUT_FILE=
   ../data/Summzoo/samsum/test/samsum.test.bart.uncased.jsonl
   ```
).

[14] `https://github.com/microsoft/UniSumm`
[15] `https://github.com/NVIDIA/apex`

After the previous steps now it is possible to evaluate the model. Below the configuration we used:

```
CUDA_VISIBLE_DEVICES=0 python decode_seq2seq.py
   --fp16
   --do_lower_case
   --model_path $MODEL_PATH
   --max_seq_length 2048
   --max_tgt_length 256
   --batch_size 4
   --beam_size 5
   --length_penalty 0.6
   --mode s2s
   --min_len 60
   --input_file $INPUT_FILE
```

Chapter 4

Results and Discussion

4.1 LRS Timeline

As demonstrated in Figure 3.2, most examined techniques were introduced starting from 2022, increasing the density of LRS approach publications over time. In order to streamline the research, we offer a timeline (Figure 4.1) that allows users to quickly view the dates of method releases and indicates whether they are open-source or closed-source.

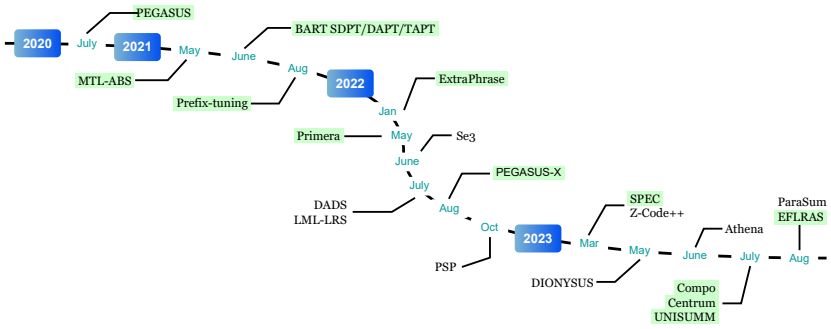


Figure 4.1: A timeline of analyzed methods sorted based on their release date (e.g. submission date to arXiv). We have highlighted the methods that have open-source code.

4.2 Analysis of Results

The methods scores are arranged in dataset-based leaderboards; as previously indicated, the ROUGE Score is the standard evaluation metric. The number of training samples taken into consideration, the ROUGE-1, ROUGE-2, and ROUGE-L scores, and an approach are indicated on each line of a leaderboard. A comprehensive overview of the survey analysis is provided by grouping the methods according to the set and subset of the proposed taxonomy. The SAMSum leaderboard is the only one for dialogue summarization and is displayed in Table 4.3; the other leaderboards for CNN/DailyMail, XSum, BillSum, and Multi-News are represented, respectively, in Tables 4.2, 4.5, Table 4.4, and Table 4.1.

	Method	#Data	R-1	R-2	R-L
Pre-training	$PEGASUS_{large}$	0	36.54	10.52	18.67
	$PEGASUS_{large}$	10	39.79	12.56	20.06
	$PEGASUS_{large}$	100	41.04	13.88	21.52
	Primera	0	39.09	13.91	19.19
	Primera	10	44.02	15.54	22.03
	Primera	100	46.01	16.76	22.91
CbPT	Centrum	0	43.5	15.7	22.4
	Centrum	10	43.4	16.6	22.2
	Centrum	100	45.7	16.8	23.2
MTR	EFLRAS	10	43.60	14.85	20.70
	EFLRAS	100	45.55	16.01	22.12
	MTL-ABS	10	38.88	12.78	19.88
	MTL-ABS	100	39.64	13.64	20.45

Table 4.1: F1-based ROUGE scores (R-1=ROUGE-1, R-2=ROUGE-2, R-L=ROUGE-L) for document summarization task on **Multi-News** dataset in low resource settings. The *Data* column represents the dataset samples used for the training. In PSP "en.", "de." and "ip." are short for encoder, decoder and inner prompts. In LED_{base} w/Se3, the 512 is the max chunk size. MTR stands for Meta-transfer and CbPT for Centroid-based Pre-training.

