Evaluation of Transformer Models on Summarization and Beyond

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Computer Science and Engineering** by Research*

by

Pawan Sasanka Ammanamanchi
2020701028
pawan.sasanka@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled 'Evaluation of Transformers on Summarization and Beyond' by Pawan Sasanka Ammanamanchi, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Manish Shrivastava

To Anchita who inspired me to keep going

Acknowledgments

To my Mom and Dad, who asked me to pursue this Master's and were patient and understanding with me while I completed the degree.

To Manish Sir, who was all I could ask for in a guide and mentor, who taught me the right ways of pursuing research, the tricks of the trade and pushed me enough, and expected more out of me than I possibly could have imagined making me the researcher that I am today, and also for inculcating the right mindset to pursue research.

To my love of Deep learning and Natural Language Processing for giving me purpose. To this long journey of years doing research and everything that comes with it. To IIIT Hyderabad for showing me that suitable environments can create magical talent and foster a deep culture of research and technical aptitude. To the intriguing challenges of Summarization as an NLP complete task, the holy grail which cannot be sought and only thought about. To Lee Sedol and Alphago, who inspired me to begin my journey in Artificial intelligence.

To my brother, who asked me to pursue research, saw in me the potential to complete it and inspired me to embark on this path. To Abhay, Anando, and Satyajeet for supporting me throughout my degree. To Siddharth Bhat for making me curious and believing you can learn everything you wish. Finally, To Anchita for being a constant, unwavering source of support, inspiring me to complete what I began, making me believe I could achieve things when I thought I couldn't have, and that there's no end to learning, something I'll be forever grateful for.

Abstract

> You shall know a word by the company it keeps.
>
> J.R. Firth

Transformer models, due to their self-attention architecture, ease of training, and high parallelization capabilities, transformer models have taken the field of deep learning by storm. They have been applied in Natural Language Processing, Computer Vision, Speech Recognition, Protein Folding, Reinforcement Learning, and other intersections of deep learning and sub-fields within artificial intelligence and beyond.

Deep Learning models in Natural language processing, especially transformer models, have given rise to skillful and well-performing language models. Recently, one such application called ChatGPT has seen 100 million users within three months of launch because of its user-friendly interface and wide-ranging task-solving capabilities through prompting, finetuning and other techniques.

We aim to investigate the capabilities of transformer language models and understand the nuances of evaluating them. We focus on applying a pre-trained language model to the summarization task and trying to extend its capabilities through finetuning. These models can have their abilities augmented and extended to solve tasks in various domains. We then look at developing a generation benchmark. Developing a benchmark gives us insight into where the field was at and where the area is going and the choices that must be made to create a benchmark that meets the needs of an ever-encompassing fast-moving natural language processing landscape and even then be challenging enough for language models so that it provides us some insight into their nature. Next, we turn our attention toward LLMs to appreciate the capabilities of language modeling transformers and their scalable nature. We see the development of BLOOM, a 176B parameter multilingual language model trained on 46 languages. We evaluate its 0-shot and 1-shot performance on various tasks to understand what one can expect from a massively trained large language model.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> Somewhere, something incredible is waiting to be known.
>
> Carl Sagan

Language is the foundation of human communication and culture, serving as a bridge between individuals and societies. Natural Language Processing (NLP) is the interdisciplinary study and development of algorithms and techniques that enable computers to process, i.e., understand and generate human language. NLP draws from linguistics, computer science, and artificial intelligence to address many tasks, including question-answering, summarization, reading comprehension, and more.

Large language models like GPT3, Chatgpt, and GPT4 in the past few months have seen an absurd amount of usage, and the adoption of Natural language processing models has seen growth hitherto unknown, as Chatgpt saw 100 million users within three months of its launch.

While Intelligent Assistants first showed up in popular culture in the form of JARVIS in Iron Man Comics. The first NLP system that showed some semblance of natural language processing was ELIZA, a chatbot created through pattern-matching rules. Since then, the field has moved through various ways to process language on computational systems, like regular expressions for file matching and searching within documents. Also, Tokenization and Lemmatization helped to understand documents better and draw insights about the text they contain, leading to algorithms like tf-idf and other information retrieval algorithms.

Statistical language modeling with Hidden Markov Models (HMM) laid the groundwork for predicting words based on their predecessors, increasing the usage of Markov assumption in language modeling more and more. However, the true potential of language modeling was realized with the introduction of vector-based word representations by Geoffrey Hinton in

the late 1980s, followed by architectural advances like Recurrent Neural Networks (RNNs) in 1982 and Long Short-Term Memory (LSTM) networks in the 1990s. The first neural language model, introduced by Hinton et al. in 2003, paved the way for further innovations like the Word2Vec and Glove embeddings based on the distributional hypothesis, which was followed by adding attention to Neural sequence-to-sequence model, ultimately leading to the development of the Transformer architecture, an architecture based on self-attention. This improvement ultimately paved the way to better contextual language modeling through pre-trained language models and is the foundation of Large language models today.

1.1 Motivation

When Word2Vec [63] and subsequently Glove embeddings were introduced, evaluating these models was divided in two ways, Intrinsic and Extrinsic evaluation. Intrinsic evaluation looked at the quality of the embeddings and language representations the model had developed, while extrinsic evaluation tried to apply these tasks to a downstream task. The first intrinsic evaluation seen in these embeddings was done by making word analogies of the form "King:man:: Queen:?" and then the WordSim323 to test relatedness in words. Extrinsic evaluation was achieved by applying these embeddings to Named Entity Recognition tasks.

As language models evolved and sequence-to-sequence models found popularity, these were directly trained and applied to end tasks. The standard evaluation metric for Language models was perplexity. It essentially tried to determine how the language model had learned the distribution of the text it was trained on and if it could predict the words well. Later, Attention [3] was added to RNNs and evaluated on the WMT14 fr-en task showing immense gains and finding popularity in other tasks. RNNs were quickly replaced by LSTMs to solve certain issues like vanishing gradients in the architecture. These found usage in predictive typing, speech and handwriting recognition, and natural language tasks like Machine translation and summarization.

Transformers [99] was introduced as an attention-driven architecture and also was implemented to take advantage of parallelization capabilities due to GPUs. Due to the data-hungry nature of these models, alternative paradigms for language modeling were explored, like developing them with pretraining objectives, which took advantage of the data and developed denser representations. BERT [22], GPT-2 [79] were general-purpose transformer language models with wide-ranging capabilities.

Now that we have modeled language through models, the natural question is, what features do we expect from a model? Are the features good? Do they universally apply to the tasks that we are interested in? If the model performs well on a dataset, then does the model generalize to all such problems across domains and task types? Evaluating language models became a multi-dimensional problem that sought answers not only to the internal workings

of the models but their applicability to external tasks. Their slow but steady improvement led to the development of sophisticated benchmarks, which evaluated not just a single task in a single domain but tried to understand all capabilities holistically.

Language models have seen rapid development due to scaling laws [43], and monumental leaps in their abilities were thus established. Models could now be directly applied to the task by simply prompting them, "What is the answer to the following question?" "What is another way of understanding newtons 3 laws?" etc. Emergent abilities slowly crept into larger models which weren't observed in their smaller counterparts. The most significant ability in language models today is to seamlessly generate language when queried in natural language as if speaking to another human being but one endowed with the vast riches of knowledge and an ability to understand it. Language models are becoming increasingly prevalent in real-world applications; therefore, it is crucial to understand their capabilities, limitations, and generalizability across various tasks and domains. In-depth capability studies are essential to ensure these models' responsible and effective deployment in diverse settings.

1.2 Summary of the Thesis

This thesis explores evaluating and assessing modern natural language processing models, explicitly focusing on transformers and large language models.

In this thesis, we first look at a pre-trained language, namely BERT, a model trained with pretraining objectives and based on transformer architecture. We apply BERT and understand its ability by applying it to a task. The task of choice is summarization, which is writing short, concise summaries for documents. We apply BERT to a dataset long-form document summarization dataset in the scientific domain called SciTldr. Although our training methods yield performance gains, we realize that language models need better evaluation methods and more language comprehension and understanding abilities to process documents and get better at summarization tasks. A better way of evaluating would help understand what the model is good at and what the model is bad at thus helping understand what parts model improvement can address. Better Language understanding abilities are possible through better models.

We then focus on looking at evaluation criteria in NLP and how it has evolved our the years. This thesis then covers GEM, a natural language generation benchmark that seeks to evaluate the generation capabilities of large language models holistically and was designed in the wake of models like BART [51], T5 [80] and GPT-3 [11]. Not only do we design the benchmark to cover task variety, we also evaluate the models through different automatic and human evaluation criteria, which paint a nuanced picture of model ability.

We finally move towards another direction of improving models and their abilities by looking at BLOOM. BLOOM is a 175B multilingual, large language model trained over 49

languages and 350 billion tokens of text and is the largest multilingual model that is open source. We walk through the data collection and training aspects before evaluating the model. Large language models are evaluated with in-context learning, an ability that improves as the model scales. To establish nuance, see its abilities to their full extent, and shed more light, we compare the model against other models evaluated similarly.

Our findings can be summarized as follows:

1. Intermediate fine-tuning helps pre-trained language models like BERT improve performance on out-of-domain tasks.

2. The development of GEM, a benchmark for evaluating language models in generation tasks.

3. The evaluation of Bloom, a 175B multilingual language model, and its comparison with various other language models.

1.3 Thesis Outline

In Chapter 2, We first provide an overview of the Transformer architecture and the objectives used in training pre-trained language models.

Next, in Chapter 3, we investigate the performance of BERT, a particular instantiation of the Transformer, in the extractive summarization of scientific documents and the impact of intermediate fine-tuning. This chapter explores domain adaptation in language models and whether language models could be augmented or trained further to work in other domains, suggesting that they aren't universal.

In Chapter 4, We then examine the development of a new benchmark, GEM, for evaluating language models in generation tasks. The foundation of this benchmark seeks to assess a model's multitudinous language generation abilities, which involve not only task solving but internal abilities like understanding complex inputs like tables and documents and others like multilingual translation, long-form comprehension, etc.

Finally, in Chapter 5, we analyze the large, multilingual language model, Bloom. Bloom is the largest open-source, multilingual model in existence to date, and we assess it by comparing its performance with other state-of-the-art models on a wide range of tasks.

In Chapter 6, we discuss the conclusion and also discuss potential future work that can be pursued.

Chapter 2

Background

> Nothing in life is to be feared, it is only to be understood. Now is the time to understand more, so that we may fear less.
>
> Marie Curie

In this chapter, We will look at the original transformer architecture and develop some intuition about how it works. We will also study the background of language modeling and get up to speed on where it is today.

While Recurrent neural networks (RNNs) and Long short-term memory (Long short-term memory) architectures were popular and useful for the sequence to sequence language modeling, the paper on applying attention to models [3] showed the effectiveness of attention in processing text and modeling language. Several improvements were made, and every task-specific architecture involved using different kinds of attention to improve performance on the target dataset/task.

Attention mechanisms being intuitive also became the bread and butter in neural sequence-to-sequence architectures but were all used with a recurrent network. Transformers attempted to introduce a new architecture with the self-attention mechanism at its heart, with practical parallelization features that would also help them take advantage of GPUs.

2.1 Transformers

The transformer architecture was introduced in [99]. It can be seen in 2.1. Most Neural sequence-to-sequence models have an encoder-decoder structure. The transformer architec-

ture uses stacked self-attention and fully connected layers to complete the architecture. We will go over the layer now.

2.1.1 Input

The Input (tokenization) and embedding look as follows, first the input text is split into pieces:

```
"The human investigates" -> [The_] [human_] [invest] [igat] [es_]
```

The tokens are then indexed into a vocabulary:

```
[The_] [human_] [invest] [igat] [es_] -> [3 721 66 3434 12]
```

Then each vocab entry learns a d_{model} -dimensional vector

```
[3 721 66 3434 12] ->
[[0.123,.0232,.....],[],[], .... vocab_size elements]
```

Attention is invariant with the position of the token in the sequence. We need to imbue this with some notion of sequences. Position embeddings are added to the input token sequences. We can add any periodic wave. The paper chooses sine and cosine functions of different frequencies.

2.1.2 Self Attention

Intuitively this can be understood as if each token in a sequence can see the whole input sequence and then decide to update its own representation based on what it sees because some words influence the presence of a word more than others. In other words, Self-attention is a way in which the model generates scores that indicate the importance of each token to every other token. Another way to understand self-attention is to think of the self-attention matrix that the transformer creates. Its input size is $batch_{size}$ x sequence_length x $embedding_size$. The sequence of tokens is embedding and copied three times to create the "query," "keys," and "values" vector. Each Matrix goes through the linear layer. The Q and K matrices are split into multiple parts. Attention allows for long-range dependencies in a sequence-to-sequence network

A self-attention layer will map an input with the sequence $(x_1, x_2, ...x_n)$ to the output with the sequence $(y_1, y_2, ...y_n)$. As discussed earlier, the idea of self-attention is for every token to look at every other token in the sequence and reveal/capture each other's relevance in the contexts.

