BloombergGPT: A Large Language Model for Finance

Shijie Wu[1,∗], **Ozan İrsoy**[1,∗], **Steven Lu**[1,∗], **Vadim Dabravolski**[1], **Mark Dredze**[1,3], **Sebastian Gehrmann**[1], **Prabhanjan Kambadur**[1], **David Rosenberg**[2], **Gideon Mann**[1]

[1] **Bloomberg, New York, NY USA**

[2] **Bloomberg, Toronto, ON Canada**

[3] **Computer Science, Johns Hopkins University, Baltimore, MD USA**

Abstract

The use of NLP in the realm of financial technology is broad and complex, with applications ranging from sentiment analysis and named entity recognition to question answering. Large Language Models (LLMs) have been shown to be effective on a variety of tasks; however, no LLM specialized for the financial domain has been reported in literature. In this work, we present BloombergGPT, a 50 billion parameter language model that is trained on a wide range of financial data. We construct a 363 billion token dataset based on Bloomberg's extensive data sources, perhaps the largest domain-specific dataset yet, augmented with 345 billion tokens from general purpose datasets. We validate BloombergGPT on standard LLM benchmarks, open financial benchmarks, and a suite of internal benchmarks that most accurately reflect our intended usage. Our mixed dataset training leads to a model that outperforms existing models on financial tasks by significant margins without sacrificing performance on general LLM benchmarks. Additionally, we explain our modeling choices, training process, and evaluation methodology. We release Training Chronicles (Appendix C) detailing our experience in training BloombergGPT.

Contents

∗. Co-first authors. Corresponding author email: `gmann16@bloomberg.net`

1 Introduction

The release of GPT-3 in 2020 (Brown et al., 2020) demonstrated the powerful benefits of training very large auto-regressive language models (LLMs). GPT-3 had 175 billion parameters, a hundredfold increase over the previous GPT-2 model, and did remarkably well across a wide range of now popular LLM tasks, including reading comprehension, open-ended question answering, and code generation. This performance has been replicated across several other models (Chowdhery et al., 2022; Scao et al., 2022; Zhang et al., 2022a). Furthermore, evidence suggests that large models exhibit emergent behaviors; growth allows them to acquire abilities not present in smaller models (Wei et al., 2022a). A notable example of emergent behavior is the ability to perform tasks via few-shot prompting, where a model can learn a task from just a few examples. This ability improves well-above random as we increase the size of language models. Broadly speaking, few-shot prompting dramatically expands the range of tasks supported by models and lowers the barrier to entry for users seeking automation for new language tasks.

After GPT-3, models grew in size to 280 billion (Gopher, Rae et al., 2021), 540 billion (PaLM, Chowdhery et al., 2022), and 1 trillion parameters (Megatron, Korthikanti et al., 2022). Work also explored other important aspects of achieving a high-performing LLM, such as different training objectives (Tay et al., 2022b), multilingual models (Scao et al., 2022), more efficient and smaller models (Black et al., 2022), and finding data and parameter-efficient training sizes (Hoffmann et al., 2022).

These efforts have almost exclusively focused on general LLMs, trained on datasets that cover a broad range of topics and domains. While these have included some datasets for specialized domains (e.g., code (Chen et al., 2021a) or biomedical articles (Gao et al., 2021)) the focus has been on building LLMs with broad capabilities. Recent efforts training models using only domain-specific data have yielded models that, while much smaller, beat general purpose LLMs on tasks within those domains, such as science (Taylor et al., 2022) and medicine (Bolton et al., 2023; Luo et al., 2022; Lehman et al., 2023). These findings motivate further development of models focused on specific domains.

Financial Technology (FinTech) is a large and growing area with NLP technologies having an increasingly important role (Xing et al., 2018; Fisher et al., 2016; Dredze et al., 2016). Financial NLP tasks (Shah et al., 2022) include sentiment analysis (Araci, 2019), named entity recognition (Salinas Alvarado et al., 2015), news classification (Sinha and Khandait, 2020), and question answering (Chen et al., 2021b, 2022). While the range of tasks is similar to those found in general NLP benchmarks, the complexity and terminology of the financial domain warrant a domain-specific system. For all of the reasons generative LLMs are attractive in general – few-shot learning, text generation, conversational systems, etc. – it would be valuable to have a LLM focused on the financial domain. While there are masked language models tuned for the financial domain (Araci, 2019), no LLM has been tuned for or evaluated on tasks for this domain.

1.1 BloombergGPT

We train BloombergGPT, a 50 billion parameter language model that supports a wide range of tasks within the financial industry. Rather than building a general-purpose LLM, or a small LLM exclusively on domain-specific data, we take a mixed approach. General models cover many domains, are able to perform at a high level across a wide variety of tasks, and obviate the need for specialization during training time. However, results from existing domain-specific models show that general models cannot replace them. At Bloomberg, we support a very large and diverse set of tasks, well served by a general model, but the vast majority of our applications are within the financial domain, better served by a specific model. For that reason, we set out to build a model that achieves best-in-class results on financial benchmarks, while also maintaining competitive performance on general-purpose LLM benchmarks.

We achieve this goal by constructing the largest domain-specific dataset yet, drawing on existing data creation, collection, and curation resources at Bloomberg. As Bloomberg is primarily a financial data company, our data analysts have collected and curated financial language documents over the span of forty years. We have extensive archives of financial data that cover a range of topics, with careful tracking of data sources and usage rights. We add this data to public datasets to create a large training corpus with over 700 billion tokens. Using a portion of this training corpus, we train a BLOOM-style, 50 billion parameter model designed based on guidelines from Hoffmann et al. (2022) and Le Scao et al. (2022). We validate the model on standard LLM benchmarks, open financial benchmarks, and a suite of Bloomberg-internal benchmarks that most accurately reflect our intended use cases. Our results demonstrate that our mixed training approach leads to a model that vastly outperforms existing models on in-domain financial tasks while being on par or better on general NLP benchmarks.

1.2 Broader Contributions

Beyond the construction of a LLM for financial data, our goal is to contribute to the broader research community. Specifically, our experience documented in this paper provides evidence that further develops the community's understanding of several open questions in the literature.

Domain-specific LLMs. The few existing domain-specific LLMs are trained exclusively on domain-specific data sources (Luo et al., 2022; Bolton et al., 2023; Taylor et al., 2022), or adapt a very large general purpose model to domain-specific tasks (Singhal et al., 2022; Lewkowycz et al., 2022). Our alternative approach – training an LLM on both domain-specific and general data sources – has not been studied so far. The resulting model does very well on domain-specific tasks, but also maintains strong performance on general-purpose benchmarks.

Training data. Nearly all language models rely in large part on web-scraped data, such as C4 (Raffel et al., 2020) and The Pile (Gao et al., 2021) (which includes OpenWebText2). This data may be cleaned or subsetted in various ways before use (Touvron et al., 2023; Rae et al., 2020; Scao et al., 2022; Jernite et al., 2022), but issues of data duplication (Carlini et al., 2020) and toxic language remain (Welbl et al., 2021). Our training data is unusual for LLM training in that it includes a significant amount of curated and prepared data from reliable sources.