	Method	#Data	R-1	R-2	R-L
Pre-training	Z-Code++$_{large}$	0	40.00	17.30	25.30
	Z-Code++$_{large}$	10	40.00	17.30	25.30
	Z-Code++$_{large}$	100	41.10	18.40	27.50
	PEGASUS$_{large}$	0	32.90	13.28	29.38
	PEGASUS$_{large}$	10	37.25	15.84	33.49
	PEGASUS$_{large}$	100	40.28	18.21	37.03
	PEGASUS-X$_{large}$	0	30.22	11.88	28.31
	PEGASUS-X$_{large}$	10	36.12	13.70	30.26
	PEGASUS-X$_{large}$	100	38.40	17.02	36.75
	PSP	300	38.31	15.94	25.41
A	ExtraPhrase	1k	34.47	12.91	31.36
MTL	LML-LRS	10	39.34	16.53	25.40
	LML-LRS	100	39.94	16.96	26.09
MTR	EFLRAS	10	39.50	16.80	25.72
	EFLRAS	100	40.53	17.61	26.64
E	ParaSum	200	40.81	17.78	36.94

Table 4.2: F1-based ROUGE scores (R-1=ROUGE-1, R-2=ROUGE-2, R-L=ROUGE-L) for document summarization task on **CNN/DailyMail** dataset in low resource settings. The *Data* column represents the dataset samples used for the training. In PSP "en.", "de." and "ip." are short for encoder, decoder and inner prompts. In LED$_{base}$ w/Se3, the 512 is the max chunk size. MTR stands for Meta-transfer, MTL for Meta-learning, E for ES and A for Data augmentation.

	Method	#Data	R-1	R-2	R-L
Pre-training	DIONYSUS$_{base}$	0	39.60	15.40	30.10
	DIONYSUS$_{large}$	0	41.30	16.20	30.90
	BART SDPT$_{w/RecAdam}$	300	45.23	19.43	35.37
	BART DAPT	300	41.22	17.88	32.40
	BART TAPT$_{w/RecAdam}$	300	41.34	17.88	32.31
	Z-Code++$_{large}$	0	26.50	7.90	20.50
	Z-Code++$_{large}$	10	40.27	17.40	33.70
	Z-Code++$_{large}$	100	47.60	22.30	38.70
	UNISUMM	0	22.17	6.88	17.08
	UNISUMM	10	43.89	18.53	34.76
	UNISUMM	100	46.93	20.65	37.28
Data augmentation	DADS	10	32.50	12.00	27.00
	DADS	100	43.90	19.70	36.10
	self-training				
	COMPO$_{base}$	147	45.42	21.23	41.42
	COMPO$_{large}$	147	49.78	24.65	45.41
	joint-training				
	COMPO$_{base}$	147	44.89	20.64	40.58
	COMPO$_{large}$	147	49.14	23.45	44.35
MTR	SPEC	10	46.06	20.90	40.34
	SPEC	100	51.94	24.75	46.97

Table 4.3: F1-based ROUGE scores (R-1=ROUGE-1, R-2=ROUGE-2, R-L=ROUGE-L) for dialogue summarization task on **SAMSum** dataset in low resource settings. The *Data* column represents the dataset samples used for the training. MTR stands for Meta-transfer.

	Method	#Data	R-1	R-2	R-L
Pre-training	PEGASUS$_{large}$	0	41.02	17.44	25.24
	PEGASUS$_{large}$	10	40.48	18.49	27.27
	PEGASUS$_{large}$	100	44.78	26.40	34.40
	PEGASUS-X$_{large}$	0	41.32	18.04	25.11
	PEGASUS-X$_{large}$	10	42.55	18.97	26.92
	PEGASUS-X$_{large}$	100	46.48	27.77	36.53
SEG	LED$_{base(512)}$w/Se3	10	46.94	23.04	29.29
	LED$_{base(512)}$w/Se3	100	50.4	27.73	33.74
	Athena	10	47.57	24.14	30.35
	Athena	100	51.59	29.36	35.04
MTL	LML-LRS	10	46.64	25.07	30.90
	LML-LRS	100	48.18	27.18	33.28
MTR	MTL-ABS	10	41.22	18.61	26.33
	MTL-ABS	100	45.29	22.74	29.56
	EFLRAS	10	46.64	25.07	30.90
	EFLRAS	100	48.18	27.18	33.28

Table 4.4: F1-based ROUGE scores (R-1=ROUGE-1, R-2=ROUGE-2, R-L=ROUGE-L) for document summarization task on **BillSum** dataset in low resource settings. The *Data* column represents the dataset samples used for the training. In PSP "en.", "de." and "ip." are short for encoder, decoder and inner prompts. In LED$_{base}$ w/Se3, the 512 is the max chunk size. MTR stands for Meta-transfer, MTL for Meta-learning and SEG for Segmentation.

