VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

FINAL REPORT ASSIGNMENT OF DEEP LEARNING

Supervisor: **Assoc. Prof. Lê Anh Cường**

Authors: **Nguyễn Trung Nguyên-520V0015**

Hồ Trọng Nghĩa-520K0163

Class: **20K50301**

School year: **24**

HO CHI MINH CITY, 2022

VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

FINAL REPORT ASSIGNMENT OF DEEP LEARNING

Supervisor: **Assoc. Prof. Lê Anh Cường**

Author: **Nguyễn Trung Nguyên-520V0015**

Hồ Trọng Nghĩa-520K0163

Class: **20K50301**

School year: **24**

HO CHI MINH CITY, 2022

ACKNOWLEDGEMENT

Firstly, We would like to express our deep gratitude to Assoc. Prof. **Lê Anh Cường**. During the teaching process, you were extremely enthusiastic and dedicated to teaching us. You have helped me gain more knowledge.

This subject is very interesting, but because our knowledge and skills are still lacking, it is difficult to avoid many mistakes in the process of doing the test. We hope that you will review and comment to help us improve our report.

We sincerely thank you!

PROJECT IS COMPLETED AT TON DUC THANG UNIVERTY

I commit that this project is my own project and is supervised by Assoc. Prof. Lê Anh Cường;. The contents, results in this topic are honest and unpublished in any form before. The data in the tables for analyzing, commenting and evaluating are collected from various sources and by the authors and the citations are specified in References section.

In addition, a number of comments and assessments as well as data from other authors and organizations are also used in the project are referenced and annotated clearly.

If this project has any cheating or plagiarism, I take full responsibility for the content of my project. Ton Duc Thang University is not related to infringement of copyright caused by me during the process of this project implementation.

Ho Chi Minh City, May 5th , 2023

Author

(signature and full name)

Nguyễn Trung Nguyên

Hồ Trọng Nghĩa

ABSTRACT

The theoretical MVP is a multi-task supervised pre-training model for natural language generation. Its architecture includes a transformer-based encoder-decoder with attention mechanism. It is characterized by its ability to pre-train on multiple tasks simultaneously, improving the quality and efficiency of text generation.

Pros of this model include improved performance on multiple tasks, reduced training time, and the ability to learn from diverse data sources. Cons include the potential for overfitting and the need for large amounts of training data.

When compared to other models such as BERT, GPT-2, and T5, the MVP model has the advantage of multi-task pre-training and improved performance on multiple text generation tasks. Model development information includes the team of developers and their approach to training and evaluating the model.

TABLE OF CONTENTS

CONTENTS

1) Description of problem:

+ Story Generation:

Story generation is a problem in the field of natural language processing (NLP) whose goal is to create stories or paragraphs by using computers to automatically write words, sentences, and paragraphs based on inputs such as subject, keyword, or format.

To create a story or paragraph, we need to do the following steps:

Collecting data: To create a story, we need a large data set of stories or passages, which can be drawn from various sources such as websites, books, magazines or newsletters.

Data preprocessing: Collected data often contains noise and unnecessary information. We need to preprocess the data to remove unnecessary information and create a normalized data set.

Model building: After preprocessing the data, we need to build a model to create the story. This model is usually built using machine learning techniques and artificial neural networks.

Model training: After building the model, we need to train the model with normalized data so that the model can learn and find relationships between words and sentences.

Model evaluation: After training the model, we need to evaluate the model using metrics such as perplexity, BLEU or ROUGE to evaluate the output quality of the model.

Using the model: Finally, we can use the trained model to create new stories or passages by providing the model with some information such as topics, keywords, or formats. The model automatically generates sentences and paragraphs based on the information provided.

To create a good story generation post, we need the following components:

Sufficient and diverse input data to increase the diversity of the generated stories.

A well-designed model that can capture the essence of the input data and produce coherent and natural stories.

Adequate computational resources to train the model and generate stories in a timely manner.

Evaluation metrics and human evaluation to measure the quality of the generated stories.

A clear understanding of the target audience and the purpose of the generated stories to ensure that the stories meet the requirements and expectations of the audience.