Evaluation. LLM evaluation remains a challenging and evolving problem (Gehrmann et al., 2022; Goyal et al., 2022), with new benchmarks trying to standardize evaluation across models (Liang et al., 2022; Srivastava et al., 2022). However, for domain-specific tasks, there remains a mismatch between evaluation and actual use cases. Evaluations are built on available datasets and not necessarily on how the model will be used in practice. We provide results on both public financial NLP benchmarks (Shah et al., 2022; Chen et al., 2021b) as well as a selection of internal Bloomberg tasks, which are better aligned with our intended use cases and directly evaluate our model's ability to perform tasks of interest.

Model Size. Early LLMs made a single training pass over a corpus of 200-400 billion tokens (Brown et al., 2020) and Hoffmann et al. (2022) posited that models were undertrained, instead focusing on training smaller models with more data, a strategy most recently employed by Touvron et al. (2023). We select a model size motivated by Hoffmann et al. (2022) and train a 50 billion parameter model on 569 billion tokens from our corpus of over 700 billion tokens to produce a model that is competitive with larger models.

Tokenizer. After assembling training data, the critical step of tokenization transforms the text into a format suitable for the language model. The importance of this step is often overlooked (Mielke et al., 2021), and many older LLMs use the same tokenizer and vocabulary, meaning that we have little evidence to support other tokenizers. We take a different approach and use a Unigram model instead of greedy merge-based sub-word tokenizers since it saves probabilities allowing for smarter tokenization at inference time (Kudo, 2018).

Model Building Challenges. GPT-3 and subsequent models were the work of large teams and required an enormous amount of computation. Initial work to reproduce these results, such as OPT (Zhang et al., 2022a), did not match the performance of the original model. With the release of each subsequent model, the community's understanding, experience, and software tools increase. In developing BloombergGPT, we benefited from existing code developed as part of the BLOOM effort (Scao et al., 2022), showing that a moderately sized team can produce a competitive model on domain-specific data. We describe our experiences training BloombergGPT in detail to support future training efforts and address each of the above topics.

2 Dataset

To train BloombergGPT, we construct "FinPile", a comprehensive dataset consisting of a range of English financial documents including news, filings, press releases, web-scraped financial documents, and social media drawn from the Bloomberg archives. These documents have been acquired through our business process over the past two decades. We augment FinPile with public data widely used to train LLMs. The result is a training corpus that is roughly half domain-specific text and half general-purpose text. For a breakdown of the full training set, see Table 1. To improve data quality, we de-duplicate each dataset (The Pile, C4, Wikipedia, FinPile) according to Lee et al. (2022a); as a side-effect, the statistics reported in Table 1 might be different from those reported in other papers.

Dataset	Docs 1e4	C/D	Chars 1e8	C/T	Toks 1e8	T%
FinPile	*175,886*	*1,017*	*17,883*	*4.92*	*3,635*	*51.27%*
Web	158,250	933	14,768	4.96	2,978	42.01%
News	10,040	1,665	1,672	4.44	376	5.31%
Filings	3,335	2,340	780	5.39	145	2.04%
Press	1,265	3,443	435	5.06	86	1.21%
Bloomberg	2,996	758	227	4.60	49	0.70%
PUBLIC	*50,744*	*3,314*	*16,818*	*4.87*	*3,454*	*48.73%*
C4	34,832	2,206	7,683	5.56	1,381	19.48%
Pile-CC	5,255	4,401	2,312	5.42	427	6.02%
GitHub	1,428	5,364	766	3.38	227	3.20%
Books3	19	552,398	1,064	4.97	214	3.02%
PubMed Central	294	32,181	947	4.51	210	2.96%
ArXiv	124	47,819	591	3.56	166	2.35%
OpenWebText2	1,684	3,850	648	5.07	128	1.80%
FreeLaw	349	15,381	537	4.99	108	1.52%
StackExchange	1,538	2,201	339	4.17	81	1.15%
DM Mathematics	100	8,193	82	1.92	43	0.60%
Wikipedia (en)	590	2,988	176	4.65	38	0.53%
USPTO Backgrounds	517	4,339	224	6.18	36	0.51%
PubMed Abstracts	1,527	1,333	204	5.77	35	0.50%
OpenSubtitles	38	31,055	119	4.90	24	0.34%
Gutenberg (PG-19)	3	399,351	112	4.89	23	0.32%
Ubuntu IRC	1	539,222	56	3.16	18	0.25%
EuroParl	7	65,053	45	2.93	15	0.21%
YouTubeSubtitles	17	19,831	33	2.54	13	0.19%
BookCorpus2	2	370,384	65	5.36	12	0.17%
HackerNews	82	5,009	41	4.87	8	0.12%
PhilPapers	3	74,827	23	4.21	6	0.08%
NIH ExPorter	92	2,165	20	6.65	3	0.04%
Enron Emails	24	1,882	5	3.90	1	0.02%
Wikipedia (7/1/22)	2,218	3,271	726	3.06	237	3.35%
TOTAL	*226,631*	*1,531*	*34,701*	*4.89*	*7,089*	*100.00%*

Table 1: Breakdown of the full training set used to train BloombergGPT. The statistics provided are the average number of characters per document ("C/D"), the average number of characters per token ("C/T"), and the percentage of the overall tokens ("T%"). Units for each column are denoted in the header.

2.1 Financial Datasets (363B tokens – 51.27% of training)

The Bloomberg Terminal has provided access to a comprehensive set of diverse structured and unstructured financial data and analytics for the past four decades. In serving this mission, Bloomberg analysts have curated a set of financial documents that were either created internally or acquired from external sources. We utilize this extensive collection of curated and maintained documents to create FinPile, which consists of company filings, financial news, and other data relevant to the financial markets.

Some documents included in the FinPile, such as company filings, are available to the general public, although collecting these documents and pre-processing them for LLM training is a non-trivial task. Other documents, such as (a subset of) Bloomberg news, must be purchased. The rest of the documents are private and available, among other sources, through the Bloomberg Terminal. Finally, we clean this data to strip off markup, special formatting, and templates.

Note that each document in FinPile is time-stamped, with dates ranging from 2007-03-01 to 2022-07-31; the quality and quantity of documents increase over this time range. While we do not utilize date information in this work, we plan to use it in the future, such as for evaluation of what the model learns about different time periods. While we cannot release FinPile, our experience training on a large, carefully curated, and clean domain-specific dataset may provide helpful insights to the community on the advantages and challenges of building a financial LLM in particular, and a domain-specific model in general. We provide a breakdown and analysis of FinPile in Table 2 and a brief description of the types of data included below.

2.1.1 Web (298B tokens – 42.01% of training)

Bloomberg collects web content by identifying sites that contain financially relevant information. While this category makes up the majority of FinPile, its classifications are rough, with content classified mainly by the location of the web domain. Within these location-specific sources, e.g. "US" (15.95% of total), "Asia-Pac" (4.72% of total), and "UK" (1.98% of total), document types are highly varied as would be expected from a web crawl. While web sources are common in existing public LLM training datasets, Bloomberg's web crawl is focused on high-quality websites that have financially relevant information, as opposed to a general-purpose crawl of the web.