	Method	#Data	R-1	R-2	R-L
Pre-training	PEGASUS$_{large}$	0	19.27	3.00	12.72
	PEGASUS$_{large}$	10	19.39	3.45	14.02
	PEGASUS$_{large}$	100	39.07	16.44	31.27
	PSP	300	32.86	11.27	25.64
	UNISUMM	0	20.72	3.62	16.56
	UNISUMM	10	26.10	7.20	19.92
	UNISUMM	100	33.33	11.36	25.85
MTL	LML-LRS	10	32.35	11.86	25.33
	LML-LRS	100	35.54	13.94	27.79
MTR	EFLRAS	10	32.65	12.10	25.82
	EFLRAS	100	36.51	14.55	29.01
	SPEC	10	32.74	10.90	24.86
	SPEC	100	35.69	12.88	27.25
E	ParaSum	1000	21.15	3.08	15.91
P	Prefix-tuning	100	35.20	13.30	28.10

Table 4.5: F1-based ROUGE scores (R-1=ROUGE-1, R-2=ROUGE-2, R-L=ROUGE-L) for document summarization task on **XSum** dataset in low resource settings. The *Data* column represents the dataset samples used for the training. In PSP "en.", "de." and "ip." are short for encoder, decoder and inner prompts. In LED$_{base}$ w/Se3, the 512 is the max chunk size. MTR stands for Meta-transfer, MTL for Meta-learning, E for ES and P for Prefix tuning.

4.2.1 Hugging Face leaderboards

To provide better navigability and more simple access, I recreated the leaderboards using *Hugging Face Spaces*[1], which facilitates the creation, hosting, and sharing of web apps. Hugging Face (HF) can permanently host webpages with *Streamlit* [76], *Gradio* [77], *Docker* [75] or static *HTML* [78] as the backend. Between the latters, we chose Gradio because it can be presented as a webpage and embedded in Python code, working in harmony with Pandas [79] and any other

[1] `https://huggingface.co/spaces`

Python library and framework.

Two key benefits are introduced by our web app[2]. Firstly, the techniques in the leaderboards can be arranged based on the ROUGE-1, ROUGE-2, or ROUGE-L scores, as well as on the taxonomy set or quantity of training samples employed. The second is that, unlike static publications (e.g. PDF format), you can expand and update the leaderboards anytime. A Git [80] repository is used to handle the source data of HF Spaces projects; therefore, it is also possible to expand the app's content with outside assistance.

4.3 Discussion

The training data's quantity significantly impacts the methods' performance in the summarization context. This fact demonstrates again the importance of LRS. This work's focus is on providing researchers with a source where the best approaches to deal with the challenge cited in the previous sentence are collected and analyzed.

It is evident from the timeline we gave and the leaderboards, particularly 4.2.1, that the more training samples there are overall, the higher the score. Nonetheless, the most recent proposals achieve highly intriguing performance, often outperforming earlier models trained on a larger number of samples.

The best results in dialogue summarization are obtained with a *meta-transfer* strategy, SPEC, which performs significantly better (among the best) in 10-shot configuration and surpasses all other approaches in 100-shot configuration. Additionally, two entirely distinct strategies — Z-Code++, a *pre-trained* model, and COMPO, based on *data augmentation* — achieve extremely good performance. Not to be overlooked is DIONYSUS, which achieves remarkable results on 0-shot summarization — particularly when compared to its rivals — even in its *base* configuration.