+ Summarization:

The problem of text summarization is the task of condensing a given piece of text into a shorter version while preserving its key information and meaning. The goal is to create a summary that captures the most important points and main ideas of the original text, allowing readers to quickly grasp the content without having to read the entire document.

Text summarization can be categorized into two main types: extractive summarization and abstractive summarization.

Extractive Summarization: In extractive summarization, the task is to identify the most important sentences or phrases from the source text and extract them to form a summary. The extracted sentences are typically selected based on their relevance, significance, and redundancy. Extractive methods do not generate new sentences but instead rely on the existing text. They are often based on statistical algorithms or machine learning techniques that assign scores to sentences and select the top-ranked ones.

Abstractive Summarization: Abstractive summarization goes beyond the extraction of sentences and aims to generate a concise summary by understanding the meaning of the source text and expressing it in a new way. It involves natural language generation techniques to produce coherent and fluent summaries. Abstractive methods use advanced algorithms, such as deep learning models like recurrent neural networks (RNNs) or transformer models like GPT (Generative Pre-trained Transformer), to capture the semantics and context of the text and generate human-like summaries.

Text summarization is a challenging problem due to the complexities of language understanding, information extraction, and content generation. It requires the model to have a deep understanding of the text, identify key ideas, handle sentence-level and document-level coherence, and generate grammatically correct and contextually appropriate summaries.

Researchers and developers continuously work on improving text summarization techniques to enhance their accuracy, readability, and ability to handle various types of texts, such as news articles, research papers, legal documents, and social

media posts. The goal is to provide users with concise and informative summaries that save time and effort in information processing and decision-making.

2) Describe the theoretical models that will be applied to solve the problem

Multi-task Supervised Pre-training for Natural Language Generation (MVP) is a theoretical MVP approach that uses multi-task learning and pre-training techniques to improve the performance of natural language generation models. The main idea behind MSPG is to pre-train a neural network on multiple tasks related to natural language generation, such as machine translation, text summarization, and text generation, and then fine-tune the pre-trained model on a specific task.

- Architecture:

The architecture of Multi-task Supervised Pre-training for Natural Language Generation (MVP) models is based on a pre-trained encoder-decoder neural network with multiple heads, where each head is responsible for a specific task. During pre-training, the model is trained on multiple tasks related to natural language generation, such as machine translation, text summarization, and text generation. During fine-tuning, the model is fine-tuned on a specific task by adjusting the weights of the corresponding head while keeping the weights of the other heads fixed.