2.1.2 News (38B tokens – 5.31% of training)

The News category includes all news sources excluding news articles written by Bloomberg journalists. Overall, there are hundreds of English news sources in FinPile including "Bloomberg Transcripts" (0.41% of total), which are transcripts of Bloomberg TV news. Generally, the content in this dataset comes from reputable sources of news that are relevant to the financial community so as to maintain factuality and reduce bias.

2.1.3 Filings (14B tokens – 2.04% of training)

Company Filings are financial statements prepared by (public) companies and made available to the general public. In some countries, like the US, public companies are mandated

Date	Bloomberg	Filings	News	Press	Web	*Total*
2007 [03-]	276	73	892	523	2,667	*4,431*
2008	351	91	1,621	628	9,003	*11,695*
2009	293	93	1,791	528	9,179	*11,883*
2010	292	111	1,917	527	11,388	*14,236*
2011	335	117	2,264	548	13,643	*16,907*
2012	403	105	2,502	529	15,015	*18,554*
2013	415	87	2,437	441	17,230	*20,610*
2014	396	251	2,458	437	18,510	*22,052*
2015	358	1,639	2,371	427	20,782	*25,576*
2016	324	1,891	2,509	418	24,337	*29,478*
2017	294	2,294	2,567	398	25,283	*30,837*
2018	275	1,791	2,702	420	26,027	*31,214*
2019	263	1,662	3,102	504	27,195	*32,726*
2020	277	1,632	2,794	805	30,928	*36,435*
2021	247	1,767	3,515	938	29,749	*36,215*
2022 [-07]	140	882	2,206	531	16,872	*20,631*
	4,939	*14,486*	*37,647*	*8,602*	*297,807*	*363,482*

Table 2: The number of tokens (in millions) contained within documents in FinPile, organized by year (rows) and type (column). Units are millions of tokens.

to prepare and submit their financial statements on a regular cadence; e.g., 10-K annual reports and 10-Q quarterly reports. In our dataset, a majority of the filings come from EDGAR, which is the SEC's online database (1.90% of total). Filings are typically long PDF documents with tables and charts that are dense in financial information, which are processed and normalized in Bloomberg. Filings are substantially different from the types of documents typically used to train LLMs, but contain critically important information for financial decision-making.

2.1.4 Press (9B tokens – 1.21% of training)

The Press category contains press releases typically issued by companies that are financially relevant. Taken together with filings, press releases represent most of the public communications of a company. However, unlike filings, press releases are similar to news stories in terms of content and style.

2.1.5 Bloomberg (5B tokens – 0.70% of training)

This category comprises Bloomberg authored news and other documents such as opinions and analyses. The largest sources are "Bloomberg News" (0.44% of total) and "Bloomberg First Word" (0.13% of total), the Bloomberg-authored wire of real-time news. While Bloomberg News covers a wide range of topics, it typically focuses on content relevant to the financial community. This dataset contains documents of varying lengths.

2.2 Public Datasets (345B tokens – 48.73% of training)

We use three widely known and available public datasets in our training corpus.

2.2.1 The Pile (184B tokens – 25.9% of training)

The Pile (Gao et al., 2021) is the dataset used in GPT-Neo (Black et al., 2021), GPT-J (Wang and Komatsuzaki, 2021), and GPT-NeoX (20B) (Black et al., 2022). We include The Pile in our training data for the following reasons. First, it has been used to successfully train an LLM. Second, it has undergone significant data cleaning and pre-processing. Third, it includes multiple domains and we believe such diverse data will aid generalization to new domains and may even support training on financial data. For example, domains such as FreeLaw and GitHub are useful to teams at Bloomberg that work on legal documents and software development, respectively. Creators of The Pile have deliberately chosen to include duplicate content, with the duplication factor being proportional to the perceived quality of the content. However, as we deduplicate each of our datasets, the size of The Pile is significantly reduced. Additionally, note that our tokenizer (§2.3) is trained on The Pile.

2.2.2 C4 (138B tokens – 19.48% of training)

The Colossal Clean Crawled Corpus (C4) is a common dataset used to train LLMs, and was introduced to support training T5 (Raffel et al., 2020). Although it overlaps with Pile-CC, C4 is cleaned and processed differently; hence, we feel that including C4 in addition to The Pile can add value more than duplicated documents would. We find that C4 contains high-quality natural language documents due to the layers of cleaning, though others have noted that the distribution across web domains is unusual, with a high fraction of data stemming from patents (Dodge et al., 2021).

2.2.3 Wikipedia (24B tokens – 3.35% of training)

Both The Pile and C4 include out-of-date copies of Wikipedia, so it could be beneficial for the factuality of the model to have up-to-date Wikipedia pages included. Therefore, we include a dump of English Wikipedia from July 1, 2022. This dataset is tokenized quite inefficiently (3.06 characters per token), indicating an above-average amount of markup, which suggests that further cleaning might benefit future model training.

2.3 Tokenization

We choose the Unigram tokenizer (Kudo, 2018) instead of a greedy merge-based sub-word tokenizer, such as Byte Pair Encoding (BPE) (Sennrich et al., 2016) or Wordpiece (Schuster and Nakajima, 2012; Wu et al., 2016), based on promising results in Kudo and Richardson (2018) and Bostrom and Durrett (2020). Following GPT-2 (Radford et al., 2019), we treat our data as a sequence of bytes rather than Unicode characters, and we include each of the 256 bytes as tokens. In a pretokenization step, the input byte sequence is broken into chunks by greedily matching the following regular expression: `[ A-Za-z]+|[0-9]|[^A-Za-z0-9]+`. This follows GPT-2 in preventing multiple character classes from appearing in a single token. However, we include spaces in the alphabetic chunks, which allows multi-word tokens to be learned, increasing information density and reducing context lengths. The pretokenization

	BLOOM	**/ours**	**NeoX**	**/ours**	**OPT**	**/ours**	**BloombergGPT**
FinPile (old)	451	110%	460	112%	456	111%	412
C4	166	121%	170	123%	170	123%	138
The Pile	203	110%	214	116%	239	130%	184
Wikipedia	21	88%	23	99%	24	103%	24
Total	*390*	*113%*	*408*	*118%*	*434*	*126%*	*345*

Table 3: Number of tokens in each training dataset with BLOOM, NeoX, OPT (GPT2), and BloombergGPT tokenizers. All token counts are in billions (B). Note that an older version of FinPile was used for this count, so token numbers will not match earlier tables.

follows the approach of PaLM (Chowdhery et al., 2022) in placing each digit in its own chunk, with the hope that this will lead to better handling of numbers. We train our tokenizer on The Pile (Gao et al., 2021) as it draws from diverse domains, including code and academic papers, in proportions that suit our use case.

Parallel Tokenizer Training. The Unigram tokenizer implementation is too inefficient to process the entire Pile dataset at once, so we use a split and merge approach. We split each of the 22 domains in the Pile into 256 chunks of roughly equal size. We then train a Unigram tokenizer with a vocabulary size of 65,536 (2^{16}) on each of the 22×256 (total $= 5,632$) chunks. We hierarchically merge the individual tokenizers by first merging the 256 tokenizers from each domain, and then combining the 22 resulting tokenizers to get the final tokenizer.