Talking about document summarization, the analysis is more difficult because there isn't a single/few datasets considered by every method, and it is impossible to test every method on a common dataset due to the closed-source nature of part of the methods. In the most simple scenario, represented by CNN/DailyMail, the best ROUGE scores are obtained by PEGASUS and Parasum, respectively *pre-trained*

[2] `https://huggingface.co/spaces/ema-arte/LRS-leaderboard`

and *ES* models. Regarding document summarization, the analysis becomes more challenging since each approach takes into account different datasets, not all in common. Because some methods are closed-source, testing each method on a shared dataset is not feasible. The best ROUGE scores are obtained by PEGASUS and Parasum, respectively *pre-trained* and *ES* models, in the simplest situation, which is represented by CNN/DailyMail. Next up is Multi-news, a dataset created primarily for MDS. Two *pre-trained* models created especially for MDS, Centrum and Primera, perform better in this context than any other technique, both in 10-shot and 0-shot summarization (valid only for Centrum). The only method that yields comparable outcomes in 100-shot scenario is EFLRAS, founded on *meta-transfer* learning. Proceeding to the examination of the outcomes acquired on XSum, where the goal synopsis is reduced to a single sentence, we discover that PEGASUS is leading by a considerable margin; EFLRAS and LML-LRS follow, creating a scenario akin to that which was observed when evaluating the performance on CNN/DailyMail. Additionally, only the *prefix-tuning* approach — significantly lighter than the others — achieves outstanding results. The dataset considered in this work for LDS in LRS context is BillSum. Athena and Se3, the only two techniques that use *segmentation* for data augmentation, yield the best outcomes. The latter are followed in the leaderboard by PEGASUS-X, a version of PEGASUS specifically designed for this purpose, after that we find LML-LRS and EFLRAS.

It may be deduced from the preceding considerations that PEGASUS-X meets the authors' goals while performing marginally worse than PEGASUS in all situations but LDS. On the other hand, LML-LRS and EFLRAS perform admirably on all tasks without ever excelling. The best options are Centrum and Primera when the objective task is MDS; however, SPEC is the best option when the objective task is dialogue summarization. As an "evolution" of Se3, which has already shown excellent outcomes, Athena is the finest method to depend on for LDS tasks. PEGASUS is often the finest baseline to consider for document summarizing because it consistently performs interestingly and has shown to be the best in various situations.

Conclusion and Future Work

La mia tesi si conclude evidenziando gli importanti contributi al campo dell'LRS che il survey alla sua base ha apportato che non sono stati affrontati da un lavoro di questo tipo prima d'ora. A causa della sua applicazione pratica in scenari in cui la scarsità di dati è una limitazione significativa, LRS è cresciuto di importanza come tema di ricerca. La panoramica approfondita di ogni tecnica di LRS accelera il lavoro dei ricercatori, consentendo alla comunità di comprendere i punti chiave dell'implementazione senza dover leggere l'intera pubblicazione. Il primo contributo consiste nel fornire una definizione chiara del termine "few-shot", che dopo un esame approfondito può essere definito come 10-shot o 100-shot. Inoltre, abbiamo presentato un'analisi dei metodi utilizzati nei diversi progetti presi in considerazione per scegliere i dati di addestramento e i risultati hanno mostrato che nella maggior parte dei casi viene utilizzato un campionamento casuale. Per classificare gli insiemi e i sottoinsiemi dei metodi di LRS, si propone una tassonomia con cinque gruppi e i corrispondenti sottogruppi. *Data augmentation* si riferisce a tutte le strategie che affrontano la scarsità di dati e sviluppano modi innovativi per aumentarne la quantità; *Segmentation* è un sottoinsieme di questa categoria. *Prefix tuning* raggruppa i metodi che ottimizzano una sequenza di vettori continui specifici per il compito, chiamati *prefix*, invece di modificare i parametri del modello linguistico, mentre *Extractive summarization* raccoglie i metodi che generano una sinossi usando solo parole e frasi prese dai testi in input. Mentre *Meta-transfer* comprende le strategie che si concentrano sul trasferimento della conoscenza e sull'adattamento del modello tra vari compiti, domini o ambienti, e *Meta-learning*, noto anche come "imparare a imparare", si concentra sul miglioramento del processo di apprendimento per conoscere meglio nuovi compiti o adattarsi rapidamente a nuovi dati. *Pre-training* è l'ultima categoria della tassonomia proposta; descrive la prima fase

dell'addestramento di un modello su un grande insieme di dati o su un modello già esistente prima di ottimizzarlo per un particolare compito; *Pre-training basato su centroidi* è un sottoinsieme di quest'ultimo e si distingue da esso per l'uso di un centroide come delega del riassunto.