Let us assume the dot product between two vectors x_i, x_j, which is the simplest way of comparison. We get

$$score(x_i, x_j) = x_i.x_j \tag{2.1}$$

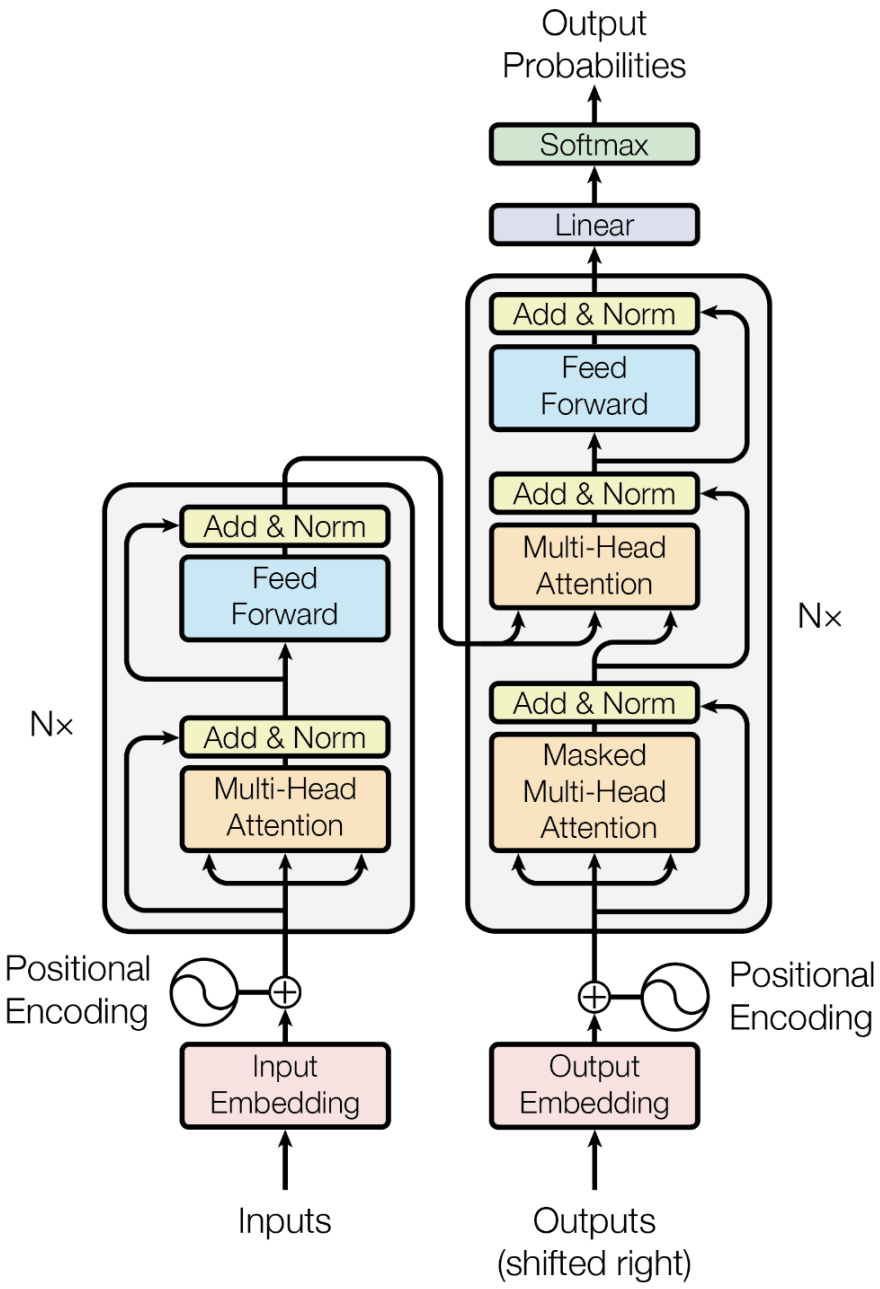

Figure 2.1 Transformer Architecture

The result of this is a number that is a scalar, so we will need to normalize this so it doesn't go out of bounds and become meaningless. We do that by collecting them to create an attention vector α_{ij}:

$$\begin{aligned}\alpha_{ij} &= softmax(score(x_i, x_j)) \\ &= \frac{exp(score(x_i, x_j)}{\sum_{k=1}^{i} exp(score(x_i, x_k))} \\ &\qquad\qquad \forall j \leqslant i\end{aligned}$$

Then the final y_i values are generated by taking the input sequence and multiplying by the attention vector:

$$y_i = \sum_{j \leqslant i} \alpha_{ij} x_j \tag{2.2}$$

The steps above ultimately are similar to any attention-based approach. We first have two sequences to compare in some context, we normalize the scores to get a probability distribution, and then we use a weighted sum of this distribution. Finally, the output is given by a computation of the weighted sum over the original input sequence.

In the case of transformers, it is slightly different because we have three different things we pay attention to. After looking at all other tokens, the query represents the sequence we are calculating attention for. The key vector represents the sequence before the current one when compared to the current one, and the value is what we get after computing the current focus of attention. We calculate these vectors by

$$q_i = W^Q x_i; k_i = W^K x_i; v_i = W^V x_i \tag{2.3}$$

Where W^Q,W^K,W^V are the weight matrices of the dimensions $\mathbb{R}^{d x d}$ where d is the dimension of the embeddings, the original paper has d_model as 1024.

The dot product attention in the case of transformers is calculated as follows:

$$score(x_i, x_j) = \frac{q_i . k_j}{\sqrt{d_k}} \tag{2.4}$$

The above equation is called the scaled dot product because it is divided by a factor of the size of embeddings. The beauty of the transformer model is that while we showed in equations till now that we consider one sequence at a time, we can parallelize this by taking N tokens at once in a single matrix $X \in \mathbb{R}^{N x d}$. This processing power gives rise to Multiheaded self-attention, which means the input sequence is used to create queries, keys, and values, and each token can look at the whole Input and update its representation based on what it sees.

2.1.3 Transformer block

Now that we have understood the various parts of the transformer stack, we look at the entire block at once.

It has a Self-attention layer, then a normalization layer, then fed into a feedforward layer, and then again layer normalization. A residual connection is also added between the original sequence and after the self-attention and feedforward layer.

We can look at this entire block in the form of two equations:

$$z = \mathrm{LayerNorm}(x + \mathrm{SelfAttention}(x))$$
$$y = \mathrm{LayerNorm}(z + \mathrm{FFN}(z))$$

Layer normalization is done to improve training and keep it stable, which is the same reason for keeping residual connections. Following ResNets in computer vision, the module computes a residual instead of a new value. Residual connections dramatically improve trainability.

The MLP is applied to each token individually as follows.

$$z_i = W_2 \mathrm{GeLU}(W_1 x + b_1) + b_2 \tag{2.5}$$

This is where the bulk of parameters in a transformer resides. It can also be thought of as a 1x1 convolution.

The encoder in the original model has 6 or 12 layers stacked on each other, and bigger language models have even more. The Decoder in the original model also has 6 or 12 layers and can have a higher number of layers. Once we have the output distribution, we can sample using greedy or beam search or any other sampling technique.

The transformer model can be used in 3 ways:

1. EDT - Encoder Decoder transformer

2. Only Encoder

3. Only Decoder

Each of these has its own advantages and disadvantages and has been applied in the space accordingly. Decoder-only Transformers are the most popular kind of transformers right now and make up the majority of large language models.

Transformers have been applied in a bunch of other sub-fields of deep learning, including Computer Vision in the form of ViT(Vision Transformers) [24] where the image is thought of patches and then fed into the transformer. Also, in Speech recognition in the form of Conformer [33], in Reinforcement learning in the form of Decision Transformer [15].

2.2 A brief history of language modeling

Language modeling is a straightforward task of modeling the probability of a sequence of tokens in a text. The joint probability of tokens in a text is represented as:

$$p(x) = p(x_1, x_2, ...x_n) = \prod_{t=1}^{n} p(x_t|x_{<t}) \tag{2.6}$$

Where n is the number of tokens in the text and $x_{<t}$ represents the sequence of tokens. This approach is also known as autoregressive language modeling.

2.2.1 Word2vec & Glove

Neural language models were originally introduced in 2001 for practical applications like keyboards, autocorrelation, etc. The original Neural language model was a feedforward network introduced by [8]. The idea was to represent tokens in vectors and the form of previous words. These are known as word embeddings today.

The word2vec model introduced in [63] paved the way for denser word embeddings. The idea was based on the distributional hypothesis and is an unsupervised word embedding approach. The model involves using a neural model to learn word associations from bodies of text. There were two types of objectives: a continuous bag of words and skip-gram.

Word embeddings continue to be a part of NLP because they have shown that models perform much better in downstream tasks when initialized with word embeddings. After all, word embeddings capture a lot of information and nuance.

Glove [73] subsequently came out to show even richer representations by incorporating statistical information of words into the model.

2.2.2 ELMO

While word embeddings like Word2vec and glove were very good, they left much to be desired and were static embeddings. They did not capture nuances like the context of the text. For example, the word "I am going to murder you!!!" and "a murder of crows" while has the same word. Murder means to "kill you" (jokingly) in the former, while it means to represent a collection of crows in the latter.

To capture these semantic and syntactic nuances, ELMO [75] was introduced, which was a bidirectional language model, using the autoregressive language model objective not only in the forward direction of text but also in the reverse direction of the same. This formulation gave the model enough information and helped capture the context of the word in the embeddings.

2.2.3 BERT

While the ELMO model captured contextual embeddings through the use of an autoregressive language modeling model applied both ways through the biLSTM model, the BERT model tried to capture contextual nuances of text through transformers. BERT stands for Bidirectional Encoder Representations for Transformers.

The BERT model is an encoder-only transformer model. It tries to learn these nuances by masking some of the input text and trying to predict it. It creates two primary objectives for training on data with transformers:

1. MLM: Masked Language modeling
2. NSP: Next Sentence Prediction

The MLM pretraining objective takes advantage of the fact that transformers essentially encode attention on all words when a sequence is applied to them. The way it is trained is that in any sequence, it masks about 15% of the words and has the model predict these masked words, thus gaining context into "whats a valid word" that can fit this missing space in a sentence thus developing contextual embeddings.

The Next sentence prediction objective is to model Question answering and Natural language understanding and is about capturing the relation between two sentences. Given two sentences, A and B, the model must predict whether sentence B comes after A.

The BERT model does a bunch of ablations to show the efficacy of these objectives and that they genuinely capture contextual representations.

There have been a variety of follow-up work since BERT in trying to improve pretraining objectives in encoder-only transformer models:

- RoBERTa [60] optimizing for design choices in BERT and training for longer with a more extensive dataset, and they also remove the NSP objective because they find in ablation studies that at scale removing the NSP objective has no substantial gains on the training of the model.
- AlBERT [47] incorporates parameter reduction techniques to reduce the size and improve the scaling of the models. They also improve performance over BERT by replacing NSP with an inter-sentence coherence loss.
- ELECTRA [18] finds that better pretraining objectives might be applied to improve the performance of encoder transformer models. They suggest the replaced token detection objective, where an MLM is used to predict masked words in a sentence, and then a discriminator is trained to compare the original unmasked sentence and the sentence generated. The discriminator has to predict which of the words was replaced or not.

This paper shows substantial improvements over BERT with this and also makes it more data efficient than BERT.

- DeBERTa improves on BERT and roBERTa by a disentangled attention mechanism and an enhanced mask decoder. In a subsequent improvement of this model called DeBERTa v3 [36] they replace the MLM objective with the replaced token detection objective and show further gains, Thus cementing the place of the ELECTRA objective for pre-trained language modeling.

2.2.4 GPT2 and the other large language models

While BERT and other encoder-only models showed that contextual embedding transformers could be used to initialize task-specific architectures and finetuned further for gaining performance at the same time, GPT-2 was one of the first works to increase the model size of decoder-only transformer models.

The objective of GPT-2 was the same core language modeling objective, and they hypothesize that transformer models are better at computing these conditional probabilities. They show consistent zero-shot capability improvement on a bunch of datasets.

GPT-2 propelled OpenAI to train GPT-3 a large language model at the 175B parameter scale, which has shown powerful capabilities by just the simple autoregressive language modeling objective and also the fact that these scaled language models had a lot of out-of-box performance without any need for finetuning, in the form of In-Context learning.

Since then, various other large language models have been trained and more results have been further established to show that decoder-only models might be better at scale because of these capabilities.

2.3 Conclusion

In this chapter, We see a very brief background of transformer architecture and the language modeling era to understand and develop some nuance about how the field of language modeling has shaped and also changed due to transformers. Next, We see the application of an encoder-only pre-trained language model BERT to a domain-specific summarization dataset.

Chapter 3

Extractive Summarization on Scientific Documents

> We can only see a short distance ahead, but we can see plenty there that needs to be done.
>
> Alan Turing

Automatic text summarization is the task of writing concise summaries of big text documents. Summarization is an attempt to address the humongous and continually increasing amount of textual data available on the internet to both help in discovering relevant info and to consume important info in a quicker way. The summary thus generated has to be concise, relevant, grammatically sound, and factually faithful to the source document, i.e., the details in the document should be represented correctly.

There have been multiple attempts to summarize automatically including rule-based systems and extractive systems which did heuristic searching with the help of tf-idf and then tried to write a summary with the same. These attempts although they saw some success it was only modest in nature because of the inability to process large documents and were limited to only smaller documents.

With the rise of Neural Natural language processing, various tasks across the field saw a natural boost because of the ability to train models with better architectures and interpret results as well. These gains were evident in Question Answering, Machine translation, and other semantic tasks like POS-tagging and Named entity recognition.

There are two types of Summarization noted below:

- Extractive Summarization: These approaches heavily derive from the document that is trying to be summarized. In this, we try to pick the sentences which maximize the coverage/content presented in the document and present them as a summary. Although

they might suffer from issues like coherence since the "extracted" sentences are disjoint. You can think of this approach as akin to a highlighter. In this approach, summaries are formed by copying and concatenating the most important spans(usually sentences).

- Abstractive Summarization: These approaches try to model what we do as humans. In this, we trying to use novel language-generating techniques to write sentences i.e. trying to comprehend what the document means and then presenting the summary in a more coherent way. You can think of this approach as akin to a pen. In this approach, target summaries contain words or phrases that were not in the original text and usually require various text rewriting operations to generate.

The Neural era in Summarization started with the introduction of the CNN-Daily Mail Dataset for Summarization introduced in [65]. The dataset was originally introduced in [37] as a Question-Answering Dataset and was repurposed to a Summarization task in [65] which was an anonymized version. CNN-Daily Mail was the first large-scale summarization dataset with multi-line summaries. The dataset is a news Summarization dataset with a document's average size of around 800 tokens and an average summary size as 60 tokens. Since then several different kinds of datasets for summarization have been released including opinion summarization, dialogue summarization, long-form summarization, domain-specific summarization benchmarks, multilingual summarization datasets, etc.

Dataset Split	Number of instances in Split
Training	287,113
Validation	13,368
Test	11,490

Table 3.1 Dataset Split and Size for CNN/DailyMail

3.1 Early Neural Summarization Work

3.1.1 Pointer generator and Reinforcement learning

In [65], Sequence to sequence networks was used to solve the abstractive summarization problem. Despite various tricks involving attention, large vocabulary, and a switching pointer-generator for unseen words, some issues became evident when using a simple sequence-to-sequence model for summarization. These were:

- Coverage: Vanilla Sequence to sequence models were not covering the entire document while generating the summary.

Text	"BOLINGBROOK, Illinois (CNN) – The disappearance of a suburban Chicago police sergeant's wife is now being treated as a potential homicide, and her husband is a suspect, authorities said Friday. Stacy Peterson, 23, has been missing from her suburban Chicago home since October 28. In another development, a judge signed an order to exhume the body of Drew Peterson's third wife, who was found drowned in a bathtub in 2004, said Will County State Attorney James Glasgow. Peterson, 53, said he last spoke to 23-year-old Stacy Peterson – his fourth wife – the night of October 28. Drew Peterson initially told the media he believed his wife ran off with another man, but he hasn't repeated that accusation. CNN has been unable to contact Drew Peterson for comment. The couple have been married four years and have two children, who have been interviewed for the investigation, Glasgow said. Drew Peterson also has older children from a previous marriage. Investigators have twice searched the couple's home and vehicles, and removed several items, including computers, said Illinois State Police Capt. Carl Dobrich....."
Summary	"NEW: Judge signs order to exhume the body of Drew Peterson's third wife. Peterson has said he believed his fourth wife left him for another man. Police: Case shifts from a missing persons search to a potential homicide. Friends and family: Stacy Peterson expressed concerns about her husband ."