Unigram tokenizers amount to probability distributions over tokens (i.e. unigram language models), and we merge tokenizers by taking a weighted average of the probabilities of corresponding tokens, with the weights determined by the relative sizes (in bytes) of the data used to train the tokenizers. The result is a tokenizer with 7 million tokens. To reduce the size of the vocabulary to 2^{17} tokens, we drop the tokens with the smallest probabilities and renormalize. To ensure we do not need an out-of-vocabulary token, we also add as tokens the 36 (of 256 possible) bytes that do not occur in The Pile, along with an `<|endoftext|>` token.

There are various considerations in choosing the vocabulary size. One advantage of a large vocabulary for LLMs is that more information can fit into the context window. On the other hand, there is overhead with a larger vocabulary: a larger proportion of model parameters are required for token embedding. We select our vocabulary size of 2^{17} tokens based on experiments with vocabulary ranging from 25,000 to 550,000. For each vocabulary size, we tokenize the C4 dataset and compute the total size (in bytes) for the dataset, where each token is represented using $\log_2$(vocabulary size) bits. Our heuristic is to choose the vocabulary size that leads to the smallest encoded representation of C4. This gives us a vocabulary size of 125,000, which we then round up to the nearest power of 2 (2^{17}, or 131,072 tokens). Our tokenizer is large, relative to the standard vocabulary size of approximately 50,000 tokens. For an analysis of tokenization efficiency, see Table 3.

Shape	
Number of Layers	70
Number of Heads	40
Vocabulary Size	131,072
Hidden Dimension	7,680
Total Parameters	50.6B
Hyperparameters	
Max Learning Rate	6e-5
Final Learning Rate	6e-6
Learning Rate schedule	cosine decay
Gradient Clipping	0.3
Training	
Tokens	569B
Hardware	64×8 A100 40GB
Throughput	32.5 sec/step
avg. TFLOPs	102
total FLOPS	2.36e23

Table 4: A summary of the hyper-parameters and their values for BloombergGPT.

3 Model

3.1 Architecture

Our model is a decoder-only causal language model based on BLOOM (Scao et al., 2022). We present an overview of the architecture, with full details in Appendix A.

The model contains 70 layers of transformer decoder blocks defined as follows:

$$\bar{h}_\ell = h_{\ell-1} + \mathrm{SA}(\mathrm{LN}(h_{\ell-1}))$$
$$h_\ell = \bar{h}_\ell + \mathrm{FFN}(\mathrm{LN}(\bar{h}_\ell))$$

where SA is multi-head self-attention, LN is layer-normalization, and FFN is a feed-forward network with 1-hidden layer. Inside FFN, the non-linear function is GELU (Hendrycks and Gimpel, 2016). ALiBi positional encoding is applied through additive biases at the self-attention component of the transformer network (Le Scao et al., 2022). The input token embeddings are tied to the linear mapping before the final softmax. Following Le Scao et al. (2022) and first used in Dettmers et al. (2022), the model has an additional layer normalization after token embeddings, formally:

$$\bar{h}_1 = \mathrm{LN}^{em}(h_0) + \mathrm{SA}(\mathrm{LN}(\mathrm{LN}^{em}(h_0))),$$

where h_0 is the initial token embedding and LN^{em} is the new component of *em*bedding layer-normalization. Notice that the second term includes two consecutive layer-normalizations.

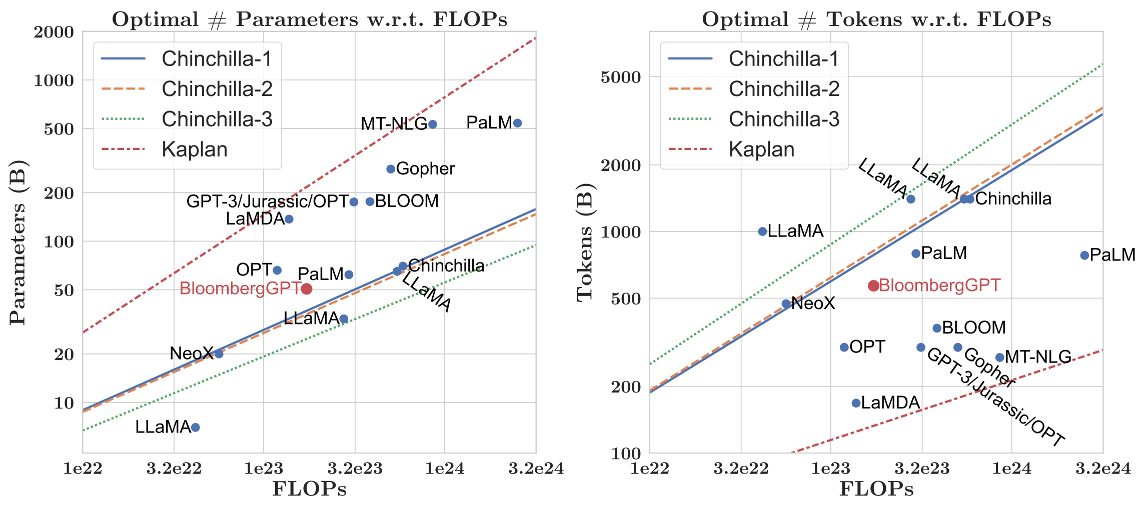

Figure 1: Kaplan et al. (2020) and Chinchilla scaling laws with prior large language model and BloombergGPT parameter and data sizes. We adopt the style from Hoffmann et al. (2022).

3.2 Model Scaling

Size. The size of our model is based on Chinchilla scaling laws (Hoffmann et al., 2022), in particular their Approach 1 and Approach 2. We start with a total compute budget of 1.3M GPU hours on 40GB A100 GPUs. Since we adopt activation checkpointing to reduce our memory footprint, this costs us an additional 0.33x TFLOPs per iteration due to repeated forward passes. To account for this additional cost, we plug in 0.75 × 1.3M into Chinchilla equations instead of the full amount.

From Hoffmann et al. (2022), we use the data reported in Table 3 for Approach 1 and Table A3 for Approach 2, and fit regression lines to their log-scaled versions. This gives us:

$$\begin{aligned} \text{Approach 1} \qquad Parameters &= \exp_{10}(\log_{10}(FLOPs) \cdot 0.498 - 1.004) = 52.993\text{B} \\ Tokens &= \exp_{10}(\log_{10}(FLOPs) \cdot 0.502 + 0.229) = 1111.112\text{B} \\ \text{Approach 2} \qquad Parameters &= \exp_{10}(\log_{10}(FLOPs) \cdot 0.490 - 0.839) = 49.753\text{B} \\ Tokens &= \exp_{10}(\log_{10}(FLOPs) \cdot 0.510 + 0.062) = 1175.766\text{B} \end{aligned}$$

These calculations imply that our dataset of ~700B tokens is too small for a "Chinchilla optimal" configuration given our compute budget (assuming just one pass through the data).[1] While we can increase the amount of general-purpose training data, we are limited in the amount of domain-specific training data at our disposal. FinPile is already among the largest domain-specific training sets, and we do not want it to represent less than half of our total training.