La sezione dello studio dedicata alle classifiche è la più interessante. Determinare quali approcci funzionano meglio per ogni compito è possibile analizzando questi ultimi e stabilendo elevati nuovi parametri di riferimento. Il metodo meta-transfer SPEC supera tutti i concorrenti nel *riassunto dei dialoghi*; tuttavia, DIONYSUS è in testa con un margine considerevole nel caso 0-shot. PEGASUS è lo standard di base più importante da prendere in considerazione nella riepilogazione dei documenti, poiché ha sempre ottenuto buoni risultati ed ha dimostrato di essere il migliore sia con impostazioni di riepilogo classiche che estreme. Centrum e Primera sono più efficaci di altri metodi quando si tratta di riassumere documenti multipli. Tuttavia, quando si considera la sintesi di documenti lunghi, la storia cambia, poiché Athena utilizza il suo approccio di aumento dei dati per superare tutti i risultati ottenuti dagli altri metodi.

In conclusione, il survey proposto rappresenta un nuovo punto di svolta nel campo del LRS. Tutti gli studiosi possono trarre vantaggio dalle preziose conoscenze e dalle definizioni formali che offre, le quali possono contribuire a far progredire questa disciplina sempre più importante nella sfera dell'apprendimento automatico.

Come lavoro futuro, ci concentreremo sul mantenere aggiornata la classifica Hugging Face, aggiungendo le tecniche che saranno presentate negli anni successivi. Inoltre, esamineremo una gamma più ampia di dataset e valuteremo gli approcci open-source in scenari a 0 o few-shot che non sono stati valutati in precedenza. L'inclusione di valutazioni effettuate con metriche diverse dai punteggi ROUGE è un'ulteriore opzione. Tutte queste aggiunte mirano a sostenere ancora di più la comunità di ricercatori che si occupano di ricerca in contesto LRS. Inoltre, il lavoro sarà presentato alla principale conferenza sull'intelligenza artificiale, l'International Joint Conferences on Artificial Intelligence (IJCAI).

My thesis concludes by highlighting the important contributions to the field of LRS that the survey at its foundation made that haven't been addressed by a work of this kind before. Due to its practical application in scenarios where data scarcity is a significant limitation, few-shot summarization has grown in importance as a research issue. The in-depth overview of every LRS technique expedites the researchers' work by enabling the community to understand the key implementation points without having to read the whole publication. Contribution number one is providing a clear definition of the term "few-shot", which can be defined as either 10-shot or 100-shot after a thorough examination. Furthermore, we presented an analysis of the methods utilized in the different projects under consideration to pick training data, and the results showed that most of the time, random sampling is employed. A taxonomy with five groups and their corresponding subgroups is suggested to categorize the sets and subsets of LRS methods. *Data augmentation* refers to any strategies that deal with data scarcity and develop innovative ways to boost the amount of available data; *Segmentation* is a subset of this category. *Prefix tuning* groups methods that optimize a sequence of continuous task-specific vectors called the *prefix* instead of fine-tuning language model parameters, and *Extractive summarization* gathers methods that generate a synopsis using only words and sentences taken from input texts. While *Meta-transfer* encompasses strategies that concentrate on knowledge transfer and model adaptation across various tasks, domains, or environments, and *Meta-learning*, also known as "learning to learn," focuses on enhancing the learning process to know new tasks better or quickly adapt to new data. *Pre-training* is the final category in the taxonomy proposed; it describes the first stage of training a model on a big dataset or an already-existing model before optimizing it for a particular task; *Centroid-based pre-training* is a subset of the latter and is distinguished from it by the use of a centroid as summary proxy.