Table 3.2 A summarization example from the CNN/DailyMail Dataset

- Repeating words: Earlier Language models were prone to succumb to neural degeneration i.e. when sampling words from the model it eventually tends to just repeat the word.
- Factual Details & OOV issue: The model was not able to handle factual details and out-of-vocabulary (OOV) words.

The pointer-generator architecture in [89] (As shown in Figure:3.1) was proposed as an answer to the issues faced by earlier summarization works. The paper solved the factual

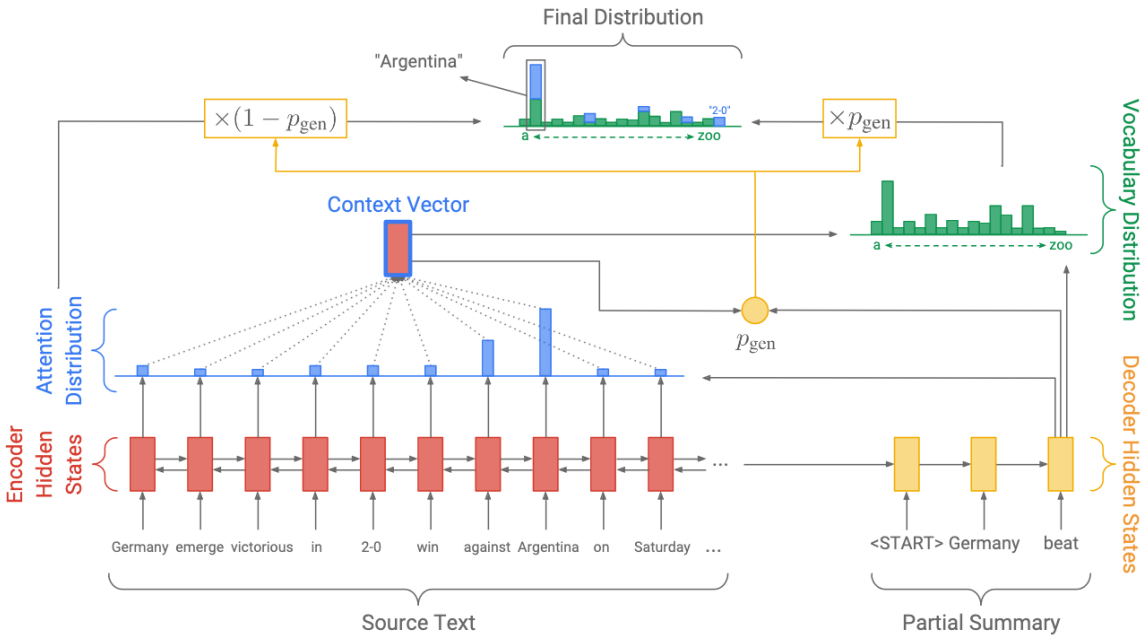

Figure 3.1 The pointer generator architecture

details and out-of-vocabulary word issue by having a probability specifically for generation called $p_{gen} \in \{0, 1\}$. It was used as a switch to either sampling a word from the vocabulary distribution or copying the word from the attention distribution. They also solved the coverage problem by proposing a coverage loss, and intuitively this loss worked in a way that after the decoder was made aware of all the parts of the document it paid attention to earlier and this helped focus on other parts of the document, this also helped in reducing the repeating words problem. The architecture worked very well since its performance was much better than that of [65].

Around the time of the pointer generator, a parallel work solving abstractive summarization was published by [72]. Looking at the generation problem from a perspective of exposure bias, they proposed using reinforcement learning to train their model. Exposure bias stems from the issue of the availability of ground truth to the decoder when generating outputs during training time but the absence of these during test time because models are trained with MLE (Maximum likelihood estimation). Also, the space of valid summaries is wide and hard to accurately decide on. The CNN/Dailymail dataset only provides 1 reference summary for each document. Therefore they use Self-Critical Sequence training (SCST) [83] to train the model. This allows the model to be trained on the metric that it is evaluated on during test time. They find that just using MLE or SCST results in issues around making the model converge so they use a mixed objective which cold starts the network by first training

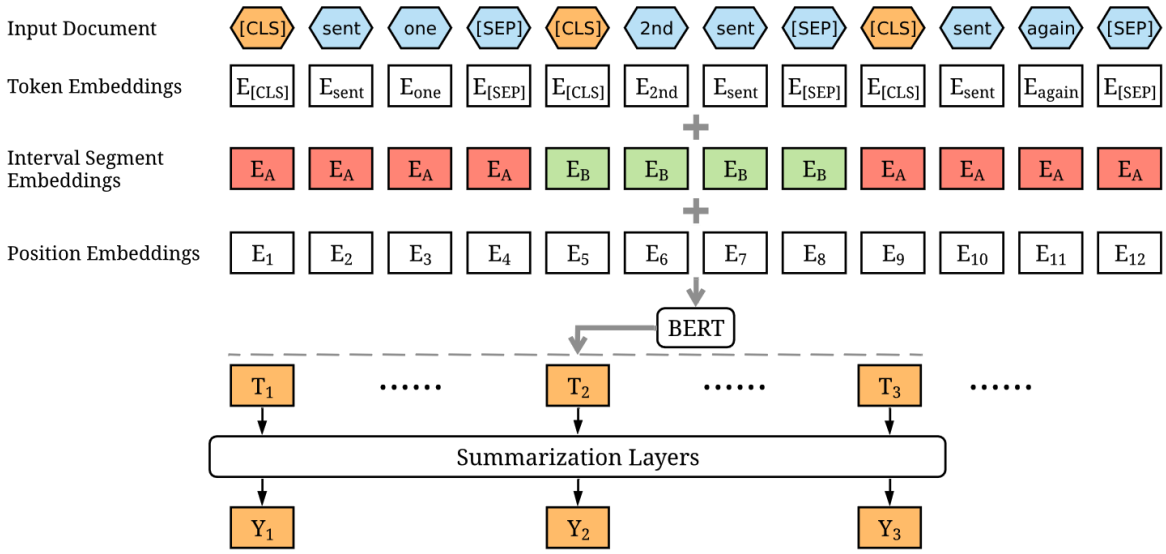

Figure 3.2 BERTSUM Architecture

it in MLE and later slowly weighing the SCST reward more. A clever trick that the paper applies to reduce repetition is using *trigram blocking*; if the generated sentence has a trigram that is already present in the summary generated before it, the sentence is skipped and not included in the summary.

3.1.2 BERT Based architecture for Summarization

The BERT model [22] trained on a large body of text was one of the first pre-trained models to show marked improvement on a bunch of tasks. The paper itself showed performance on Natural Language Understanding benchmarks like GLUE [102], showcasing that it had rich representations for language. It was also applied to SQUAD [81]a question-answering task but had yet to be adopted a wider range of tasks since the paper showed that the finetuning step could be applied and model performance could thus be further improved depending on the choice of task.

BERTSUM [59], was the first work in summarization to apply the BERT model to summarization. They applied the BERT model to both extractive and abstractive summarization. While a variety of architectures were proposed before using communicating agents [13], copying mechanisms like the pointer generator architecture and reinforcement learning techniques like [72], this model just used BERT in addition to a summarization layer for generating or selecting sentences resulting in a relatively simple model.

The encoder used for summarization in the BERTSUM model is the BERT model but since the model was trained with sentence segmentation as a part of its next sentence prediction task, directly feeding the document to the model would not have worked. The authors thus add [SEP], [CLS] (as shown in Figure 3.2) tokens between the sentences to demarcate sentence boundaries and allow the BERT model to process the document and produce rich paragraph and document level embeddings. These embeddings were then fed into the summarization layers in the case of extractive summarization and a decoder in the case of abstractive summarization. The extractive summarization layers were a 2-layer transformer with a sigmoid layer at the end and this model was called BERTSUMEXT. While the abstractive summarization decoder was a 6-layer transformer decoder to generate words and this model was called BERTSUMEXTABS. These models achieved state-of-the-art scores on the datasets that they were tested on.

3.2 The effect of pretraining on Scientific Summarization

3.2.1 The problem

While BERTSUM [59] showed better performance on a bunch of datasets including CNN/Dailymail, the model wasn't thoroughly tested on out-of-domain tasks. In our work, We focus on the task of scientific summarization thus studying domain adaptation of BERT Based models in a task and also studying the effect of intermediate finetuning and if this helps improve task performance.

3.2.2 The approach

Even though more powerful models like T5 [80] and BART [51] were introduced for better abstractive summarization, an issue found in abstractive summarization models was that they were hallucinating content that didn't exist and weren't faithful to the source document. In [61] and [45], they report that hallucinations are present in more than 70% of single sentence summaries and most hallucinations are neither factual nor faithful while mentioning that pre-trained models might have lower issues they weren't non-existent. In [40] they find that extractive models are better than abstractive models because of factual consistency. Extractive models do suffer from the issue of unnecessary text in the output (because sentences are directly selected), but abstractive summarization models suffer from omission (missing important details) and hallucination. They find that pretraining models work best and even improve the content selection capabilities of summarization models. In [25], the authors establish a tradeoff between faithfulness and the abstractive nature of the output.

For scientific summarization, We hence build on work from [59] and choose an extractive summarization model to solve the task since it is very important that a summary generated for a scientific document is both faithful (only generates or selects things in source document) and accurate when stating facts (does not add unnecessary things which might not be present).

While BERT-based models were better than the models before them on out-of-domain tasks, there was a lot of room left to be desired for further improvement. In [34], the authors show that adapting pre-trained language models to more domains with task and domain adaptive pretraining is helpful. In [100], intermediate pretraining steps are helpful in task transfer and especially help in the case of low-resource tasks. We find that this has not really been studied in the context of summarization. Since we are solving a scientific summarization task and BERT is a general-purpose model, we try task adaptation strategies before evaluating the model.

3.2.3 The dataset

We study the effects of intermediate pretraining with the help of two scientific summarization datasets, SciTldr and PubMed.

SciTldr introduced in [12] is an extreme summarization scientific summarization task. Collected from Openreview with the Openreview API[1] from authors who write short summaries/abstracts for the papers they submit. The inputs can be of two subsets:

- SciTldr-A - Abstract only, where the summary is generated just from the abstract.
- SciTldr-AIC - Abstract + Introduction + Conclusion, where the summary has to be generated with the help of all three.

We use the splits mentioned in the original paper and report numbers on both subsets.

PubMed originally introduced in [19] was introduced to test abstractive summarization on the longer forms of documents. It's collected from PubMed.com, where the abstract of the scientific document is considered the summary. Due to the long nature of the document and the limitation of the number of tokens that can be processed by BERT-like models, only the introduction is used to generate the summary. We use the splits in [114].

3.2.4 Experimental Setup

3.2.4.1 Model choice and Implementation details

We use two pre-trained language models as our base embedding models namely BERT [22] and SciBERT [7]. SciBERT was a model similar to BERT trained on scientific papers and

[1] https://github.com/openreview/openreview-py

Dataset Name	# documents				# tokens	
	Total	Train	Valid	Test	Doc	Sum
SciTldr	3,229	1,922	619	618	5,000	21
PubMed	93,204	83,233	4,946	5,025	444	209.5

Table 3.3 SciTldr and Pubmed Statistics

data in the domains of biomedicine and computer science. They showed a bunch of gains across a variety of tasks when it came to the scientific domain with this model.

Our summarization model is based on [59]. We first both the models without training on the SciTldr dataset to establish a baseline. Our intermediate pretraining experiments for SciTldr are as follows:

1. Train the model on training split of PubMed.
2. Train the model on training split of CNN/Dailymail but only use as many documents as PubMed, i.e., 83k documents, for fair comparison of the domain.
3. Train the model on a mixed dataset of PubMed and CNN/Dailymail of about 83k documents
4. Train the model on the full train split of CNN/Dailymail

For intermediate pretraining experiments on PubMed, we just use the full train split of CNN/Dailymail.

All training steps have a dropout rate of 0.1 and a learning rate of 2e-3 similar to [59]. We use a batch size of 3000 for the experiments. The selection of the best model is done by validating the model. The validation is achieved by validating rouge scores for one-line summaries. This is to factor in the "extreme" summarization nature of SciTldr. Ultimately to generate summaries, we choose 1 line for SciTldr and 6 lines for PubMed as done in [114].

3.2.4.2 Evaluation Criteria

All summarization models are evaluated with ROUGE [54]. ROUGE stands for Recall-oriented Understudy for Gisting Evaluation. ROUGE-1 compares unigrams of the generated summary and the reference summary to give a score, ROUGE-2 uses bigrams, and ROUGE-L uses the longest common subsequence. ROUGE is an automatic metric. When comparing extractive summarization systems, the best possible score is when the system is an Oracle.

Oracle scores are calculated by selecting the best possible subset of sentences, which maximizes the rouge score against the gold truth. These often act as "the best possible" extractive summarization systems for comparison.

3.2.4.3 Model Comparison

We compare our results to a model called MatchSum introduced in [114]. The model solves the extractive summarization task as a text-matching problem and shows state-of-the-art scores in a variety of different datasets.

3.2.5 Results

In table 3.4, We note the results of intermediate pretraining experiments. We find that while PubMed helps as a task, CNN/Dailymail helps more. The mixed dataset shows the maximum gains, and we conclude that selectively choosing articles from a mix of high-quality task data (in this case CNN/Dailymail) and target domain task data (in this case PubMed) is useful and beneficial for domain task adaptation. Using SciBERT, i.e., a pre-trained language model in the target domain does not seem to help with performance suggesting that it's better to start with a general pre-trained model like BERT and then adapt it to the target domain rather than using domain-specific pre-trained models.

In table 3.5, we report the results for PubMed. We see that intermediate pretraining is of little use for this dataset and conclude that intermediate pretraining might not be as useful in the case of target datasets which are very large (in the case of PubMed about 83k documents).

We do two ablation studies whose results are shown in table 3.6 and table 3.7. For the first ablation study, the numbers which are presented in table 3.6, we do the influence of intermediate pretraining dataset size on the final performance of the model on the target dataset. We find that although there are some gains when the model goes from 83k documents to 176k documents, the model quality does not improve with more documents. This suggests that there is some optimal size of dataset useful for low resource task adaptation. For the second ablation study, the scores for which are presented in table 3.7, we change the input length of the document and see if there is any change in the performance, while there is no improvement when going from 512 tokens to 1024 tokens, there is some improvement in the case of 1500 tokens. The gains achieved with greater input length seem to be substantial in nature.