1. The scaling law derived by Chinchilla is tokenizer-specific. Our tokenizer can encode the same document more compactly due to the support of multi-word expressions and the larger vocabulary size. It's still an open question how well these scaling laws transfer across tokenizers, and how vocabulary size impacts token and parameter trade-offs assuming fixed compute. We leave this exploration to future work.

Since we are data limited, we choose the largest model that we can, while ensuring that we can train on all our tokens and still leave ~30% of the total compute budget as a buffer for unforeseen failures, retries, and restarts. This leads us to a 50B parameter model, which is also roughly the Chinchilla optimal size for our compute budget. Figure 1 provides a summary of the scaling laws and how BloombergGPT compares to other models.

Shape. To determine how to allocate the 50B parameters to different model components (i.e., the "shape" of our model), we follow Levine et al. (2020), who propose that for a total number of self-attention layers L, the optimal hidden dimension D is obtained by:

$$D = \exp(5.039)\exp(0.0555 \cdot L)$$

We sweep L over a range of integer values and pick the (L, D) combination that yields a total of ~50B parameters. This leads to the choice of $L = 70$ and $D = 7510$ as our target shape parameters. However, we also want to follow the tradition that the hidden dimension is evenly divisible by the number of attention heads, with the quotient giving the attention head dimension. Furthermore, we want the dimensions to be multiples of 8 to achieve higher performance in Tensor Core operations (NVIDIA, 2023). We settle on 40 heads, each having a dimension of 192, resulting in a total hidden dimension of $D = 7680$ and a total of 50.6B parameters. Table 4 provides a summary of the hyper-parameters used in BloombergGPT.

3.3 Training Configuration

Training. BloombergGPT is a PyTorch model trained with a standard left-to-right causal language modeling objective. Following Brown et al. (2020), we want all our training sequences to be exactly the same length, in our case 2,048 tokens, to maximize GPU utilization. To achieve this, we concatenate all our tokenized training documents with an `<|endoftext|>` token as a document separator. We then break this token sequence into chunks of 2,048 tokens. Note that with this approach, each training sequence may contain multiple documents from different domains. Also note that, because we're using ALiBi positional encoding, BloombergGPT can be applied to sequences longer than 2,048 at inference time. For optimization efficiency, training sequences are grouped together into batches, as described in more detail below.

Optimization. We use the AdamW optimizer (Loshchilov and Hutter, 2019). We set β_1 to 0.9, β_2 to 0.95, and weight decay to 0.1. Following Brown et al. (2020), we set the maximum learning rate to 6e-5 and use the cosine decay learning rate scheduler with linear warmup. We warm up the learning rate in the first 1800 steps. Following Hoffmann et al. (2022), the final learning rate is 0.1x the max learning rate, i.e. 6e-6. We also employ batch size warmup (Brown et al., 2020): in the first 7,200 steps, we use a batch size of 1,024 (2.1M tokens), then switch to a batch size of 2,048 (4.2M tokens) for the remainder of training.

We set dropout to 0.0 in all layers in our initial run, although we add dropout later as explained in §4. The model parameters are randomly initialized to samples from a normal distribution with zero mean and standard deviation $\sqrt{1/(3D)} = 0.006588$ (Smith et al., 2022). Following Megatron-LM (Shoeybi et al., 2019), we rescale the standard deviation of the second layer in the MLP and the output layer of the attention by $1/\sqrt{2L}$. We use the technique of `query_key_layer_scaling` (Shoeybi et al., 2019), which was proposed to improve numerical stability for FP16 mixed-precision training but may also help in BF16.

Training Instability. LLMs optimization requires running convex optimization algorithms over incredibly complex non-convex loss surfaces. Previous work has reported various instabilities while training LLMs. For example, Chowdhery et al. (2022) found that the loss spiked roughly 20 times while training PaLM, despite the fact that gradient clipping was enabled. They mitigated these issues by re-starting training from a checkpoint roughly 100 steps before the spike started, and then skip 200–500 data batches. They hypothesized that spikes occur due to the combination of specific data batches with a particular model parameter state. Similarly, during OPT training, Zhang et al. (2022a) noticed spikes in the gradient and activation norms, or divergences in the training perplexity. After these behaviors, they lowered their learning rate, which stabilized these norms and allowed training to continue. Interestingly, Scao et al. (2022) report only a single loss spike, from which the model recovered on its own.

Hardware Stack. We use the Amazon SageMaker service provided by AWS to train and evaluate BloombergGPT. We use the latest version available at the time of training and train on a total of 64 p4d.24xlarge instances. Each p4d.24xlarge instance has 8 NVIDIA 40GB A100 GPUs with NVIDIA NVSwitch intra-node connections (600 GB/s) and NVIDIA GPUDirect using AWS Elastic Fabric Adapter (EFA) inter-node connections (400 Gb/s). This yields a total of 512 40GB A100 GPUs. For quick data access, we use Amazon FSX for Lustre, which supports up to 1000 MB/s read and write throughput per TiB storage unit.

3.4 Large-scale Optimization

To train BloombergGPT, which has a larger memory footprint than available GPU memory on cloud instances, we rely on stage 3 of ZeRO optimization (Rajbhandari et al., 2020). We utilize the proprietary SageMaker Model Parallelism (SMP) library from AWS, which enables the automatic distribution of large models across multiple GPU devices and instances (Karakus et al., 2021). After experimenting with various techniques, we achieve 102 TFLOPs on average and each training step takes 32.5 seconds. We find the following setup to be the best performing in our training.

ZeRO Optimization (stage 3). ZeRO shards the training state (model parameters, gradients, and optimizer state) across a group of GPUs. We shard a model across 128 GPUs, and we have 4 copies of the model during training.

MiCS. Zhang et al. (2022b) decrease training communication overhead and memory requirements for cloud training clusters. MiCS includes such features as hierarchical communication, 2-hop gradient update, scale-aware model partitioning.

Activation Checkpointing. Chen et al. (2016) minimizes training memory consumption by removing activations at the expense of additional computation during backward passes. When a layer has activation checkpointing enabled, only the layer input and outputs are kept in memory following a forward pass, while any intermediate tensors are discarded from memory. During the backward pass, these intermediate tensors may be recomputed. We apply activation checkpointing to each transformer layer.