The leaderboard section of the study is the most interesting. Determining which approaches work best for each task is feasible by analyzing the latters and setting new, high baselines. The meta-transfer method SPEC surpasses all competitors in *dialogue summarization*; nevertheless, DIONYSUS leads by a considerable margin in the 0-shot case. PEGASUS is the greatest baseline to take into account in *document summarization* since it constantly performs well and has proven to be the best in both classic and extreme summarizing settings. Centrum and Primera are more effective

than other methods when it comes to MDS. However, when LDS is considered, the narrative shifts, as Athena uses its data augmentation approach to surpass all the results obtained by the other methods.

In conclusion, the suggested survey is a new turning point in the LRS field. All academics can benefit from the valuable knowledge and formal definitions it offers, which can assist in advancing this increasingly significant discipline of machine learning.

As future work, we will focus on keeping the HF leaderboard up-to-date, adding the techniques that will be presented in the following years. Additionally, we will examine a wider range of datasets and assess open-source approaches in 0-shot or few-shot scenarios that haven't been evaluated before. Including evaluations carried out using metrics other than ROUGE scores is an additional option. All these additions aim to support the community of researchers working on LRS research even more. In addition, the work will be submitted to the leading artificial intelligence conference, the International Joint Conferences on Artificial Intelligence (IJCAI).

Acknowledgements

The completion of this thesis is the culmination of extensive research and analysis of proposed implementations in the field of LRS. I am deeply grateful to all those who have played a significant role in shaping this work.

First and foremost, I would like to express my profound gratitude to my dedicated supervisor, Professor GIANLUCA MORO, whose guidance, expertise, and support have been the cornerstone of this research.

I make my heartfelt thanks to my supervisor, Professor GIANLUCA MORO, and his team at DISI UniBo NLP. Their generous sharing of resources, expertise, and support have been instrumental in the successful execution of my research approach. I would like to extend a special acknowledgement to GIACOMO FRISONI and LUCA RAGAZZI, whose expert insights and valuable advice have significantly shaped the trajectory of this work.

To my family, who have been my pillars of strength throughout this academic journey, I extend my deepest thanks. Your love, encouragement, and unwavering belief in me have been my constant motivation.

EMANUELE ARTEGIANI
NOVEMBER, 2023

Bibliography

[1] Yujian Tang. What is ai text summarization and how can i use it?, 2021. URL https://pythonalgos.com/2021/11/27/what-is-ai-text-summarization-and-how-can-i-use-it/.

[2] Shen Gao, Xiuying Chen, Zhaochun Ren, Dongyan Zhao, and Rui Yan. From standard summarization to new tasks and beyond: Summarization with manifold information, 2020.

[3] Yongtai Liu, Joshua Maynez, Gonçalo Simões, and Shashi Narayan. Data augmentation for low-resource dialogue summarization. July 2022.

[4] Yu Li, Baolin Peng, Pengcheng He, Michel Galley, Zhou Yu, and Jianfeng Gao. Dionysus: A pre-trained model for low-resource dialogue summarization. *arXiv*, May 2023.

[5] Tiezheng Yu, Zihan Liu, and Pascale Fung. Adaptsum: Towards low-resource domain adaptation for abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 5892–5904, June 2021.

[6] Pengcheng He, Baolin Peng, Song Wang, Yang Liu, Ruochen Xu, Hany Hassan Awadalla, Yu Shi, Chenguang Zhu, Wayne Xiong, Michael Zeng, Jianfeng Gao, and Xuedong Huang. Z-code++: A pre-trained language model optimized for abstractive summarization. *arXiv*, Mar 2023.

[7] Yulong Chen, Yang Liu, Ruochen Xu, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Yue Zhang. Unisumm and summzoo: Unified model and diverse

benchmark for few-shot summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, volume 1: Long Papers, page 12833–12855, July 2023.