We also note that in the case of SciTldr, our approach leads to higher scores when compared to MatchSum and comparable performance when it comes to PubMed. The Matchsum model also starts with a pre-trained language model as its base, but since its a Siamese network, it takes about 8 Tesla-V100-16 GPUs to train, while our intermediate pretraining step

	SciTldr-A			SciTldr-AIC		
	R1	R2	RL	R1	R2	RL
Oracle	49.2	26.0	39.9	53.7	29.9	43.9
MatchSum† (BERT-base)	42.7	20.0	34.0	38.6	16.4	30.1
Our Models						
Pretraining Datasets	**Using BERT**					
-	39.71	18.91	32.63	36.99	16.14	29.64
Pubmed (83K)	41.49	19.57	33.40	40.82	18.98	32.84
CNN/DM (83K)	41.69	19.55	33.44	41.93	20.10	33.95
Mixed (83K)	**42.32**	**20.50**	**34.30**	**42.78**	**21.06**	**34.83**
CNN/DM (Full)	42.26	20.32	34.09	42.21	20.24	34.19
	Using SciBERT					
-	39.93	18.50	32.32	37.16	15.94	29.65
CNN/DM (83K)	40.60	19.04	32.93	40.74	19.09	32.95
Pubmed (83K)	41.10	19.33	32.87	40.61	18.69	32.68
CNN/DM (Full)	40.66	19.08	32.59	41.25	19.40	33.37

Table 3.4 SciTldr ROUGE scores. † Results from [12]

can be achieved on a single Nvidia 2080, thus also offering a less computationally intensive alternative to domain adaptive summarization.

3.3 Conclusion

In this chapter, we study pre-trained models and their influence on the summarization task. We specifically learn the BERTSUM [59] architecture and also extend its capabilities. With the help of experiments, we understand that it is possible to improve summarization models that use BERT-like models as their input embedding models. Domain adaptation of these models is possible, and this can also be achieved by training these models on the task of choice first before applying them to the target task. This especially helps in the case of low-resource task data, which is often seen in specialized domains. This also helps us understand what we mean by domain adaptability and generalization and what we seek from pre-trained transformer language models.

This work saw the application of BERT-based models to summarization, specifically long-form summarization in the scientific domain, and we discovered that BERT models could be

	Pubmed		
	R1	R2	RL
Oracle	45.12	20.33	40.19
MatchSum† (BERT-base)	41.21	14.91	36.75
	Our Models		
Pretraining Datasets	**Using BERT**		
-	40.65	14.85	36.18
CNN/DM (Full)	40.77	14.92	36.29
	Using SciBERT		
-	**41.08**	**15.16**	**36.59**
CNN/DM (Full)	40.59	14.76	36.12

Table 3.5 PubMed Rouge scores.† Results from [114]

Dataset Size	R1	R2	RL
83K articles	41.93	20.10	33.95
176K articles	**42.27**	**20.37**	**34.32**
286K articles	42.21	20.24	34.19

Table 3.6 CNN/DailyMail intermediate pretraining size variation while finetuning on SciTldr-AIC.

further trained to improve their performance on target tasks. It was apparent that further leaps of performance in these tasks could not be gained without better language models, which would thus lead to better language understanding and better evaluation of these models, which help us gauge the ability of these models to evaluate models. There was also a need to move away from the single-dimensional ROUGE evaluation and metrics that add nuance to the models' abilities. This led to the subsequent work on designing a general-purpose natural language generation benchmark to measure the abilities of language models holistically. This is the focus of the following two chapters in this thesis. The next chapter looks at evaluating models better, while the latter chapter focuses on developing a large multilingual language model and furthering its language understanding abilities and studying those.

In Chapter 4, we learn about the rise of benchmarks to evaluate Natural language processing models and their motivations. We also understand the design with the help of a generation benchmark called GEM.

	SciTldr-AIC			Pubmed		
Input Length	R1	R2	RL	R1	R2	RL
512	42.21	20.24	34.19	40.65	14.85	36.18
1024	42.21	20.34	34.35	42.44	16.39	37.86
1500	**42.23**	**20.65**	**34.41**	**42.65**	**16.59**	**38.03**

Table 3.7 Input length variation. The pretraining dataset is CNN/DailyMail for SciTldr-AIC and none for Pubmed.

Chapter 4

Evaluation in the LM era & The GEM Benchmark

> The real voyage of discovery consists not in seeking new landscapes, but in having new eyes.
>
> Marcel Proust

As we saw earlier in Chapter 2, Language models are evaluated using extrinsic and intrinsic evaluation. The intrinsic evaluation involved testing models on linguistic capabilities while extrinsic evaluation was about testing models on end tasks in the form of Question answering, Machine Translation, Summarization, etc.

Most models developed in the statistical NLP era and early deep learning era were about developing task-specific architectures. These models solved the target task of choice and were designed with inductive biases tailored to the ask which helped them better. For question answering, [91] developed the BIDAF architectures, which stands for Bidirectional attention flow, for machine comprehension. This was a multistage architecture that developed representations at the character, word, and paragraph levels in the hope of solving the task better. For summarization, we saw the pointer generator architecture by [89] in Chapter 3 which had a copy mechanism for factual details, and coverage loss for paying attention to the entire article. Both architectures helped improve performance by significant margins and spurred the research of more task-specific architectures.

Task-specific architectures were not very good at Natural language understanding and seemed unable to have a general understanding of language. These models were designed only to solve a specific task after all and only for a specific dataset with a specific domain, thus struggling to generalize to other tasks and have linguistic capabilities. There was a need to develop more general-purpose models.

Name	Task	Train	Test	Metric
		Single sentence tasks		
CoLA	Acceptability	8.5k	1k	Matthews Corr.
SST-2	Sentiment Analysis	67k	1.8k	accuracy
		Similarity and paraphrastic tasks		
MRPC	Paraphrasing	3.7k	1.7k	accuracy/F1
STS-B	Sentence Similarity	7k	1.4k	Pearson/Spearman Corr.
QQP	Paraphrasing	364k	391k	accuracy/F1
		Inference tasks		
MNLI	NLI	393k	20k	matched acc./mismatched acc.
QNLI	QA/NLI	105k	5.4k	accuracy
RTE	NLI	2.5k	3k	accuracy
WNLI	NLI	634	146	accuracy

Table 4.1 GLUE Tasks

ELMO [75] came out with a deep contextual representation model using a bidirectional LSTM. They showed that the model was not only good at linguistics (syntax and semantics) but at extrinsic tasks as well like natural language inference, question answering, and sentiment analysis.

How do you measure the capabilities of more general models? What aspects do you judge? GLUE [102] came out in 2018 as an answer to these questions.

4.1 GLUE: General Language Understanding Evaluation

GLUE benchmark is a collection of natural language understanding tasks. These include tasks like QA (question answering), sentiment analysis, and textual entailment. The idea was not to place limits on what kind of model was developed or trained but rather to understand model capabilities. The only assumption is that the model should be able to process sentences and sentence pair inputs. The benchmark was created by curation of existing datasets rather than creating new data. The benchmark ultimately tried to test linguistic knowledge and the ability of this knowledge effectively transfer to other tasks.

They also announced a leaderboard still present at a website[1] which also served as an online evaluation platform.

[1] https://gluebenchmark.com

4.1.1 Tasks in GLUE

There are 9 tasks in GLUE, and all of them are centered around the English language. These are mostly sentence understanding tasks and the primary motivation again was to spur general language NLP systems.

- CoLA from [105] called the Corpus of linguistic acceptability is a dataset of English acceptability judgments drawn from documents on linguistic theory. The task is to judge if a given sentence is grammatical or not. The evaluation metric for this is Matthews correlation.
- SST-2 from [97] has movie reviews in the form of sentences and sentiments. The task is to predict the sentiment of the review. The evaluation metric is accuracy on the labels.
- MRPC from [23] are pairs of sentences extracted from news websites. The task is to judge whether the sentences are semantically the same, i.e., they mean the same thing. The evaluation metric is accuracy on the labels, and also the F1 score since the classes in the dataset is imbalanced.
- QQP from [41] was released by Quora and is a collection of questions. The task is to judge whether two questions are semantically the same, i.e., they mean the same thing. Similar to MRPC, both accuracy, and F1 are calculated.
- STS-B from [14] is a collection of sentences from a variety of sources. The task is to predict similarity scores between pairs of sentences. The evaluation criteria are Pearson and Spearman correlation.
- MNLI is a natural language inference dataset from [108]. Given a premise sentence and a hypothesis sentence, the model has to output if the hypothesis follows the premise (entailment), contradicts it (contradiction), or neither (neutral). Both matched (in the domain), and mismatched (cross-domain) accuracy scores are reported.
- QNLI is a question-answering dataset by [81], the original dataset is recast to the task of determining if a context sentence has an answer to a question. This recast dataset is called QNLI. The metric used is accuracy.
- RTE Recognizing textual entailment is a dataset formed by combining RTE-1 [20], RTE-2 [5], RTE-3 [32], RTE-5 [9]. The task is to predict entailment or not entailment. The metric used is accuracy.
- WNLI is the Winograd schema challenge [50] originally a correference resolution task. The task is recast into a sentence pair classification task by changing the uncertain pronoun with possible references. The task is to predict if the sentence with the pronoun follows the original sentence. The converted dataset is called WNLI (Winograd NLI).

Name	Task	Train	Dev	Test	Metric
BoolQ	QA	9427	3270	3245	acc.
CB	NLI	250	57	250	acc./F1
COPA	QA	400	100	500	acc.
Multi-RC	QA	5100	953	1800	$F1_\alpha$/EM
ReCoRD	QA	101k	10k	10k	F1/EM
RTE	NLI	2500	278	300	acc.
WiC	WSD	6000	638	1400	acc.
WSC	Coref	554	104	146	acc.

Table 4.2 SUPERGLUE Tasks

While GLUE saw quick popularity in the NLP community and became a successful evaluation framework. The introduction of GPT [78] and BERT [22] saw improvements on the task that were not expected while creating the benchmark. The progress had made it impossible to continue using GLUE Benchmark since models that surpassed human performance on GLUE had already been introduced. To address this, the authors released a tougher and more rigorous benchmark called SuperGLUE.

4.2 SUPERGLUE

SuperGLUE [101] followed the same evaluation benchmark design as GLUE. It had more challenging tasks and task formats were increased to now include question answering and coreference resolution.

4.2.1 Tasks in SUPERGLUE

There are 8 tasks in SuperGLUE, and all of them are centered around the English language and are solvable by most college-educated English speakers. These are question answering, natural language inference, and coreference resolution tasks to judge a broader surface area of tasks solvable by models and evaluate it more thoroughly.

- BoolQ Boolean Questions from [17] is a question-answering task containing a short passage and a yes/no question about the passage. The metric of the evaluation was accuracy.

- CB The Commitment Bank from [21] is a collection of short texts in which sentences contain embedded clauses. The task is a natural language inference task. Because of class imbalance, F1 is also reported along with accuracy.

- COPA Choice of Plausible alternative from [85] is a causal resoning task. The task is to determine the effect or cause given a premise sentence. Evaluated using accuracy.

- MultiRC Multi Sentence Reading comprehension from [44] is a question-answering task consisting of a short text, a question about it, and some possible answers. The model must determine which of the answers are true and which of them are false. This is a multi-domain dataset. The evaluation criteria are F1 overall answer options ($F1_\alpha$) and exact match (EM).

- ReCoRD Reading comprehension with commensense reasoning dataset from [111] is a multiple choice questioning answering task. It comprises news articles and questions about the article with a word/entity masked out (these types of questions are called cloze-type questions). The model has to predict which of the entities in the passage are the answer to the question. Evaluated with F1 and EM.

- RTE retained from GLUE because the only task in GLUE which was not beaten and had a lot of margin for growth.

- WiC Word in Context from [76] is a word sense disambiguation task, basically when you use the word to mean separate things in different contexts. For example, "This notebook is mine" and "I am going to work in the diamond mine," The word "mine" means two different meanings/senses in a different sentence. The task is given two texts and a polysemous word to determine if the word is used in the same sense or not. Accuracy is the evaluation criterion of this task.

- Wsc Winograd Schema Challenge from [50] is a coreference resolution task. Given a sentence that consists of a pronoun and a list of nouns from the sentence, the model must resolve which of the nouns the pronoun was referring to. The evaluation metric for the task is accuracy.

4.3 The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics

The popularity of GLUE [102] and superGLUE [101] led to the rise of leaderboard wars, where models focused on trying to gain some marginal points on performance [56]. Another

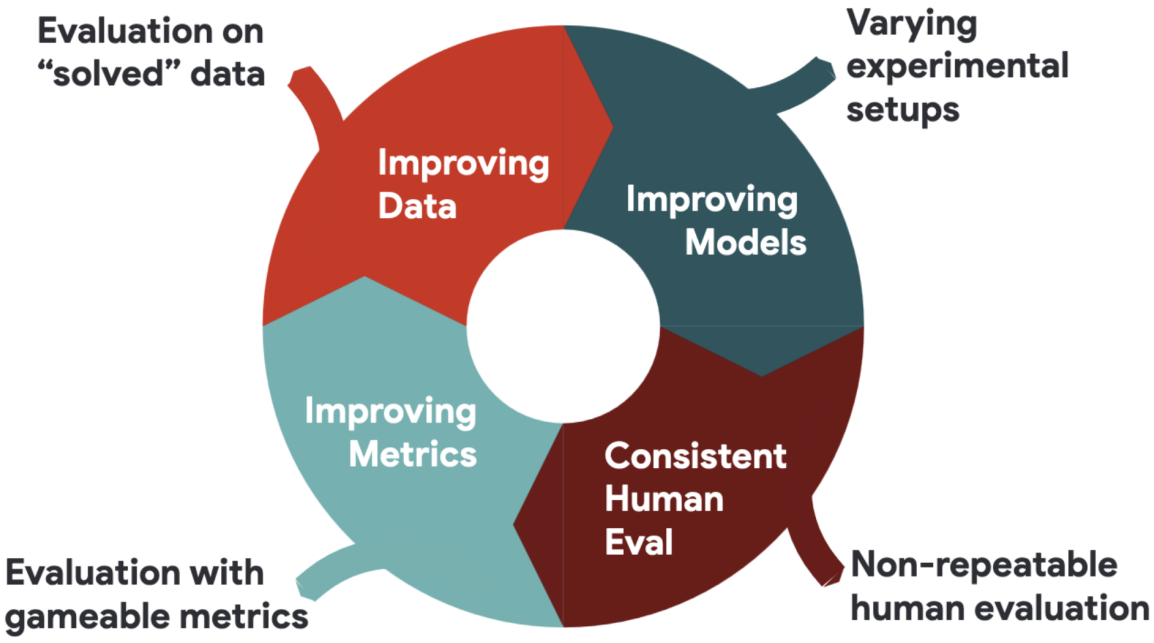

Figure 4.1 GEM Benchmark philosophy

trend observed in recent years with the advent of GPT-2 [79], GPT-3 [11], T5 [80] etc. accelerated the trend of Natural language generation systems which are much more capable than earlier BERT-based models, and GLUE and superGLUE majorly focused on NLU tasks which did not capture the evaluation of generation capabilities. Also with the rise of mBERT [22], mT5 [109], and other multilingual models there is a lack of benchmarks that capture and test capabilities not only in a single language i.e., English but also a wide set of other languages when models are trained with multiple languages new capabilities like cross-lingual transfer and other abilities emerge which have to be adequately tested by benchmarks. Although GLGE (General language Generation Evaluation) [57] the benchmark falls to the pitfall of basing the entire benchmark on a single metric like ROUGE which does not capture enough nuance about models and their capabilities. Thus, we introduce and design the GEM benchmark [31], a "living" benchmark for Natural language generation, its evaluation, and metrics.