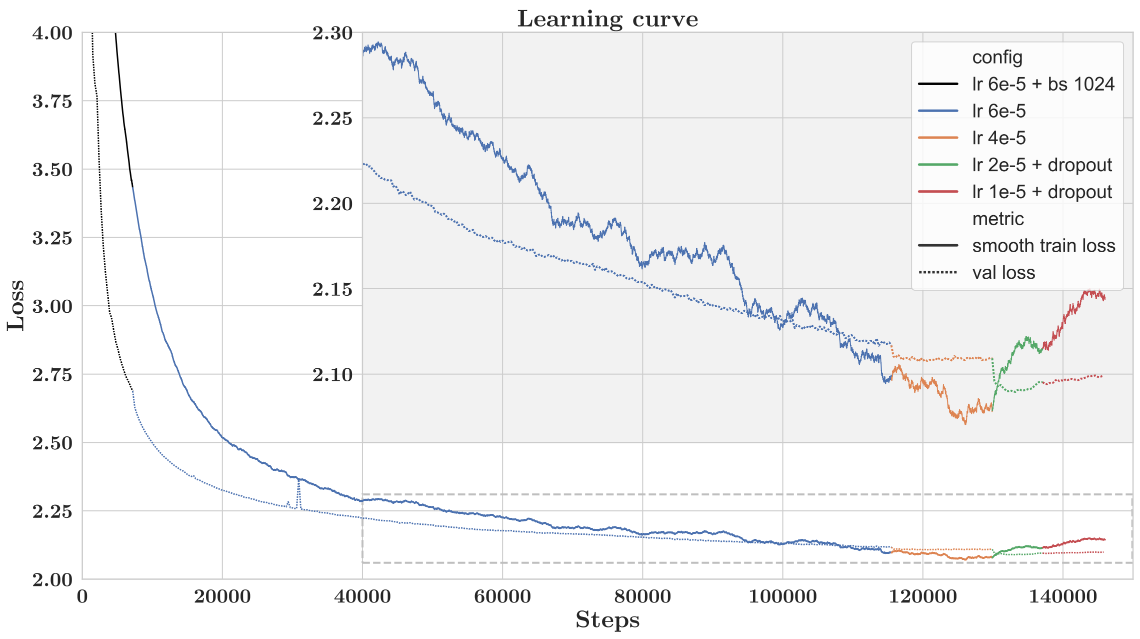

Figure 2: (Smoothed) training and validation losses for BloombergGPT. Inner plot is a zoomed-in version of the area within dashed rectangle in the outer plot (with shared x-axis). Colors denote different hyperparameter configurations. Styles denote training vs validation loss.

Mixed Precision Training. To reduce the memory requirements, forward and backward passes are done in BF16, while parameters are stored and updated in full precision (FP32). The ALiBi matrices are computed in full precision and stored in BF16. We also use FP32 to calculate fused softmax in the Attention block and store its results in BF16. Finally, the softmax calculations in the loss function are computed in FP32.

Fused Kernels. Another possibility for optimization is combining composition of several operations into a single GPU operation. This can both reduce peak memory usage by avoiding storage of intermediate results in the computation graph, as well as help improve speed. Similar to Megatron-LM (Shoeybi et al., 2019), we use a masked-causal-softmax fused kernel in SMP in the self-attention module. In practice, we observe 4-5 TFLOPs improvement for speed, and avoid out-of-memory errors given the rest of the configuration.

4 Training Run

The process of training BloombergGPT involved decisions along the way based on the progress of model training. We share some highlights of this process. A detailed presentation appears in the Training Chronicles (Appendix C). Figure 2 shows the learning curves for both training and validation sets. The solid lines show (smoothed) training loss and the dotted lines show loss on the held-out validation set. Changes in the color of the lines indicate changes to the optimization hyperparameter configurations, either as scheduled, or in response to increasing or stagnating validation loss. This plot shows the path taken by the successful model training run. To present a clear plot, the Figure does not show other attempts with different model configurations, overwritten partial runs after a rollback, or other training strategies not utilized in the final model.

We measured training loss every five steps on the current batch. The raw values vary wildly, causing large jitter when plotted. The plot smoothes the training loss by showing a running average $y_t = \frac{\sum_{i=0}^{t} x_i \cdot (1-\alpha)^{(t-i)}}{\sum_{i=0}^{t} (1-\alpha)^{(t-i)}}$ where $\alpha = 0.001$. Smoothing is not needed for the validation loss since it is measured on the entire validation set every 300 steps.

We trained for a total of 139,200 steps (~53 days) and ended model training after completing ~80% of one epoch through our training data (569B tokens out of the 709B tokens available). We ended training early because the loss on our held-out development set was no longer improving, although it's possible that substantially longer training may have yielded further improvements.

We began the run with a warm-up batch size of 1,024 for 7,200 steps, after which we switched to the regular batch size of 2,048 (color changes from black to blue). Change in batch size manifests as a visible curvature change in the validation loss at step 7,200. Most of the remainder of the training performed stably with decreasing training and validation losses. Intervention was required at later stages, after step 115,500, when we observed flat or increasing validation loss. We then applied the following corrective modifications in sequence:

- Step 115,500 (blue to orange): Shrink learning rate to two-thirds
- Step 129,900 (orange to green): Halve learning rate, and add dropout (with 0.1 probability)
- Step 137,100 (green to red): Halve learning rate again

We ended the run at step 146,000 based on the lack of observable progress on the validation loss. We selected the checkpoint at step 139,200 as the final model based on validation loss and downstream evaluations.

5 Evaluation

We evaluated the performance of BloombergGPT on two broad categories of tasks: finance-specific and general purpose. The finance-specific tasks help us test our hypothesis that training on high-quality finance-specific data will yield better results on financial tasks. The general purpose tasks investigate whether the performance of our model is directly comparable to previously published results. For financial tasks, we assembled publicly available financial datasets that include a range of NLP tasks. Then, to directly test BloombergGPT's ability on Bloomberg tasks of interest, we also included tasks drawn from Bloomberg-internal high-quality evaluation sets for sentiment analysis and named entity recognition. For general-purpose tasks, we draw from multiple existing benchmarks and group results into the following categories: BIG-bench Hard, Knowledge Assessments, Reading Comprehension, and Linguistic Tasks. The number of tasks per type and the definitions of the groups are presented in Table 5.

Suite	Tasks	What does it measure?
Public Financial Tasks	5	Public datasets in the financial domain
Bloomberg Financial Tasks	12	NER and sentiment analysis tasks
Big-bench Hard (Suzgun et al., 2022)	23	Reasoning and general NLP tasks
Knowledge Assessments	5	Testing closed-book information recall
Reading Comprehension	5	Testing open-book tasks
Linguistic Tasks	9	Not directly user-facing NLP tasks

Table 5: Evaluation Benchmarks. We evaluate BloombergGPT on a high-coverage set of standard benchmarks that assess downstream performance, taken from HELM, SuperGLUE, MMLU, and the GPT-3 suite. Since these have significant overlap and/or include each other, we restructure them into the categories presented here. We only evaluate on one setup per dataset. We further assess BloombergGPT on a suite of internal and public financial tasks.

Name	# Tokens (B)	# Params. (B)	Compute
BloombergGPT	569	50.6	1.00×
GPT-NeoX	472	20	0.33×
OPT	300	66	0.69×
BLOOM	366	176	2.24×
GPT-3	300	175	1.82×

Table 6: Evaluation model cohort. OPT and BLOOM each have multiple sizes available and we report those we evaluated. We note that compute numbers are only partially comparable between models: For example, BLOOMs training data is only 1/3 English, and OPT repeated some of its training data. We report GPT-3 results whenever available but did not run it ourselves due to lack of availability.

We compare BloombergGPT to the three closest models described in §7 based on model size, type of training data, overall performance, and most importantly, access. An overview of the model sizes and compute is provided in Table 6.

1. GPT-NeoX (Black et al., 2022): According to Liang et al. (2022), this model is the best performing available model under 50B parameters.