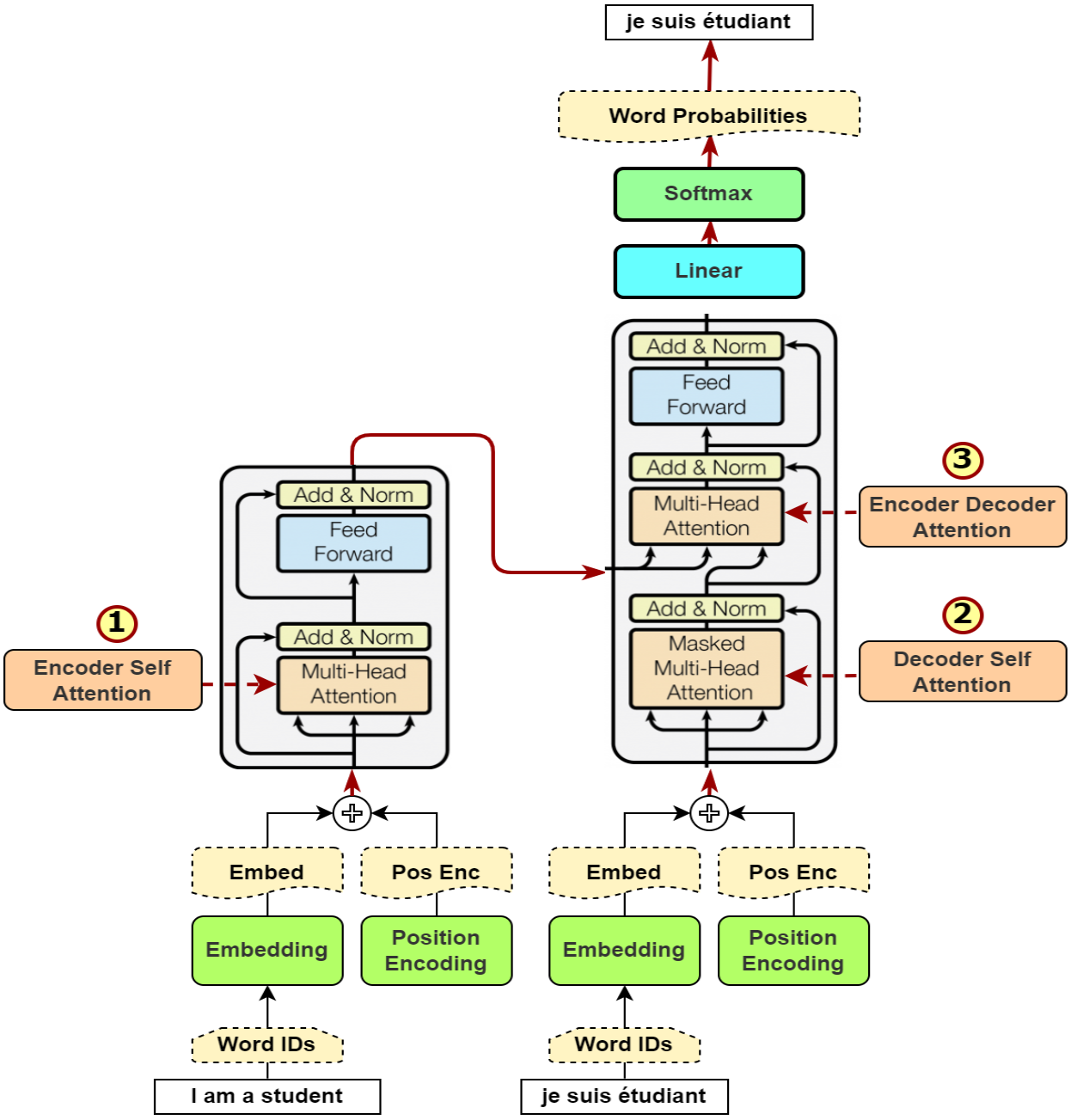
The pre-trained encoder-decoder model typically consists of an encoder network and a decoder network. The encoder network takes the input text and generates a fixed-length representation, or embedding, of the input. The decoder network takes the encoder output and generates the output text, word by word. During

pre-training, the model is trained to predict the next word or sequence of words in the input sentence for each task. This is done by using a combination of teacher forcing and sampling from the decoder output.

The multiple heads in the MVP architecture correspond to the different tasks that the model is trained on during pre-training. For example, one head may correspond to machine translation, while another head corresponds to text summarization. During fine-tuning, the weights of the corresponding head are adjusted while keeping the weights of the other heads fixed. This allows the model to learn task-specific representations without losing the generalization capabilities learned during pre-training.

The MVP architecture is similar to other pre-training and multi-task learning approaches, such as BERT and T5, but is specifically designed for natural language generation tasks, which require a different set of skills than natural language understanding tasks. The architecture is flexible and can be adapted to different types of natural language generation tasks by adjusting the specific pre-training tasks and fine-tuning procedures.

Multi-task Supervised Pre-training for Natural Language Generation (MVP) models can also incorporate attention mechanisms to improve their performance on natural language generation tasks. Attention allows the model to focus on specific parts of the input when generating the output, which can help to improve the quality and coherence of the generated text.



In the MVP architecture, attention can be applied in several ways. One common approach is to use self-attention, which allows the model to attend to different parts of the input text when generating the output. This can be particularly useful for

tasks such as text generation, where the model needs to generate a coherent and cohesive output based on a potentially long input sequence.

Another way to incorporate attention in the MVP architecture is to use cross-attention, where the model attends to different parts of the input and output sequences. This can be useful for tasks such as machine translation, where the model needs to generate a target sentence based on an input sentence in a different language.

The attention mechanism can be incorporated into the pre-training and fine-tuning procedures of MVP models by modifying the loss function to include a term that encourages the model to attend to relevant parts of the input. During fine-tuning, the attention weights can also be fine-tuned to further improve the model's performance on the specific task.