The GEM benchmark ethos as shown in Fig 4.1 is the idea of continually evolving benchmarks. GLUE and superGLUE etc. show some gameable metrics, and as superGLUE did with GLUE there's a need to continually choose more challenging tasks in a rapidly evolving and improving NLG model landscape. There's a need for consistent human evaluation as well since automatic evaluation does not paint a nuanced picture. Overall, a living bench-

mark is one that is supposed to improve and evolve trying to meet the criteria and test the models that exist during the times and not get obsolete by ever-evolving models.

4.3.1 Tasks in GEM

The datasets chosen for GEM 1.0 are shown in table 4.3. They measure two aspects of generation one content selection and planning i.e., "What to say" given a body of text how do you select the best parts of the data and make sure they are a part of your generation and surface realization i.e., "how to say it" given a body of text how do you generate fluent and high-quality text about the data. Models need to be capable of simplifying sentences, paraphrasing them, generating concise text, not falling into the usual pitfalls of text generation, etc. While designing the benchmark questions around task focus vs task diversity, low resources vs high resource languages were asked.

GEM features 18 languages and the size of the datasets is from 5000 to 500,000. Some of the tasks do not have any English component and are just multilingual in nature. In accordance with guidelines in Data-to-text tasks from [74] and general evaluation of NLP models from [84], we add another dimension of testing by proposing challenge sets for 10 tasks. Last but not least to not fall into the pitfall of using a single evaluation criterion and paint a broader nuanced picture of model performance we select a variety of evaluation criteria both automatic and human.

In this work, I have contributed to the choice of datasets in the broader GEM benchmark and also those specifically pertaining to summarization researching the choices available and choosing ones that would help evaluate language models better both across domains and languages and also suggested some initial evaluation criteria for Summarization beyond ROUGE. I have also contributed to the general design of data cards which resulted into the work on Reproducible Data cards [62] and is used in huggingface datasets [52].

The 11 datasets are chosen from a wider range of 35 datasets. The datasets are as follows:

- CommonGEN from [53] is a dataset from common sense reasoning it requires compositional generalization and relational reasoning to solve. The task is to generate a sentence that is concise and fluent, given a set of common concepts. The language is English.

- Czech Restaurant from [27] is a dataset in Czech in the restaurant domain. The Czech language is morphologically rich and requires inflecting named entities and delexicalization to be good at the task. The task is to generate concise text for a meaning representation. The language is Czech.

- DART from [66] is a data-to-text generation dataset. The data is a table. The table is represented in the form of triples. The task is to generate coherent text for the given triples. The language is English.

- E2E clean from [68], [26] is a task to generate coherent text key-value attribute pairs from the restaurant domain similar to the Czech restaurant task. The language is English.
- MLsum from [88] is a multilingual summarization dataset that is scraped from online newspapers. There are 5 languages in the original dataset namely French, German, Spanish, Russian, and Turkish. We only choose German and Spanish for the purposes of our benchmark. The task is to generate a relevant fluent summary for the news article. The language is English.
- Schema guided Dialog from [82] is a dataset that is a response generation dataset. The task is to generate an output when a dialog act is presented to it. The domain is the hotelier industry majorly. The language is English.
- ToTTo from [71] is a task to generate text from a table. The table is given, and the highlighted cells in the table are also marked. The task is to generate a coherent text for these highlighted cells. The language is English.
- XSUM from [67] is a form of Extreme summarization task. The dataset is collected from an online news website. The task is to generate the summary of the dataset in one sentence. The language is English. The dataset is shown to have hallucination flaws in [61]. To combat this 500 document summary pairs, which were manually annotated are used to train a classifier and all document summary pairs for which $\mathtt{P(faithful)} > 0.8$ are removed.
- WebNLG from [30] is a collection of DBpedia triple sets. The task is to generate coherent texts for RDF triples. The language is English and Russian.
- WikiAuto + Turk [42] and ASSET [1] are datasets that cover the task of sentence simplification. Given a sentence from Wikipedia, the task is to generate a sentence in simplified layman's language. The language is English.
- Wikilingua from [46] is a large-scale multilingual dataset with article summary pairs across 18 languages. Whenever present an article has parallel data for the same in another language as well. This also allows the dataset to be used for cross-lingual abstractive summarization and we use it for the same idea. Although present for 18 languages we focus on a cross-lingual setup from English, Spanish, Russian, Turkish, and Vietnamese to English summaries.

4.3.2 Evaluation Criteria

To avoid succumbing to the pitfalls of single evaluation criteria benchmarks, we choose a wide array of evaluation criteria to allow for a more nuanced understanding of models.

The idea is to change "System X performs best" because we have a singular leaderboard to "System X performs better on these metrics, not so well on these and comparable in those," so we understand the model in a broader fashion.

The evaluation criteria for the benchmark is as follows:

- ROUGE [54] is chosen to test for lexical similarity and is the standard evaluation criterion for summarization tasks. Although it is shown in recent work that the metrics performance and correlation to human judgments the number of references, it is shown on a system level to still correlate well with human judgments. We use ROUGE 1/2/L which measures the unigram, bigram and longest subsequence overlap.

- BLEU from [70] is used to measure lexical overlap as well; it is mostly used in Machine translation.

- Meteor from [4] is used to measure lexical overlap as well. This differs from ROUGE and BLEU in the sense that it tries to evaluate with the generalized concept of the world instead of doing word-to-word lexical overlap like the other metrics. This essentially helps in accepting and scoring a wider range of valid summaries.

- BERTscore from [113] is used to measure semantic equivalence. In this contextual embeddings are used to compute token similarities.

- BLEUrt from [90] is used to measure semantic equivalence. It is a learned metric that models human judgment, and given a reference summary, the generated summary and the source document it returns a score.

In [35], the authors talk about unifying statistical evaluation in NLP and how they cover a variety of measures. They also argue that generation models trade off diversity for quality. To measure diversity metrics, we measure the Shannon entropy [92] over unigrams and bigrams H_1, H_2. We also measure the ratio of distinct n-grams over the total number of n-grams and count the total number of unique n-grams. These measures capture how diverse the output of the NLG model is.

4.3.3 Model Baseline and other details

To establish some model baselines, we run the benchmark over some standard multilingual generation models found in the wild. These models include BART [51], T5 [80], mT5 [109], mbart [58], PEGASUS [110]. We only show the best models for each respectively in table 4.4.

As mentioned earlier challenge sets are proposed for about 10 tasks out of the 11 in the benchmark. These test sets are created by either manipulating the existing test set by changing a small attribute e.g. Numerical variation where the cardinal value in the documents

is changed with random values. Also done by breaking down test sets into different complexities For e.g. taking randomized samples of validation and training data. Also done by compilation of new challenge sets. For e.g., we collect covid19-related keyword articles for XSUM and MLSUM and release them as challenge sets.

4.4 Future of GEM and conclusion

In this chapter, We learned about the nature of benchmarks, a brief insight into why they were created, and the growth they spurred in the field of NLP. We also saw the development of the GEM benchmark. This multilingual natural language generation benchmark addressed issues for previous benchmarks while ensuring it did not become obsolete by becoming a "living" benchmark. The GEM benchmark also tries to offer more nuance into model abilities by having a wide net of evaluation criteria.

Next, we learn about the development of the largest open-source, multilingual language model called BLOOM.

Dataset	Task Description	Language(s)	Size	Input Type
CommonGEN [53]	Generate text for all concepts	en	67k	Concept Set
Czech Restaurant [27]	Generate text for given intent and preferred attributes	cs	5k	Meaning Representation
DART [66]	Describe table given in triples	en	82k	Triple Set
E2E Clean [68], [26]	Describe a restaurant given preferred attributes	cs	42k	Meaning Representation
MLSUM [88]	Summarize News	de/es	520k	Article
Schema-Guided Dialog [82]	Provide the surface realization for virtual assistant	en	165k	Dialogue Act
ToTTo [71]	Generate text for highlighted cells	en	136k	Highlighted table
XSUM [67]	One line summary of news	en	25k	Article
WebNLG [30]	Generate natural text for Triple	en/ru	50k	RDF Triple
WikiAuto+Turk /ASSET [42] [1]	Text Simplification	en	594k	Sentence
Wikilingua [46]	High quality summary of article	*ar/cs/de/en es/fr/hi/id/it ja/ko/nl/pt/ru th/tr/vi/zh	550k	Article

Table 4.3 Description of datasets in the GEM Benchmark

Dataset	Model	Metrics (Lexical Similarity and Semantic Equivalence)						
		METEOR	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	BLEURT
CommonGen	BART	0.301	63.5	32.5	55.1	27.5	0.943	-0.400
Czech Restaurant	mT5-XL	0.229	52.1	31.3	47.3	17.0	0.905	–
DART	T5	0.115	8.4	0.0	8.4	0.02	0.901	-0.091
E2E clean	BART	0.373	73.6	48.5	57.8	43.5	0.948	0.190
MLSum (de)	mT5-XL	0.102	12.6	3.7	10.5	5.3	0.832	–
MLSum (es)	mT5-XL	0.247	33.1	15.0	27.2	11.9	0.849	–
Schema-Guided	T5	0.331	58.2	36.8	52.6	33.4	0.874	0.009
ToTTo	T5	0.363	70.1	48.3	60.1	42.2	0.914	0.179
XSum	PEGASUS	0.216	46.5	23.2	38.1	17.0	0.918	-0.186
WebNLG (en)	mBART	0.462	83.4	63.1	70.3	66.1	0.967	0.458
WebNLG (ru)	mBART	0.613	34.8	13.4	33.0	47.0	0.888	–
Turk	BART	0.556	90.3	86.1	89.9	88.3	0.967	0.358
ASSET	BART	0.560	90.1	82.3	89.6	92.4	0.982	0.407
WikiLingua (es→en)	mBART+	0.196	40.7	16.9	34.1	14.3	0.858	-0.248
WikiLingua (ru→en)	mBART+	0.174	37.3	14.9	31.9	12.0	0.851	-0.303
WikiLingua (tr→en)	mBART+	0.204	43.7	20.8	37.9	17.5	0.866	-0.252
WikiLingua (vi→en)	mBART+	0.183	38.1	15.4	32.5	13.3	0.853	-0.284

Table 4.4 Best models respectively amongst the ones tested for each task in GEM.

Chapter 5

Bloom: A large multilingual language model

> Any sufficiently advanced technology is indistinguishable from magic.
>
> Arthur C. Clarke

Large language models have become a cornerstone of Natural language processing in the last three years. The idea was slowly pioneered over the years. It started with the idea of contextual embeddings in ELMO [75], and having a language modeling step before task training [39]. Then the first relatively bigger language model was trained by OpenAI, GPT [78], BERT, a pre-trained language model [22] furthered the cause, both the models showing that models gain language abilities when trained over large data and then fine-tuning them on target tasks helped improve their performance further. GPT-2[79] was the first model to be trained in the billion parameter scale with the 1.5B parameter model, GPT-3 [11] was subsequently trained with a 175B parameter model, both the papers showed that when trained on large bodies of data, these large language models gained the ability to tasks directly without any additional training or fine-tuning.

Neural Scaling laws were first established and proven in [43]. The paper proved that increasing compute (measured in PF-days i.e. Petaflop days), increasing dataset size, and increasing parameters improved the model's performance and suggested that the maximum amount of performance is gained when all the parameters are increased in tandem. Chinchilla [38] investigated the optimal model size and amount of tokens under a given "compute" budget, they find that scaling the model size should be done equally, and for better performance, every doubling of parameters should be coupled with doubling of tokens that the model is trained on. They cement this result by training a model called Chinchilla with 70B parameters but similar performance to the GPT3-175B and other large language models. In [107], the authors study the emergent abilities of language models. They define emergent

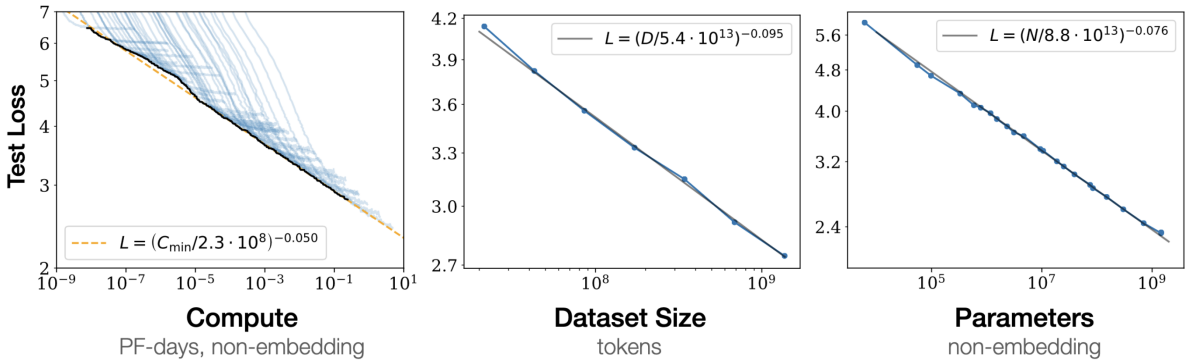

Figure 5.1 Scaling Laws shown in [43]

ability as "An ability is emergent if it is not present in smaller models but is present in larger models." and find several such tasks which aren't solved by models with a smaller number of parameters but are solvable by larger models beginning especially in the 70B parameters and above range.

Most of these large language models are neither public nor multilingual (although training with hundreds of billions of tokens does result in some multilingual performance). To address these issues, we present the "BigScience Large Open-science Open-access Multilingual Language Model" (BLOOM, BigScience Workshop, 2022[87]). The model is a 176-billion-parameter large language model trained on 59 languages (46 languages, 13 programming) and was developed as a part of the BigScience workshop. The compute for training the model was provided by a French public grant and involved giving access to the IDRIS Jean Zay supercomputer.

The scale of this work was global and collaborative, with 900 individuals contributing and 350 authors to the paper. I was a part of the 35 core contributors working on the evaluation of the trained models. I have contributed to the experimental design of evaluating the BLOOM models, the choice of datasets, the nature of the evaluation (in-context learning), and the choice of evaluation metrics. Also contributing to the selection of models to compare the BLOOM model to and running evaluations on almost all models except the BLOOM 175B. I also contributed to writing prompts for the datasets and integrating the lm-evaluation-harness repository, a repository made for easy few-shot evaluation of language models, and the prompt source repository, a repository made for the development of prompts for various tasks.