2. OPT_{66B} (Zhang et al., 2022a): We chose to compare to OPT_{66B} since our model size and structure roughly match, though our model is smaller.

3. $\text{BLOOM}_{\text{176B}}$ (Scao et al., 2022): While this model is substantially larger than BloombergGPT, we use the same model architecture and software stack. We note that $\text{BLOOM}_{\text{176B}}$ is multilingual, so while it is much larger, it also is trained on data from more languages.

All three models use some of the same general-purpose datasets we use in our training corpus. We additionally report results from the original GPT-3 (Brown et al., 2020) whenever externally available.[2]

We prefer running models ourselves to ensure identical evaluation setups, and we place any results that have been reported elsewhere and were not run by us into a separated group. To fairly compare the models, we avoid any tuning of prompts and other techniques that could lead to improved results for some, but not all, models. For that reason, every task is tested via "standard" prompting (shown in Table 7), i.e., without any parameter changes to the underlying model, without task descriptions, and without Chain-of-Thought prompting (Wei et al., 2022b). The number of few-shot examples presented to the model depends on the task, and we include these details in the respective sections. For each group of results, we further present a win rate similar to Liang et al. (2022) that represents the fraction of "wins" in side-by-side comparisons over individual tasks between all model pairs for which we have run the evaluation ourselves.

5.1 Few-shot Methodology

For tasks where a set of candidates are given, we perform likelihood-based classification, following Brown et al. (2020). We consider three methods for classification: regular, calibration, and normalization. Formally,

- Regular: $\arg\max_\alpha p(\alpha|\mathbf{s})$
- Calibration: $\arg\max_\alpha p(\alpha|\mathbf{s})/p(\alpha|\text{"Answer:"})$
- Normalization: $\arg\max_\alpha p(\alpha|\mathbf{s})/\text{len}(\alpha)$

where α is a candidate, $\mathbf{s}$ is the context, and len measures the number of sub-word tokens. We report the performance of the best method for each model and task. For other tasks, we perform generation via greedy decoding.

We use the official split and report performance on the test set whenever possible. If the test labels are not publicly available, we report performance on the dev set instead. If an official split for a dataset does not exist, we create train and test splits by selecting 20% of examples to be the test and the rest as train. All few-shot context examples are sampled from the training set. To reduce the variance of few-shot evaluation, we sample different shots for each test example, unless otherwise specified. For the sake of consistency, for each test example, all models have identical surface form as input in our evaluation.

5.2 Heldout Loss

We begin by testing how well BloombergGPT models the language distribution of the in-distribution finance data. We evaluate the bits per byte of the different models on a heldout dataset that contains examples from all sections of FinPile (described in §2). To limit data leakage and better simulate real-world usage of LLMs, we select a temporally heldout

2. Another related general-purpose model at a comparable size (LLaMA, Touvron et al., 2023), was released during the preparation of this manuscript, but third-party evaluation results were not available and we haven't received access to the model weights.

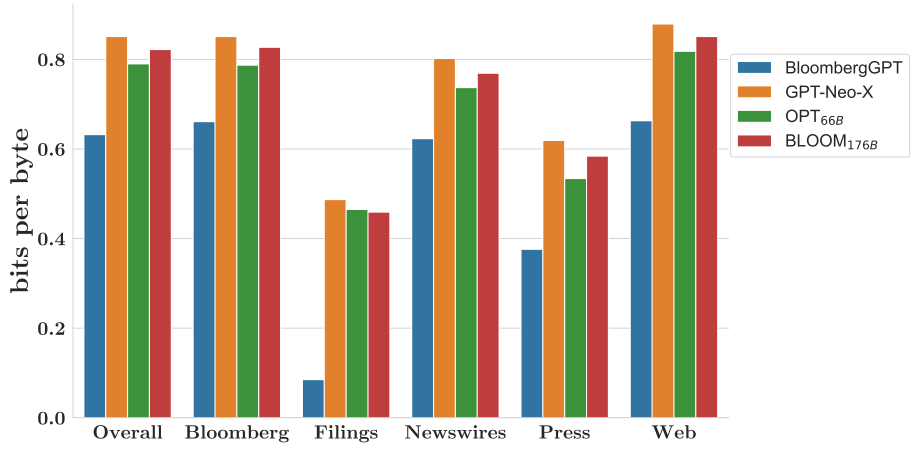

Figure 3: Bits per byte on a heldout test set of each data type in our FinPile (lower is better). The set of documents is held out in time and deduplicated with the training set, such that all of it is completely unseen by BloombergGPT. Regardless, we observe a large gap between the models. The improvement is largest for specialized in-domain documents like Filings.

dataset that is strictly further in the future than the training set, and perform deduplication between the training and heldout set. During evaluation, for documents that are longer than 2,048 tokens, we use a sliding window approach with half window size as context. That means that any token beyond the first 2,048 has at least 1,024 tokens as context during prediction. We report the loss breakdown by the type of document in FinPile.

Figure 3 shows that BloombergGPT consistently outperforms other models. While this is expected and mainly serves as a sanity check, it also provides valuable insight into the generalization capabilities of the other models. For example, the gap to BloombergGPT is most significant in the Filings category, likely because these documents, while public, are typically in PDF format and thus not included in any existing datasets.

5.3 Financial Tasks

The NLP tasks most often considered in finance are also common in the broader NLP literature; but, these tasks take on different characteristics and challenges when performed on financial data. Take the example of sentiment analysis, where a headline such as "COMPANY to cut 10,000 jobs" portrays negative sentiment in the general sense but can at times be considered positive for financial sentiment towards COMPANY, as it might result in the stock price or investor confidence increasing. We use a combination of public and internal benchmarks to assess the performance of BloombergGPT, BLOOM$_{176B}$, GPT-NeoX, and OPT$_{66B}$. All task types considered and their corresponding prompt templates are shown in Table 7.

Task	Template/Example
Discriminative	

Sentiment Analysis	`{sentence}`
	`Question: what is the sentiment?`
	`Answer: {negative/neutral/positive}`

Aspect Sentiment Analysis	`{sentence}`
	`Question: what is the sentiment on {target}?`
	`Answer: {negative/neutral/positive}`

Binary Classification	`{sentence}`
	`Question: {question}?`
	`Answer: {Yes/No}`

Generative	
NER	Steve Jobs is the CEO of Apple Extract named entity: Steve Jobs (person), Apple (organization)
NER+NED	AAPL stopped using Intel Chips Extract ticker: AAPL, INTC

QA	`{context}`
	`Question: {question}?`
	`Answer: {answer}`

Table 7: Template for the different tasks we evaluate in the financial domain.

5.3.1 External Financial Tasks

Our public financial benchmarks include four tasks from the FLUE benchmark (Shah et al., 2022) and the ConvFinQA dataset (Chen et al., 2022). As LLM performance on most of these financial tasks have not been broadly reported, there is no standard testing framework. Thus, we adapt them to a few-shot setting (see Section §5.1). Our guiding principle in designing the experiments was to select the number of shots such that the average performance across all the models was best. While non-LLM numbers of custom models for these tasks are available, we omit reporting them here due to differences in the evaluation setup. As a result, our claims are restricted to comparisons of LLMs. We evaluate on the following tasks (more details provided in Appendix B):

- **FPB** (Malo et al., 2014): The Financial Phrasebank Dataset includes a sentiment classification task on sentences from financial news. Any news that could benefit/hurt an investor is considered positive/negative and neutral otherwise. We create our own splits and report F1 score weighted by support in a 5-shot setup.