Overall, incorporating attention mechanisms into MVP models can help to improve their performance on natural language generation tasks by allowing the model to attend to relevant parts of the input and generate more coherent and cohesive output.

- **Characteristic:**

Multi-task learning: MVP models are pre-trained on multiple natural language generation tasks, such as machine translation, text summarization, and text generation. This allows the model to learn general language representations that can be fine-tuned for specific natural language generation tasks.

Supervised learning: MVP models are trained using supervised learning, where the model is trained to predict the correct output for a given input. During pre-training, the model is trained to predict the next word or sequence of words in the input sentence for each task. During fine-tuning, the model is fine-tuned on a specific task by adjusting the weights of the corresponding head while keeping the weights of the other heads fixed.

Pre-training and fine-tuning: MVP models are pre-trained on multiple tasks and then fine-tuned on a specific task. This allows the model to learn task-specific representations while retaining the generalization capabilities learned during pre-training.

Encoder-decoder architecture: MVP models typically use an encoder-decoder architecture with multiple heads. The encoder network generates a fixed-length representation of the input text, while the decoder network generates the output text, word by word.

Attention mechanism: MVP models can incorporate attention mechanisms to improve their performance on natural language generation tasks. Attention allows the model to focus on specific parts of the input when generating the output, which can help to improve the quality and coherence of the generated text.

Flexibility: MVP models are flexible and can be adapted to different types of natural language generation tasks by adjusting the specific pre-training tasks and fine-tuning procedures.

Overall, MVP models are characterized by their ability to learn general language representations through pre-training on multiple natural language generation tasks, while also being able to fine-tune on specific tasks to learn task-specific representations. The incorporation of attention mechanisms and the flexibility of the architecture make MVP models a promising approach for natural language generation tasks.

- **Pros and cons:**

Pros:

MVP models can leverage the benefits of pre-training to learn general language representations that can be fine-tuned for specific natural language generation tasks.

By pre-training on multiple tasks, MVP models can learn to capture different aspects of natural language, such as syntax, semantics, and discourse, which can improve their performance on a wide range of natural language generation tasks.

The flexibility of the MVP architecture allows for easy adaptation to different types of natural language generation tasks, making it a highly versatile approach.

The incorporation of attention mechanisms can improve the coherence and quality of the generated text.

MVP models have achieved state-of-the-art performance on several natural language generation tasks, including machine translation, text summarization, and text generation.

Cons:

Pre-training MVP models can be computationally expensive, requiring large amounts of data and computational resources.

The performance of MVP models may be task-specific, and their performance on a specific task may not generalize well to other tasks.

The architecture of MVP models can be complex and may require specialized expertise to implement and optimize.

MSPG models may require significant fine-tuning on specific tasks to achieve optimal performance, which can be time-consuming and resource-intensive.

Overall, MVP models have several advantages for natural language generation tasks, including their ability to leverage pre-training to learn general language representations and their flexibility to adapt to different types of tasks. However, they also have some limitations, such as their computational requirements and the need for significant fine-tuning to achieve optimal performance on specific tasks.

- **Compare with other models:**

+ vs BERT

Multi-task Supervised Pre-training for Natural Language Generation (MVP) models and BERT models are both pre-trained natural language processing models, but they differ in their specific architectures and intended applications.

BERT models are pre-trained on a masked language modeling task, where a certain percentage of words in the input sequence are masked, and the model is trained to

predict the correct word based on the context provided by the surrounding words. BERT models have achieved state-of-the-art performance on a wide range of natural language processing tasks, including sentiment analysis, named entity recognition, and question answering.

In contrast, MVP models are pre-trained on multiple natural language generation tasks, such as machine translation, text summarization, and text generation. The intention of MVP is to learn general language representations that can be fine-tuned for specific natural language generation tasks. MVP models can also incorporate attention mechanisms to improve their performance on natural language generation tasks.

The main differences between MVP models and BERT models are:

Task-specific vs. task-agnostic pre-training: BERT models are pre-trained on a single task (masked language modeling), while MVP models are pre-trained on multiple tasks, some of which are natural language generation tasks.

Architecture: BERT models use a transformer encoder architecture, while MVP models use an encoder-decoder architecture with multiple heads.