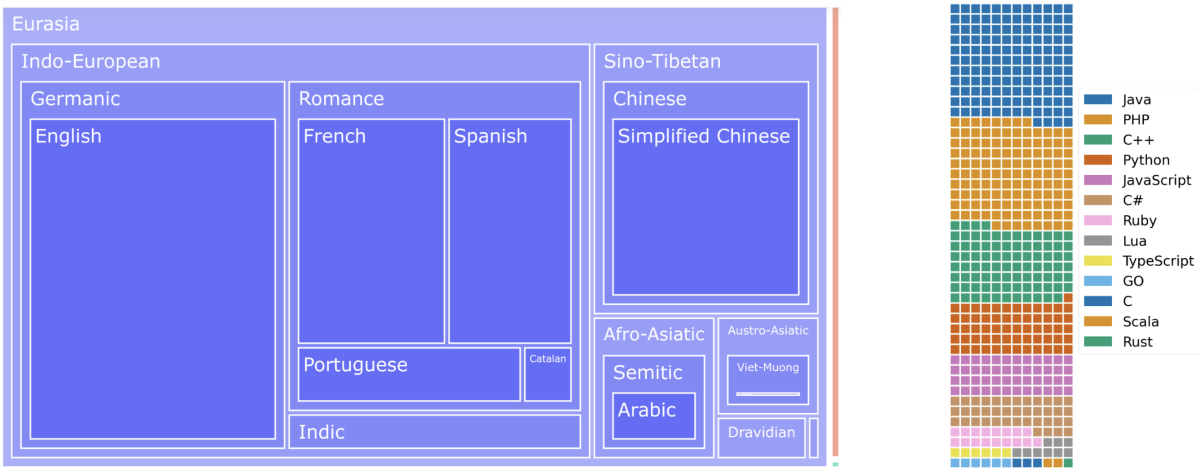

Figure 5.2 ROOTS dataset

5.1 Training dataset

BLOOM was trained on the ROOTS corpus [48], a collection of HuggingFace datasets and, as mentioned earlier, comprises 59 languages, which is made up of 46 natural languages and 13 programming languages. A broad view can be seen in the treemap in fig: 5.2. The code dataset was collected through BigQuery [1].

Deduplication is done by removing datasets with a limited amount of natural language and small datasets for high-resource languages. Documents with high repetition of words and characters, highly filled with special characters, are removed. Also, documents with flagged words are removed. Personally identifiable information is removed, and details like phone numbers, credit card numbers, email addresses, social media handles, and IP addresses are replaced with rule-based approaches with regular expressions.

To study the 46 languages inside the ROOTS corpus, we need to understand that they come from 3 macro areas and nine families of languages. These families include Dravidian, Austronesian, Basque, Indo-European, Mande, Austro-Asiatic, Niger-Congo, Sino-Tibetan, and Afro-Asiatic. English is the dataset's most prominent language, after which comes Chinese, French, Spanish, and Portuguese. The size of the corpus, when compared to other corpora, can be seen in Fig 5.3. C++, Java, and PHP account for more than half of all coding-related data in the corpus.

While we will look at BLOOMZ in detail later in this chapter, to prepare a dataset to train the BLOOMZ model, a large-scale hackathon was done by contributors to write prompts for

[1] https://cloud.google.com/blog/topics/public-datasets/github-on-bigquery-analyze-all-the-open-source-code

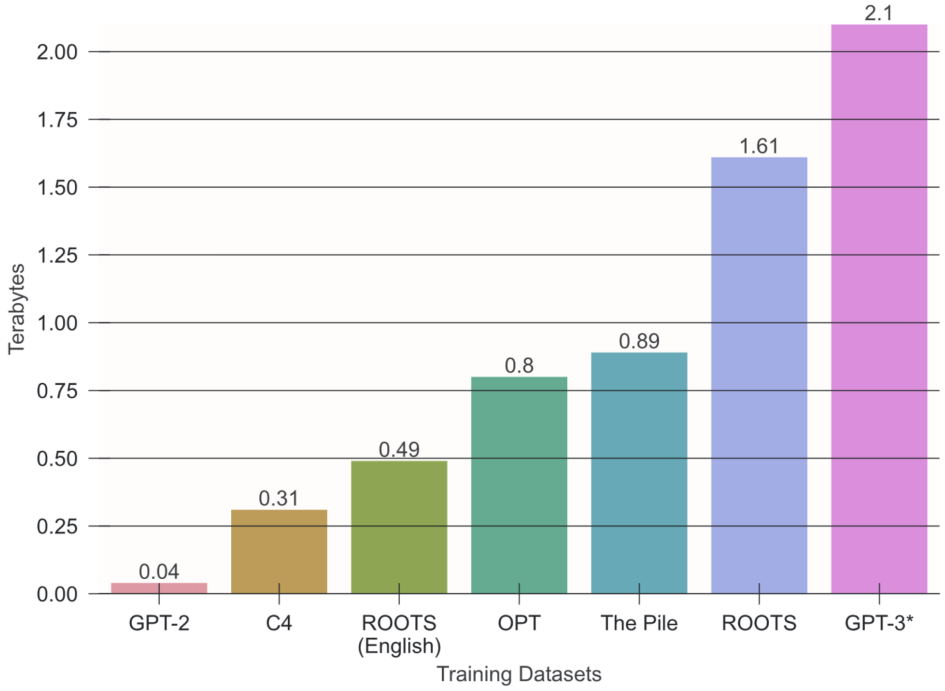

Figure 5.3 ROOTS dataset size when compared to other recent large datasets

about 180 datasets. This public pool of prompts was used to train the BLOOM model with multilingual zero-shot possibilities, and we call this bigger model BLOOMZ.

5.2 Experimental Setup

5.2.1 Model Architecture

In the work of [104], they explore which language model architecture and pretraining objectives work best for Zero-shot generalization. They do a variety of experiments, as seen in fig 5.4. The parameters that are ablated are pretraining objective, adaptation, i.e., how well a model on one architecture can be retrained for new ones, fine-tuning impact, i.e., how well a model adapt to newer tasks, domains, or multilingual capabilities, and finally, evaluating with some end tasks to validate these ideas. The experiment methodology is shown in Fig 5.4. The findings from their paper are summarized as follows:

1. Multitask fine-tuning helps zero-shot generalization a lot.

2. Decoder-only models, when trained with full language modeling objectives, are better than others after self-supervised pretraining.

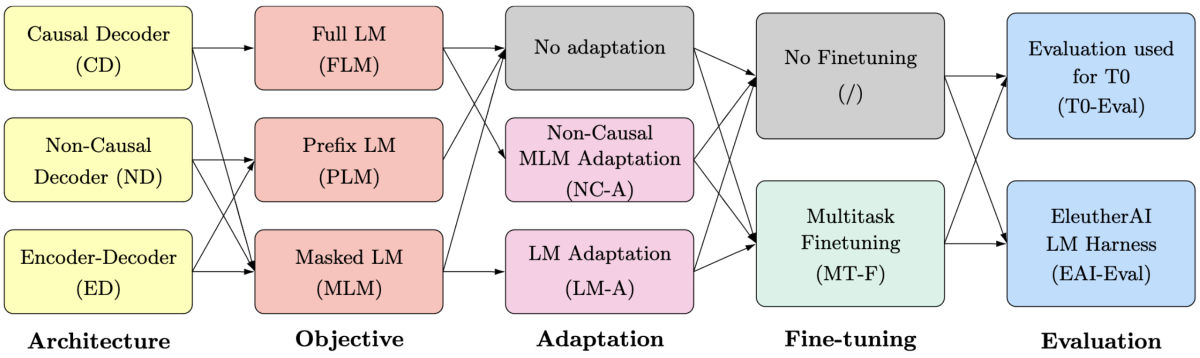

Figure 5.4 Ablation experiments for determining architecture in [104]

3. Encoder-decoder models are better after multitasking and fine-tuning.

As we saw earlier in the case of the Chinchilla model [38], The authors mention that increasing compute, amount of tokens, and parameters of the model can improve the performance of a model. One of the questions ultimately that is natural to ask when you have a lot of compute is, "What kind of language model can I train in this much compute?". In [49], they explore precisely this question. The findings from their paper are summarized as follows:

1. High-quality sources improve zero-shot generalization. This fact also suggests that curating your pretraining data to include high-quality data is a good idea. It should be as diverse and cover as many domains as possible.

2. ALiBi [77] PEs (Positional embeddings) perform better than others.

3. Layer normalization degrades zero-shot generalization.

The results from [104] and [49] ultimately influence the BLOOM architecture. Most of these experiments were done on the same eval set as To[86] and Eleuther LM-Eval Harness[29]. A mixture of experts [93] are not considered because their effectiveness at scale was not proven.

The original transformer model is an encoder-decoder model. Since then, various transformer architectures have been tried. BERT [22] is an encoder-only model. T5 [80] is an encoder-decoder model trained to generate sequences. GPT-2 [79], GPT-3 [11] and the original GPT[78] are all decoder only models and have shown at the 100B+ scale that they outperform everything. Although T5 showed that the encoder-decoder worked better than most, results have yet to be established at the 100B+ scale. Coupled with results from [104], which showed that causal decoder models perform well soon after pretraining, we chose a decoder model as our architecture.

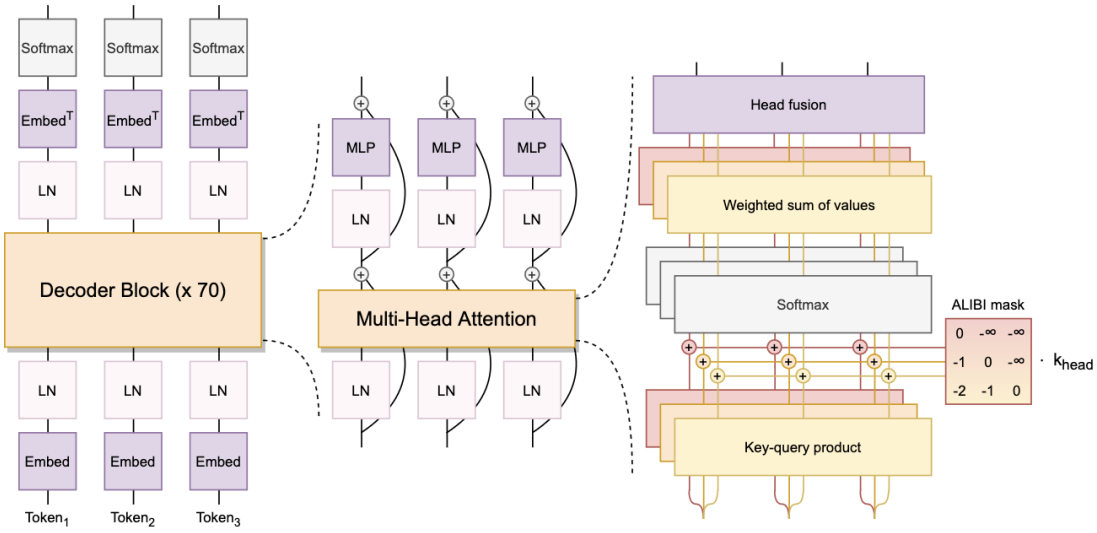

Figure 5.5 The BLOOM architecture

For positional embeddings, we choose ALiBi [77] positional embeddings, which have shown that not only do they extrapolate to longer sequence lengths they also perform pretty well at the original sequence length when compared to the original positional embedding in [99] and a newer type of work in embeddings called the rotary embeddings [98].

We added an additional layer norm for the embedding layer, which we found helped in training stability. At a scale of 100B+, even marginal improvements in training abilities are welcome. We settle for a 250k-sized vocabulary. We ultimately decided on the size 250680 for GPU parallelism and other reasons. The tokenizer is learned using byte pair encoding for standard reasons and also because it maximizes vocabulary sharing between languages.

5.2.2 Model Training

As we spoke earlier, the hardware for training the model was given to us in the form of Jean Zay a French supercomputer. BLOOM was trained on about 48 nodes of 8 Nvidia A100 80GB GPUs each, in total about 1 million hours in A100 compute. BLOOM took about three and a half months to train. 4 nodes were maintained in reserve in case a few faulty nodes went under.

The Framework for training Large Language models is possible through Megatron-Deepspeed [96]. The basic idea can be seen in Fig: 5.6. It consists of using the repository created for training MegatronLM by Nvidia [95] and a framework that helps for distributing loads over GPUs through different parallelism components and has the ZeRO Optimizer. This allows us essentially to achieve something called 3D parallelism.

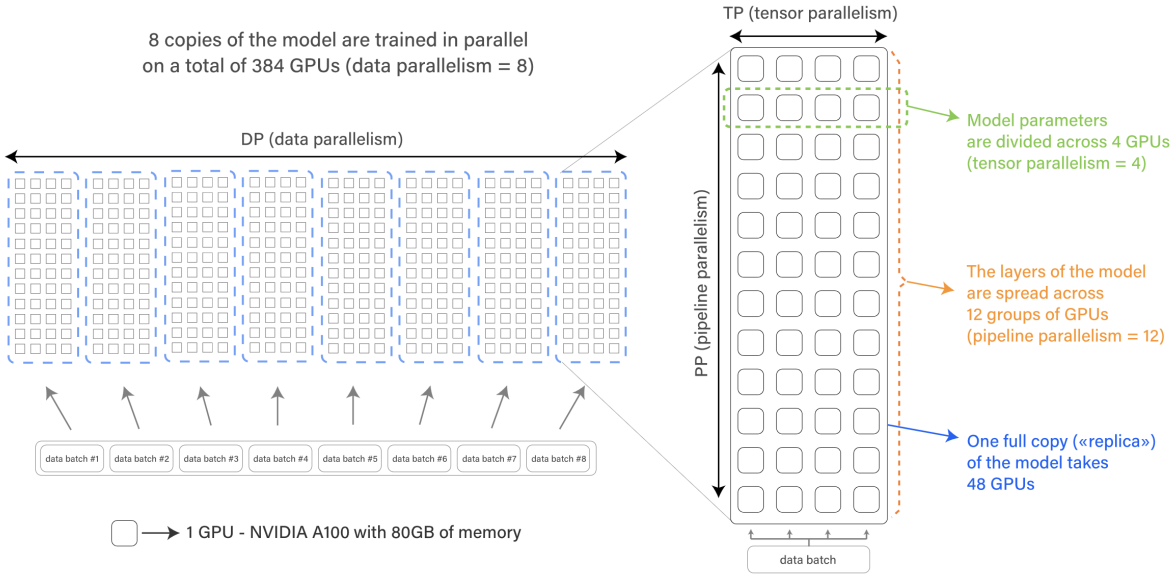

Figure 5.6 Using 3D parallelism for training BLOOM

3D parallelism is:

- Data parallelism - In this, the model is replicated multiple times and it's kept on a different device and fed different parts of the data, the models are synced at the end of training steps.
- Tensor parallelism - Also called as intra-layer parallelism, we partition individual layers of the model and put them across multiple GPUs.
- Pipeline parallelism - Also called as vertical parallelism, We split up layers in the model on multiple devices.