[8] Mengsay Loem, Sho Takase, Masahiro Kaneko, and Naoaki Okazaki. Extraphrase: Efficient data augmentation for abstractive summarization. *arXiv*, Jan 2022.

[9] Gianluca Moro and Luca Ragazzi. Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11085–11093, June 2022.

[10] Gianluca Moro and Luca Ragazzi. Align-then-abstract representation learning for low-resource summarization. *Neurocomputing*, page 126356, June 2023.

[11] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv*, July 2020.

[12] Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. Primera: Pyramid-based masked sentence pre-training for multi-document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, volume 1: Long Papers, pages 5245–5263, May 2022.

[13] Xiaochen Liu, Yang Gao, Yu Bai, Jiawei Li, Yinan Hu, Heyan Huang, and Boxing Chen. Psp: Pre-trained soft prompts for few-shot abstractive summarization. In *Proceedings of the 29th International Conference on Computational Linguistics*, page 6355–6368, Oct 2022.

[14] Taehun Huh and Youngjoong Ko. Lightweight meta-learning for low-resource abstractive summarization. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2629–2633, July 2022.

[15] Yi-Syuan Chen and Hong-Han Shuai. Meta-transfer learning for low-resource abstractive summarization. *arXiv*, May 2021.

[16] Taehun Huh and Youngjoong Ko. Efficient framework for low-resource abstractive summarization by meta-transfer learning and pointer-generator networks. *Expert Systems with Applications*, Aug 2023.

[17] Moming Tang, Chengyu Wang, Jianing Wang, Cen Chen, Ming Gao, , and Weining Qian. Parasum: Contrastive paraphrasing for low-resource extractive text summarization. In *Knowledge Science, Engineering and Management (KSEM)*, volume LNAI 14119, pages 106–119. Springer, Aug 2023.

[18] Laura Mascarell, Ribin Chalumattu, and Julien Heitmann. Entropy-based sampling for abstractive multi-document summarization in low-resource settings. In C. Maria Keet, Hung-Yi Lee, and Sina Zarrieß, editors, *Proceedings of the 16th International Natural Language Generation Conference*, pages 123–133, Prague, Czechia, September 2023. Association for Computational Linguistics.

[19] Nida Shafiq, Isma Hamid, Muhammad Asif, Qamar Nawaz, Hanan Aljuaid, and Hamid Ali. Abstractive text summarization of low-resourced languages using deep learning. *PeerJ Computer Science*, 9:e1176, 2023.

[20] Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, Giacomo Frisoni, Claudio Sartori, and Gustavo Marfia. Efficient memory-enhanced transformer for long-document summarization in low-resource regimes. *Sensors*, 23(7), 2023. ISSN 1424-8220. doi: 10.3390/s23073542.

[21] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. Benchmarking large language models for news summarization, 2022.

[22] IBM. What is natural language processing? URL `https://www.ibm.com/topics/natural-language-processing`.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

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

[26] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. URL `https://api.semanticscholar.org/CorpusID:49313245`.

[27] BigScience Workshop et al. Bloom: A 176b-parameter multilingual language model, 2023.

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

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

[30] Xiachong Feng, Xiaocheng Feng, and Bing Qin. A survey on dialogue summarization: Recent advances and new frontiers, 2022.

[31] Xiachong Feng, Xiaocheng Feng, Libo Qin, Bing Qin, and Ting Liu. Language model as an annotator: Exploring dialogpt for dialogue summarization. In *Proc. of ACLIJCNLP*, 2021.

[32] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. Pretrained language models for text generation: A survey, 2021.

[33] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: enhanced language representation with informative entities. In *ACL*, 2019.

[34] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation, 2019.

[35] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*, 2020.

[36] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2019.

[37] Rebecca M Blank. Guide for conducting risk assessments, 2011.

[38] Xiuying Chen, Zhangming Chan, Shen Gao, Meng-Hsuan Yu, Dongyan Zhao, and Rui Yan. Learning towards abstractive timeline summarization. In *IJCAI*, 2019.

[39] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, 2019.