Intended applications: BERT models are intended for a wide range of natural language processing tasks, while MVP models are specifically intended for natural language generation tasks.

In terms of performance, MVP models have achieved state-of-the-art results on a variety of natural language generation tasks, including machine translation, text summarization, and text generation. However, BERT models are still considered the gold standard for many natural language processing tasks, and their performance has been widely tested and validated across different domains and languages.

In summary, while both MVP models and BERT models are pre-trained natural language processing models, they differ in their specific architectures, intended applications, and pre-training tasks. MVP models are specifically designed for natural language generation tasks and have shown promising results, while BERT models are widely used for a variety of natural language processing tasks and are considered the benchmark for many of these tasks.

+ vs GPT-2:

Multi-task Supervised Pre-training for Natural Language Generation (MVP) models and GPT-2 models are both pre-trained natural language processing models, but they differ in their specific architectures, pre-training tasks, and intended applications.

GPT-2 models are pre-trained on a large corpus of text using a language modeling objective, where the model is trained to predict the next word in a sentence given the previous words. The goal of GPT-2 models is to learn general language representations that can be fine-tuned for a variety of natural language processing tasks, including text classification, question answering, and text generation.

In contrast, MVP models are pre-trained on multiple natural language generation tasks, such as machine translation, text summarization, and text generation. The goal of MVP is to learn general language representations that can be fine-tuned for specific natural language generation tasks. MVP models can also incorporate attention mechanisms to improve their performance on natural language generation tasks.

The main differences between MVP models and GPT-2 models are:

Pre-training tasks: GPT-2 models are pre-trained on a language modeling task, while MSPG models are pre-trained on multiple natural language generation tasks.

Architecture: GPT-2 models use a transformer decoder architecture, while MVP models use an encoder-decoder architecture with multiple heads.

Intended applications: GPT-2 models are intended for a variety of natural language processing tasks, while MVP models are specifically designed for natural language generation tasks.

In terms of performance, both MVP models and GPT-2 models have achieved state-of-the-art results on natural language processing tasks. However, GPT-2 models have been shown to perform particularly well on text generation tasks, such as generating coherent and diverse sentences and paragraphs. MVP models have also shown promising results on text generation tasks, but their main advantage lies in their ability to incorporate multiple natural language generation tasks during pre-training.

In summary, MVP models and GPT-2 models are both pre-trained natural language processing models, but they differ in their specific architectures, pre-training tasks, and intended applications. MVP models are designed for natural language generation tasks and can incorporate multiple tasks during pre-training, while GPT-2 models are intended for a variety of natural language processing tasks and have shown particular strength in text generation tasks.

+ **vs T5:**

Multi-task Supervised Pre-training for Natural Language Generation (MVP) models and T5 (Text-to-Text Transfer Transformer) models are both pre-trained natural language processing models, but they differ in their specific architectures, pre-training tasks, and intended applications.

T5 models are pre-trained on a text-to-text transfer learning task, where the model is trained to convert input text from one format to another, such as translating text

from one language to another or summarizing long pieces of text. The goal of T5 models is to learn general language representations that can be fine-tuned for a variety of natural language processing tasks, including text classification, question answering, and text generation.

In contrast, MVP models are pre-trained on multiple natural language generation tasks, such as machine translation, text summarization, and text generation. The goal of MVP is to learn general language representations that can be fine-tuned for specific natural language generation tasks. MVP models can also incorporate attention mechanisms to improve their performance on natural language generation tasks.

The main differences between MVP models and T5 models are:

Pre-training tasks: T5 models are pre-trained on a text-to-text transfer learning task, while MVP models are pre-trained on multiple natural language generation tasks.

Architecture: T5 models use a transformer encoder-decoder architecture, while MVP models use an encoder-decoder architecture with multiple heads.

Intended applications: T5 models are intended for a variety of natural language processing tasks, while MVP models are specifically designed for natural language generation tasks.

In terms of performance, both MVP models and T5 models have achieved state-of-the-art results on natural language processing tasks. However, T5 models have shown particular strength in text-to-text transfer learning tasks, such as machine translation and text summarization, while MVP models are designed specifically for natural language generation tasks and can incorporate multiple tasks during pre-training.