We were ultimately able to achieve 156 TeraFlops (TFLOPS) on our fastest configuration. 6 Different sizes of the model are trained as seen in table 5.1.

5.2.3 BLOOMZ

Recent work like T_0 [86], and FLAN [106] has shown the advantages of fine-tuning language models to grant them additional capabilities. This is done because the language model is explicitly trained with a supervised dataset in a multitask fashion. The training involves training on multiple tasks at once. These tasks are specified in the form of prompts.

Why do prompts matter? In [11], it was shown that large language models possess out-of-the-box capabilities to do some tasks. While earlier pre-trained language models were elaborately trained on the target task data, they found that when given a document and at

the end appending "tl;dr" or "please summarize this document," the model then generates a summary. The appending sentence is known as a prompt, and this act is called prompting the model. When one feeds the model a single document/input and asks for an answer, it is known as zero-shot prompting, while multiple documents with some examples can also be given to achieve few-shot prompting. All of this happens within the input context size of the model and does not require gradient training. This nature of large language models and their ability to process tasks and the ability to be prompted is known as "In-Context learning" in popular parlance.

In [86], the authors set out to find if there are any other ways of repurposing or equipping the model with better zero-shot capabilities in the form of prompt processing. They find that multitasking fine-tuning works very well. As mentioned in the training dataset section, five prompts for 170 datasets were collected. All of these are used to fine-tune BLOOM to give rise to BLOOMZ [64]. The models are fine-tuned for 13 billion tokens.

5.3 Evaluation

Due to the recent rise of large language model capabilities, the primary way of testing these models has shifted from fine-tuning to "In-context learning" based evaluation. Therefore we evaluate BLOOM on 0 shot and few shot contexts.

We evaluate the BLOOM model on superGLUE, machine translation, and summarization in zero-shot and one-shot results. We also measure it with the BLOOMZ model, i.e., after multitask fine-tuning. For an example of what kind of prompts are used for machine translation, refer to table 5.2.

The models we compare with the following baseline models wherever appropriate:

- mGPT [94] is a model with a GPT-style architecture, and it is trained in more than 50 languages.
- GPT-NEO [10], [103] is a set of GPT style models trained by eleuther ai.
- To [86] is a multitask finetuned model trained with the architecture of T5 [80] with prompted datasets
- OPT [112] is a large language model trained on a bunch of datasets and has parameters as high as 175B
- XGLM [55] is a model with GPT-style architecture trained in about 100 languages.
- M2M [28] a multilingual model which was exclusively trained to translate between languages, it was about 100 languages.
- Codex [16] a GPT style architecture model trained on code.

All datasets had about five prompts written for them, and all evaluations is run over all the prompts. The choice of using 5 prompts is to study prompt variance and ensure that we do not prompt engineer too much and show the results of the model grounded in reality. We report the average of the five prompts whenever it is a final number.

SuperGLUE We test on a subset of this benchmark using the tasks Ax-b, Ax-g, Boolq, CB (the Commitment bank), RTE, Wic, WSC. These are English-only tasks but are standard datasets to evaluate large language models.

Machine Translation We evaluate on WMT14 en $\leftrightarrow$ fr and en $\leftrightarrow$ hi and Diabla [6] as well. We evaluate using SacreBleu as an implementation of BLEU. While previously task-specific architectures were evaluated by beam search, we use greedy decoding to generate from the model.

Summarization We evaluate Wikilingua from the gem benchmark. We focus on 1-shot results because initial studies showed that 0-shot results for this task were not great. To test for BLOOM's multilingual ability, we exclusively test "language $\leftrightarrow$ language pairs". We choose nine languages for this ar (Arabic), en (English), es (Spanish), fr (French), hi (Hindi), id (Indonesian), pt (Portuguese), vi (Vietnamese), zh (Chinese). Evaluating multilingual summarization is extremely difficult because of non-standard evaluation practices. While we report rouge scores, we use the sentence piece tokenizer instead. We use the exact decoding and generation process as machine translation.

The performance on SuperGLUE is shown in Fig 5.7. We see that To is consistently better than most of the other models, and we attribute this to its being good at prompt evaluation since it's trained explicitly on multitask prompt-tuned datasets. The multiple circles in each model show the different prompts. Some prompts show much better performance than others, as the chart shows. As models go from 0-shot performance to 1-shot performance, something to observe is that prompt variance comes down, and the performance on the task is more uniform despite prompt variance. When additional model ablations are done over BLOOM and OPT models, we show that both families of large language models improve slightly with scale.

The performance on machine translation is shown in Fig 5.3. As mentioned earlier, the example prompts for the machine translation task are shown in Fig 5.2. There is a massive leap from 0 shot to 1 shot, as we find that in 0 shot, the model has usual generation issues like repeatability and inability to generate coherently. Most of these issues are alleviated in 1-shot, and with the right prompt, it performs comparatively but lags behind dedicated, supervised models like M2M(e.g., 43.8 BLEU when going from English to French). The two major problems observed in this case are the inability to produce the correct language and overgeneration. The former problem could perhaps be resolved by clever prompt techniques which flag the language to the model and steer it in the proper direction.

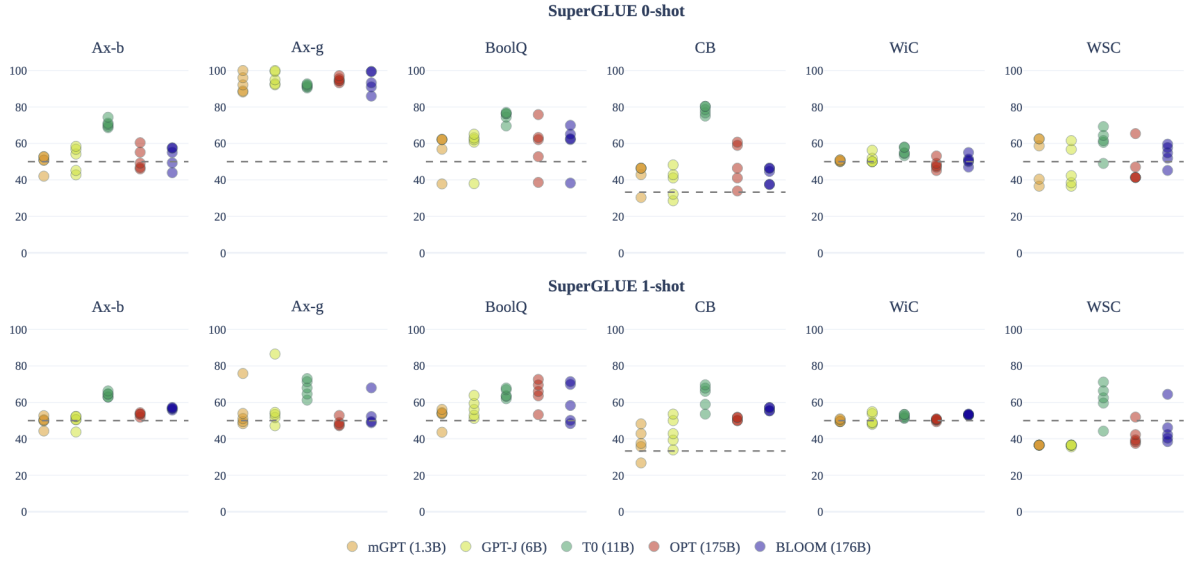

Figure 5.7 Performance of models on SuperGLUE. Left to right the models are mGPT-1.3B, GPT-J-6B, To-11B, OPT-175B, BLOOM-176B

The performance on summarization is consistently better than that of OPT. It is because of the multilingual training of BLOOM, which also increases with parameter count. The continual growth of performance in multilingual languages also suggests that there is more performance that can be gained from training even bigger models. The performance on code generation is not substantially better than codex models even though coding data is 11% of the entire training corpus of BLOOM. We also see no improvement in code generation even after multitask fine-tuning, which might be because no tasks specifically handle code generation in the dataset.

5.4 Conclusion

In this chapter, We saw the development of BLOOM, the largest multilingual model, and study it through a lens of data collection, training, and evaluation and understand how large language models are evaluated today. While there are a variety of improvements that can be made to get the best out of the model, large language models are much better equipped to learn/fine-tune because of in-context learning.

Hyperparameter (↓)	BLOOM-560M	BLOOM-1.1B	BLOOM-1.7B	BLOOM-3B	BLOOM-7.1B	BLOOM
Architecture hyperparameters						
Parameters	559M	1,065M	1,722M	3,003M	7,069M	176,247M
Precision			`float16`			`bfloat16`
Layers	24	24	24	30	30	70
Hidden dim.	1024	1536	2048	2560	4096	14336
Attention heads	16	16	16	32	32	112
Vocab size			250,680			250,680
Sequence length			2048			2048
Activation			`GELU`			`GELU`
Position emb.			`Alibi`			`Alibi`
Tied emb.			`True`			`True`
Pretraining hyperparameters						
Global Batch Size	256	256	512	512	512	2048
Learning rate	3.0e-4	2.5e-4	2e-4	1.6e-4	1.2e-4	6e-5
Total tokens			341B			366B
Warmup tokens			375M			375M
Decay tokens			410B			410B
Decay style			`cosine`			`cosine`
Min. learning rate			1e-5			6e-6
Adam (β_1, β_2)			(0.9, 0.95)			(0.9, 0.95)
Weight decay			1e-1			1e-1
Gradient clipping			1.0			1.0
Multitask finetuning hyperparameters						
Global Batch Size	1024	1024	2048	2048	2048	2048
Learning rate	2.0e-5	2.0e-5	2.0e-5	2.0e-5	2.0e-5	2.0e-5
Total tokens			13B			13B
Warmup tokens			0			0
Decay style			`constant`			`constant`
Weight decay			1e-4			1e-4

Table 5.1 BLOOM & BLOOMZ Training Hyperparameters.

Prompt name	Prompt	Target
a_good_translation-source+target	Given the following source text: [source sentence], a good L2 translation is	[target sentence]
gpt3-target	What is the L2 translation of the sentence: [source sentence]?	[target sentence]
version-target	if the original version says [source sentence]; then the L2 version should say:	[target sentence]
xglm-source+target	L1: [source sentence] = L2:	[target sentence]

Table 5.2 MT prompts examples

Prompt	en → fr		fr → en		en → hi		hi → en	
Shots	0	1	0	1	0	1	0	1
a_good_translation-source+target	15.38	36.39	14.15	36.56	1.90	14.49	10.19	24.60
gpt3-target	7.90	32.55	12.73	33.14	0.26	6.51	0.66	9.98
version-target	21.96	34.22	26.79	35.42	1.96	13.95	11.48	25.80
xglm-source+target	14.91	27.83	15.52	34.51	6.80	13.62	12.05	25.04

Table 5.3 WMT'14 zero- and one-shot results (BLEU) for BLOOM-176B. The prompts used are described in Table 5.2.

Chapter 6

Conclusion

In this thesis, We study the evaluation of large and pre-trained language models through multiple lenses.

First, learning about the task of summarization and task-specific architectures, which led to initial development, and how pre-trained language models helped improve performance. We choose long-form summarization in the scientific domain to study the generalization abilities and domain adaptation of pre-trained language models. BERT models can be enhanced through methods like intermediate finetuning on the target task with the help of high-resource datasets, which can help it perform better on low-resource datasets and cross-domain tasks. While they aren't perfect, we also see that they provide immediate improvements over task-specific architectures and can even be data efficient when it comes to low-resource data in the target domain. Understanding that a pre-trained language model like BERT is not the final step and further task performance would need better language understanding abilities and evaluation criteria, we move to further work on evaluation and training better models.

We then look at the initial development of NLP benchmarks and understand their original motivations. Recognizing that natural language understanding benchmarks are lacking in nature and do not holistically assess current models, which are much more capable and also have generation capabilities, we study the current state of the art in the generation benchmarks and see what choices have to be made and nuances that have to be considered and kept in mind while designing these. While important, we know that leaderboard benchmarks lack nuance in model capabilities and hence develop a more nuanced dataset with broader evaluation criteria to understand models better.

We finally look at BLOOM, a large language model of 176 billion parameters trained on over 40 languages. Data collection is one of the more critical steps in large language models. We first understand the data collection process for 49 languages and how it was sourced. We then look into how most Large language models are trained today and understand the training setup specific to BLOOM. Then we see that most evaluation today happens through

in-context learning, and our earlier methods of intermediate finetuning are now achieved in context with the help of few-shot learning. The evaluation of this model, while broad, still has nuances that can be explored. We also learn that while in-context learning makes learning much more data-efficient, We also learn that at the same time, prompt variance impacts the quality of models a lot, and that has to be kept in mind when writing prompts for languages, and hence the prompt design must be carefully done to get the best out of large language models.

6.1 Future Work

While it was possible to evaluate BLOOM because the model is open source, most newer models like GPT-3 [11], GPT-4 [69] and PALM-2 [2] can only be evaluated through APIs. While doing the internal calculation, we find that evaluating GPT-4 on Superglue costs about 5000$, which also hints towards evaluation becoming costlier; this means that there is a dire need for evaluation benchmarks that cover the same surface area as earlier benchmarks but with a fraction of the data points since model finetuning through in-context learning has also made evaluation sample efficient. Most of these benchmarks were developed in an era when task-specific architectures were trained in traditional machine learning methods of having training, development, and test sets which required large high-quality datasets for models to work well, which are no longer true today.

At the same time, while evaluation benchmarks move forward, there is a lot we do not understand about large language models and their capabilities and the best way to bring these out. A lot of research is needed toward better prompt design or removing it altogether by automating the search for better prompts.

In closing, I hope reading this thesis has given you some insight into the world of evaluating language models and the various nuances to keep in mind. We have a long way to go.

Related Publications

The Effect of Pretraining on Extractive Summarization for Scientific Documents. - Yash Gupta, **Pawan Sasanka Ammanamanchi**, Shikha Bordia, Arjun Manoharan, Deepak Mittal, Ramakanth Pasunuru, Manish Shrivastava, Maneesh Singh, Mohit Bansal, and Preethi Jyothi. - In Proceedings of the Second Workshop on Scholarly Document Processing 2021 - NAACL 2021

The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics. - Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, **Pawan Sasanka Ammanamanchi**, Anuoluwapo Aremu, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna-Adriana Clinciu, Dipanjan Das, Kaustubh Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Chinenye Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, et al. - In Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics GEM 2021 - ACL 2021

BLOOM: A 176B-Parameter Open-Access Multilingual Language Model - Scao, T.L., Fan, A., Akiki, C., Pavlick, E., Ili'c, S., Hesslow, D., Castagn'e, R., Luccioni, A.S., Yvon, F., Gallé, M., Tow, J., Rush, A.M., Biderman, S.R., Webson, A., **Ammanamanchi, P.S.**, Wang, T., Sagot, B., Muennighoff, N., del Moral, A.V., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Ortiz Suarez, P., Sanh, V., Laurenccon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel, C. and et al. . ArXiv, abs/2211.05100. *Under Review*

Bibliography

[1] F. Alva-Manchego, L. Martin, A. Bordes, C. Scarton, B. Sagot, and L. Specia. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679, Online, July 2020. Association for Computational Linguistics.