[40] Shen Gao, Xiuying Chen, Piji Li, Zhaochun Ren, Lidong Bing, Dongyan Zhao, and Rui Yan. Abstractive text summarization by incorporating reader comments. In *AAAI*, 2019.

[41] Ian Goodfellow, Jean PougetAbadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, page 2672–2680, 2014.

[42] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 6546–6555, 2018.

[43] Jiaan Wang, Fandong Meng, Duo Zheng, Yunlong Liang, Zhixu Li, Jianfeng Qu, and Jie Zhou. A survey on cross-lingual summarization, 2022.

[44] Yi-Syuan Chen, Yun-Zhu Song, and Hong-Han Shuai. Spec: Summary preference decomposition for low-resource abstractive summarization. *arXiv*, Mar 2023.

[45] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *In Text Summarization Branches Out*, 2004.

[46] Kishore Papineni, Salim Roukos, Todd Ward, , and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, July 2002.

[47] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020.

[48] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, page 65–72, June 2005.

[49] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointergenerator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, page 1073–1083, 2017.

[50] Shashi Narayan, Shay B. Cohen, and Mirella 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*, 2018.

[51] Kornilova, Anastassia, Eidelman, and Vladimir. Billsum: A corpus for automatic summarization of US legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5406. URL `https://aclanthology.org/D19-5406`.

[52] Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and ab-

stractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1102. URL https://aclanthology.org/P19-1102.

[53] Siru Ouyang, Jiaao Chen, Jiawei Han, and Diyi Yang. Compositional data augmentation for abstractive conversation summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, volume 1, page 1471–1488, July 2023.

[54] Xiang Lisa Li and Percy 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*, page 4582–4597, Aug 2021.

[55] Jason Phang, Yao Zhao, and Peter J. Liu. Investigating efficiently extending transformers for long input summarization. *arXiv*, Aug 2022.

[56] Ratish Puduppully, Parag Jain, Nancy F. Chen, and Mark Steedman. Multi-document summarization with centroid-based pretraining. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, volume 2: Short Papers, page 128–138, July 2023.

[57] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing, 2017.

[58] Daniel Matthew Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder, 2018.

[59] Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. Towards a human-like open-domain chatbot, 2020.

[60] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation, 2020.

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

[62] Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 7870–7881, 2020.

[63] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pretraining text encoders as discriminators rather than generators. In *ICLR*, 2020.

[64] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

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

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

[67] Ani Nenkova and Rebecca Passonneau. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL*, pages 145–152. Association for Computational Linguistics, 2004.

[68] Jiawei Han, Micheline Kamber, and Jian Pei. Cluster centroid. https://www.sciencedirect.com/topics/computer-science/cluster-centroid, 2012.

[69] Xiaotao Gu, Yuning Mao, Jiawei Han, Jialu Liu, Hongkun Yu, You Wu, Cong Yu, Daniel Finnie, Jiaqi Zhai, and Nicholas Zukoski. Generating representative headlines for news stories, 2020.

[70] Nikolai Vogler, Songlin Li, Yujie Xu, Yujian Mi, and Taylor Berg-Kirkpatrick. An unsupervised masking objective for abstractive multi-document news summarization, 2022.

[71] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.

[72] Yang Liu and Mirella Lapata. Text summarization with pretrained encoders, 2019.

[73] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

[74] Python. URL https://www.python.org/.

[75] Dirk Merkel. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), 2014. ISSN 1075-3583.

[76] Streamlit. URL https://streamlit.io/.

[77] Abubakar Abid, Ali Abdalla, Ali Abid, Dawood Khan, Abdulrahman Alfozan, and James Zou. Gradio: Hassle-free sharing and testing of ml models in the wild. *arXiv preprint arXiv:1906.02569*, 2019.

[78] Hypertext markup language (html). URL https://it.wikipedia.org/wiki/HTML.

[79] The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL https://doi.org/10.5281/zenodo.3509134.

[80] Linus Torvalds. Git, 2005. URL https://git-scm.com/.