In summary, MVP models and T5 models are both pre-trained natural language processing models, but they differ in their specific architectures, pre-training tasks, and intended applications. MVP models are designed for natural language generation tasks and can incorporate multiple tasks during pre-training, while T5 models are intended for a variety of natural language processing tasks and have shown particular strength in text-to-text transfer learning tasks.

- **Information**

Pre-trained language models (PLMs) have achieved remarkable success in natural language generation (NLG) tasks. Up to now, most NLG-oriented PLMs are pre-trained in an unsupervised manner using the large-scale general corpus. In the meanwhile, an increasing number of models pre-trained with labeled data (i.e., ``supervised pre-training'') showcase superior performance compared to unsupervised pre-trained models. Motivated by the success of supervised pre-training, we propose Multi-task superVised Pre-training~(MVP) for natural language generation. We collect a large-scale natural language generation corpus, MVPCorpus, from 77 datasets over 11 diverse NLG tasks. Then we unify these examples into a general text-to-text format to pre-train the text generation model MVP in a supervised manner. For each task, we further pre-train specific soft prompts to stimulate the model's capacity to perform a specific task. Extensive experiments have demonstrated the effectiveness and generality of our MVP model in a number of NLG tasks, which achieves state-of-the-art performance on 13 out of 17 datasets.

Submission history

From: Tianyi Tang

[v1] Fri, 24 Jun 2022 07:49:47 UTC (235 KB)

[v2] Mon, 19 Dec 2022 11:44:38 UTC (235 KB)

3) Experimental results.

- **Dataset:**

+WritingPrompts:

The Writing Prompt dataset is a collection of sentences, requests, prompts, or topics used to initiate text generation. This dataset is used to train text generation models to create stories, paragraphs, essays, news articles, and other content based on the provided prompts.

Writing Prompt datasets are usually collected from various sources, including writing competitions, websites that generate prompts, and popular topics in society, politics, sports, science, culture, and the arts. To ensure diversity and richness in the prompts, the dataset is typically collected from multiple sources and processed to remove duplicates and inappropriate prompts.

The use of the Writing Prompt dataset depends on the purpose and type of text generation model used. However, a common use is to train text generation models to create paragraphs, stories, essays, news articles, and other content based on the provided prompts. Once trained, these models can be used to automatically generate content for various needs, such as generating content for a website or producing automatic content for other products.

+ **XSum:**

The XSum dataset is a large-scale dataset specifically designed for the task of single-document summarization. It was introduced by Donahue et al. in 2020 and has since become widely used in natural language processing (NLP) research, particularly in the area of text summarization.

The goal of the XSum dataset is to generate a concise summary given a longer document. The dataset consists of approximately 226,711 news articles, collected from various online sources. Each article is accompanied by a single-sentence summary, which is created by professional human summarizers.

The articles in the XSum dataset cover a wide range of topics, including news, opinion pieces, blogs, and editorials. They represent diverse domains such as politics, sports, technology, entertainment, and more. The dataset is designed to capture the challenges of real-world summarization tasks by encompassing a variety of writing styles, document lengths, and summarization requirements.

XSum provides a valuable resource for training and evaluating text summarization models. It enables researchers to explore different approaches to summarization, including extractive and abstractive techniques. Extractive summarization involves selecting important sentences or phrases from the source document, while abstractive summarization aims to generate a summary by paraphrasing and rephrasing the content in a more human-like manner.

The XSum dataset has a hierarchical structure, with each article-summary pair treated as a single example. This allows models to learn the correspondence between the source document and its corresponding summary. The dataset is commonly split into training, validation, and test sets, following standard evaluation protocols in NLP research.

Due to its large size and the high quality of the human-written summaries, the XSum dataset has become an essential benchmark for evaluating summarization models. It has facilitated advancements in the field, leading to the development of more accurate and effective techniques for automatic text summarization.

In summary, the XSum dataset is a valuable resource for single-document summarization research. It provides a diverse collection of news articles along with human-written summaries, enabling the development and evaluation of advanced text summarization models.

REFERENCES

https://github.com/google-research/text-to-text-transfer-transformer

https://huggingface.co/docs/transformers/model_doc/mvp