[2] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. T. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. Clark, L. E. Shafey, Y. Huang, K. S. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. 'Abrego, J. Ahn, J. Austin, P. Barham, J. A. Botha, J. Bradbury, S. Brahma, K. M. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crépy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. C. D'iaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. García, S. Gehrmann, L. González, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. R. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. H. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M.-L. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z.-Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. Ros, A. Roy, B. Saeta, R. Samuel, R. M. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. W. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu. Palm 2 technical report. *ArXiv*, abs/2305.10403, 2023.

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[4] S. Banerjee and A. 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*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[5] R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, volume 6, pages 6–4. Venice, 2006.

[6] R. Bawden, S. Rosset, T. Lavergne, and E. Bilinski. Diabla: a corpus of bilingual spontaneous written dialogues for machine translation. *Language Resources and Evaluation*, 55:635 – 660, 2019.

[7] I. Beltagy, K. Lo, and A. Cohan. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[8] Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2000.

[9] L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

[10] S. Black, L. Gao, P. Wang, C. Leahy, and S. R. Biderman. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. 2021.

[11] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[12] I. Cachola, K. Lo, A. Cohan, and D. Weld. TLDR: Extreme summarization of scientific documents. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4766–4777, Online, Nov. 2020. Association for Computational Linguistics.

[13] A. Celikyilmaz, A. Bosselut, X. He, and Y. Choi. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[14] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics.

[15] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In A. Beygelzimer,

Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[16] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. Ponde, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. W. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, I. Babuschkin, S. A. Balaji, S. Jain, A. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *ArXiv*, abs/2107.03374, 2021.

[17] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

[18] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020.

[19] A. Cohan, F. Dernoncourt, D. S. Kim, T. Bui, S. Kim, W. Chang, and N. Goharian. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[20] I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer, 2005.

[21] M.-C. de Marneffe, M. Simons, and J. Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. 2019.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[23] W. B. Dolan and C. Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[25] E. Durmus, H. He, and M. Diab. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online, July 2020. Association for Computational Linguistics.

[26] O. Dušek, D. M. Howcroft, and V. Rieser. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426, Tokyo, Japan, Oct.–Nov. 2019. Association for Computational Linguistics.

[27] O. Dušek and F. Jurčíček. Neural generation for Czech: Data and baselines. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 563–574, Tokyo, Japan, Oct.–Nov. 2019. Association for Computational Linguistics.

[28] A. Fan, S. Bhosale, H. Schwenk, Z. Ma, A. El-Kishky, S. Goyal, M. Baines, O. Celebi, G. Wenzek, V. Chaudhary, N. Goyal, T. Birch, V. Liptchinsky, S. Edunov, M. Auli, and A. Joulin. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48, 2021.

[29] L. Gao, J. Tow, S. Biderman, C. Lovering, J. Phang, A. Thite, Fazz, N. Muennighoff, T. Wang, sdtblck, tttyuntian, researcher2, Z. Kasner, K. Almubarak, J. Hsu, P. S. Ammanamanchi, D. Groeneveld, E. Tang, C. Foster, kkawamu1, xagi dev, uyhcire, A. Zou, B. Wang, J. Clive, igoro, K. Wang, N. Kross, F. Milo, and silentvox. Eleutherai/lm-evaluation-harness: v0.3.0, Dec. 2022.

[30] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini. Creating training corpora for NLG micro-planners. In R. Barzilay and M. Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 179–188. Association for Computational Linguistics, 2017.

[31] S. Gehrmann, T. Adewumi, K. Aggarwal, P. S. Ammanamanchi, A. Aremu, A. Bosselut, K. R. Chandu, M.-A. Clinciu, D. Das, K. Dhole, W. Du, E. Durmus, O. Dušek, C. C. Emezue, V. Gangal, C. Garbacea, T. Hashimoto, Y. Hou, Y. Jernite, H. Jhamtani, Y. Ji, S. Jolly, M. Kale, D. Kumar, F. Ladhak, A. Madaan, M. Maddela, K. Mahajan, S. Mahamood, B. P. Majumder, P. H. Martins, A. McMillan-Major, S. Mille, E. van Miltenburg, M. Nadeem, S. Narayan, V. Nikolaev, A. Niyongabo Rubungo, S. Osei, A. Parikh, L. Perez-Beltrachini, N. R. Rao, V. Raunak, J. D. Rodriguez, S. Santhanam, J. Sedoc, T. Sellam, S. Shaikh, A. Shimorina, M. A. Sobrevilla Cabezudo, H. Strobelt, N. Subramani, W. Xu, D. Yang, A. Yerukola, and J. Zhou. The GEM benchmark: Natural language generation, its evaluation and metrics. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 96–120, Online, Aug. 2021. Association for Computational Linguistics.

[32] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*,

pages 1–9. Association for Computational Linguistics, 2007.

[33] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang. Conformer: Convolution-augmented transformer for speech recognition. *ArXiv*, abs/2005.08100, 2020.

[34] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online, July 2020. Association for Computational Linguistics.

[35] T. B. Hashimoto, H. Zhang, and P. Liang. Unifying human and statistical evaluation for natural language generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1689–1701, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[36] P. He, J. Gao, and W. Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing, 2021.

[37] K. M. Hermann, T. Kociský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015.

[38] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, O. Vinyals, J. Rae, and L. Sifre. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030. Curran Associates, Inc., 2022.

[39] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[40] D. Huang, L. Cui, S. Yang, G. Bao, K. Wang, J. Xie, and Y. Zhang. What have we achieved on text summarization? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 446–469, Online, Nov. 2020. Association for Computational Linguistics.

[41] S. Iyer, N. Dandekar, and K. Csernai. First quora dataset release: Question pairs, 2017.

[42] C. Jiang, M. Maddela, W. Lan, Y. Zhong, and W. Xu. Neural CRF model for sentence alignment in text simplification. In D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7943–7960. Association for Computational Linguistics, 2020.

[43] J. Kaplan, S. McCandlish, T. J. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

[44] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth. Looking beyond the surface:a challenge set for reading comprehension over multiple sentences. In *Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

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

[46] F. Ladhak, E. Durmus, C. Cardie, and K. McKeown. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048, Online, Nov. 2020. Association for Computational Linguistics.

[47] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

[48] H. Laurençon, L. Saulnier, T. Wang, C. Akiki, A. V. del Moral, T. L. Scao, L. V. Werra, C. Mou, E. G. Ponferrada, H. Nguyen, J. Frohberg, M. Šaško, Q. Lhoest, A. McMillan-Major, G. Dupont, S. Biderman, A. Rogers, L. B. allal, F. D. Toni, G. Pistilli, O. Nguyen, S. Nikpoor, M. Masoud, P. Colombo, J. de la Rosa, P. Villegas, T. Thrush, S. Longpre, S. Nagel, L. Weber, M. R. Muñoz, J. Zhu, D. V. Strien, Z. Alyafeai, K. Almubarak, V. M. Chien, I. Gonzalez-Dios, A. Soroa, K. Lo, M. Dey, P. O. Suarez, A. Gokaslan, S. Bose, D. I. Adelani, L. Phan, H. Tran, I. Yu, S. Pai, J. Chim, V. Lepercq, S. Ilic, M. Mitchell, S. Luccioni, and Y. Jernite. The bigscience ROOTS corpus: A 1.6TB composite multilingual dataset. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[49] T. Le Scao, T. Wang, D. Hesslow, S. Bekman, M. S. Bari, S. Biderman, H. Elsahar, N. Muennighoff, J. Phang, O. Press, C. Raffel, V. Sanh, S. Shen, L. Sutawika, J. Tae, Z. X. Yong, J. Launay, and I. Beltagy. What language model to train if you have one million GPU hours? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 765–782, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[50] H. Levesque, E. Davis, and L. Morgenstern. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2012.

[51] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for*

Computational Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[52] Q. Lhoest, A. Villanova del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Šaško, G. Chhablani, B. Malik, S. Brandeis, T. Le Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. Rush, and T. Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[53] B. Y. Lin, W. Zhou, M. Shen, P. Zhou, C. Bhagavatula, Y. Choi, and X. Ren. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online, Nov. 2020. Association for Computational Linguistics.

[54] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[55] X. V. Lin, T. Mihaylov, M. Artetxe, T. Wang, S. Chen, D. Simig, M. Ott, N. Goyal, S. Bhosale, J. Du, R. Pasunuru, S. Shleifer, P. S. Koura, V. Chaudhary, B. O'Horo, J. Wang, L. Zettlemoyer, Z. Kozareva, M. Diab, V. Stoyanov, and X. Li. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[56] T. Linzen. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, Online, July 2020. Association for Computational Linguistics.

[57] D. Liu, Y. Yan, Y. Gong, W. Qi, H. Zhang, J. Jiao, W. Chen, J. Fu, L. Shou, M. Gong, P. Wang, J. Chen, D. Jiang, J. Lv, R. Zhang, W. Wu, M. Zhou, and N. Duan. GLGE: A new general language generation evaluation benchmark. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 408–420, Online, Aug. 2021. Association for Computational Linguistics.

[58] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

[59] Y. Liu and M. Lapata. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Con-*

ference on Natural Language Processing (EMNLP-IJCNLP), pages 3730–3740, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[60] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[61] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, July 2020. Association for Computational Linguistics.

[62] A. McMillan-Major, S. Osei, J. D. Rodriguez, P. S. Ammanamanchi, S. Gehrmann, and Y. Jernite. Reusable templates and guides for documenting datasets and models for natural language processing and generation: A case study of the HuggingFace and GEM data and model cards. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 121–135, Online, Aug. 2021. Association for Computational Linguistics.

[63] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*, 2013.

[64] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. R. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z. X. Yong, H. Schoelkopf, X. Tang, D. R. Radev, A. F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff, and C. Raffel. Crosslingual generalization through multitask finetuning. *ArXiv*, abs/2211.01786, 2022.

[65] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[66] L. Nan, D. Radev, R. Zhang, A. Rau, A. Sivaprasad, C. Hsieh, X. Tang, A. Vyas, N. Verma, P. Krishna, Y. Liu, N. Irwanto, J. Pan, F. Rahman, A. Zaidi, M. Mutuma, Y. Tarabar, A. Gupta, T. Yu, Y. C. Tan, X. V. Lin, C. Xiong, R. Socher, and N. F. Rajani. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online, June 2021. Association for Computational Linguistics.

[67] S. Narayan, S. B. Cohen, and M. Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics.

[68] J. Novikova, O. Dušek, and V. Rieser. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany, Aug. 2017. Association for Computational Linguistics.

[69] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[70] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[71] A. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online, Nov. 2020. Association for Computational Linguistics.

[72] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, 2018.

[73] J. Pennington, R. Socher, and C. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[74] L. Perez-Beltrachini and C. Gardent. Analysing data-to-text generation benchmarks. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 238–242, Santiago de Compostela, Spain, Sept. 2017. Association for Computational Linguistics.

[75] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[76] M. T. Pilehvar and J. Camacho-Collados. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[77] O. Press, N. Smith, and M. Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022.

[78] A. Radford and K. Narasimhan. Improving language understanding by generative pre-training. 2018.

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

[80] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jan 2020.

[81] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, Nov. 2016. Association for Computational Linguistics.

[82] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696, 2020.

[83] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1195, 2016.

[84] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online, July 2020. Association for Computational Linguistics.

[85] M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011.

[86] V. Sanh, A. Webson, C. Raffel, S. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, T. L. Scao, S. Biderman, L. Gao, T. Wolf, and A. M. Rush. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022.

[87] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilic, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, J. Tow, A. M. Rush, S. Biderman, A. Webson, P. S. Ammanamanchi, T. Wang, B. Sagot, N. Muennighoff, A. V. del Moral, O. Ruwase, R. Bawden, S. Bekman, A. McMillan-Major, I. Beltagy, H. Nguyen, L. Saulnier, S. Tan, P. O. Suarez, V. Sanh, H. Laurençon, Y. Jernite, J. Launay, M. Mitchell, C. Raffel, A. Gokaslan, A. Simhi, A. Soroa, A. F. Aji, A. Alfassy, A. Rogers, A. K. Nitzav, C. Xu, C. Mou, C. Emezue, C. Klamm, C. Leong, D. van Strien, D. I. Adelani, and et al. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100, 2022.

[88] T. Scialom, P.-A. Dray, S. Lamprier, B. Piwowarski, and J. Staiano. MLSUM: The multilingual summarization corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067, Online, Nov. 2020. Association for Computational Linguistics.

[89] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguis-*

tics (Volume 1: Long Papers), pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[90] T. Sellam, D. Das, and A. P. Parikh. Bleurt: Learning robust metrics for text generation. In *Proceedings of ACL*, 2020.

[91] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*, 2017.

[92] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, IL, 1949.

[93] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[94] O. Shliazhko, A. Fenogenova, M. Tikhonova, V. Mikhailov, A. Kozlova, and T. Shavrina. mgpt: Few-shot learners go multilingual. *ArXiv*, abs/2204.07580, 2022.

[95] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *ArXiv*, abs/1909.08053, 2019.

[96] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. A. Korthikanti, E. Zhang, R. Child, R. Y. Aminabadi, J. Bernauer, X. Song, M. Shoeybi, Y. He, M. Houston, S. Tiwary, and B. Catanzaro. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *ArXiv*, abs/2201.11990, 2022.

[97] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[98] J. Su, Y. Lu, S. Pan, B. Wen, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding, 2021.

[99] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[100] T. Vu, T. Wang, T. Munkhdalai, A. Sordoni, A. Trischler, A. Mattarella-Micke, S. Maji, and M. Iyyer. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online, Nov. 2020. Association for Computational Linguistics.

[101] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In

H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[102] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics.

[103] B. Wang and A. Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

[104] T. Wang, A. Roberts, D. Hesslow, T. L. Scao, H. W. Chung, I. Beltagy, J. Launay, and C. Raffel. What language model architecture and pretraining objective works best for zero-shot generalization? In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 22964–22984. PMLR, 17–23 Jul 2022.

[105] A. Warstadt, A. Singh, and S. R. Bowman. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

[106] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

[107] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. Survey Certification.

[108] A. Williams, N. Nangia, and S. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics, 2018.

[109] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics.

[110] J. Zhang, Y. Zhao, M. Saleh, and P. Liu. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR, 13–18 Jul 2020.

[111] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

[112] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068, 2022.

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

[114] M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online, July 2020. Association for Computational Linguistics.