- **FiQA SA** (Maia et al., 2018): The second sentiment analysis task is to predict the aspect-specific sentiment in English financial news and microblog headlines, which were published as a part of the 2018 challenge on financial question answering and opinion mining. While the original dataset is annotated on a continuous scale, we discretize the data into a classification setup with negative, neutral, and positive classes. Like with FPB, we create our own splits including microblogs and news, and use a 5-shot setup, reporting weighted F1.

	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}
ConvFinQA	**43.41**	30.06	27.88	36.31
FiQA SA	**75.07**	50.59	51.60	53.12
FPB	**51.07**	44.64	48.67	50.25
Headline	**82.20**	73.22	79.41	76.51
NER	60.82	**60.98**	57.49	55.56
All Tasks *(avg)*	**62.51**	51.90	53.01	54.35
All Tasks *(WR)*	**0.93**	0.27	0.33	0.47

Table 8: Results on financial domain tasks.

- **Headline** (Sinha and Khandait, 2020): This is a binary classification task of whether a news headline in the gold commodity domain includes certain information. This human-annotated dataset consists of English news headlines about "gold". Each news article carries a subset of the following tags: "price or not", "price up", "price down", "price stable", "past price", "future price", "past general", "future general", "asset comparison". We verbalize each tag into a question using the official documentation, use 5 shots, and report the average weighted F1 score across all categories.

- **NER** (Salinas Alvarado et al., 2015): This is a named entity recognition task on financial data gathered for credit risk assessment from financial agreements filed with the SEC. The annotated entity types follow the standard CoNLL format (Tjong Kim Sang and De Meulder, 2003) and are annotated with PER, LOC, ORG, and MISC. As it is nontrivial to learn to predict empty outputs in few-shot setups, we drop sentences that do not contain any entity. We further drop MISC tags due to their ambiguous definition. All the models required more shots to perform well and we thus selected 20 shots and report the entity-level F1 score.

- **ConvFinQA** (Chen et al., 2022): Given input from S&P 500 earnings reports that includes text and at least one table with financial data, the task is to answer conversational questions that require numerical reasoning over the input. This task requires numerical reasoning, an understanding of structured data and financial concepts, and a model needs to relate follow-up questions to the dialog turns.

  For ConvFinQA, we use an entire gold conversation and its context is used as input to the models. As each "turn" of the conversation concludes, the "turn" along with the answer for that turn is appended as context for future turns. We report the exact match accuracy on the public development set.

BloombergGPT performs best of all models for four of the five tasks (ConvFinQA, FiQA SA, FPB, and Headline) and comes in second in NER (Table 8). Consequently, BloombergGPT also has the highest win rate among all the models that we tested. The gap to equally-sized models is especially pronounced for ConvFinQA which is challenging due to the requirement to use conversational input to reason over tables and generate an answer.

Name	Time	Tokens	Test Size	% Pos	% Neu	% Neg
Equity News	2018–2019	150-200	1,000	7	87	6
Equity Social Media	2015–2020	15-20	1,000	10	83	7
Equity Transcript	2008–2020	70-80	800	19	75	6
ES News	2016–2019	100-120	1,000	32	53	15
Country News	2009–2021	50-1,000	1,000	18	60	22

Table 9: An overview of the Bloomberg-internal sentiment analysis tasks. Input token and label distribution numbers are computed on the test set.

5.3.2 Internal Task: Sentiment Analysis

For the Bloomberg-internal tasks, we consider aspect-specific sentiment analysis, which is prevalent in financial literature. All of the datasets we use are in English.

Our annotation process consists of a discovery phase during which we establish the annotation and sampling procedures, understand how many annotators are typically required per example, and determine the level of training that is needed for the annotators (Tseng et al., 2020). Depending on the complexity of the task, our annotators are a dedicated team of financial experts at Bloomberg, consultant workers, or a combination of both. In each case, ties are resolved by adjudication from additional annotators and ambiguous examples are excluded. All the datasets in this section were annotated by 2 annotators with a third annotator breaking any ties.

We measure the performance of LLMs for the internal datasets using a five-shot evaluation, similar to the external datasets. As the datasets are large, we randomly sample at most 1k test examples. We report F1 weighted by the support of each label. Note that, similar to the external datasets, it is likely that the *unlabeled* versions of the data used in our internal datasets occur in FinPile and are therefore seen by BloombergGPT during training. However, since some of FinPile is also available on the web, other LLMs we compare against may have also been trained on unlabeled versions of this data. Dataset statistics are provided in Table 9.

- **Equity News Sentiment**: This task is to predict the aspect-specific sentiment expressed in the news story toward a company. The dataset consists of English news stories from Bloomberg, premium, and web content. Annotations of “positive”, “negative”, or “neutral” indicate that the news story is likely to increase, decrease, or not change the long-term investor confidence in the company.

- **Equity Social Media Sentiment**: The task is similar to “Equity News Sentiment” but instead of news, we use financially-relevant English social media content.

- **Equity Transcript Sentiment**: This task is also similar to “Equity News Sentiment” but instead of news, we use transcripts from company press conferences. The transcripts are made available through the use of speech recognition and at times, human edits. Long transcripts are processed in chunks, and each chunk in our dataset typically contains between 70 and 80 tokens.

	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}
Equity News	**79.63**	14.17	20.98	19.96
Equity Social Media	**72.40**	66.48	71.36	68.04
Equity Transcript	**65.06**	25.08	37.58	34.82
ES News	**46.12**	26.99	31.44	28.07
Country News	**49.14**	13.45	17.41	16.06
All Tasks *(avg)*	**62.47**	29.23	35.76	33.39
All Tasks *(WR)*	**1.00**	0.00	0.67	0.33

Table 10: Results on internal aspect-specific sentiment analysis datasets. BloombergGPT far outperforms all other models on sentiment analysis tasks.

Name	Tokens	Test Size	LOC	ORG	PER
BFW	~21	500	0.2	1.6	0.0
BN	~30	500	0.7	1.0	0.6
Filings	~32	500	0.1	1.3	0.4
Headlines	~50	500	0.7	2.7	1.0
Premium	~29	500	0.6	1.4	0.3
Transcripts	~23	500	0.6	0.6	0.3
Social Media	~12	500	0.4	1.4	0.2

Table 11: An overview of statistics of our internal NER test set. We report average number of LOCation, ORGanization, PERson per example.

- **ES News Sentiment**: While this task is to predict the aspect-specific sentiment expressed in the news story towards a company (aspect), the goal is not to indicate effect on investor confidence. The stories are annotated "positive", "negative", or "neutral" if the news story contains content that reflects good, bad, or neutral news about the company's environmental and social policies.

- **Country News Sentiment**: This task is different from the other sentiment tasks in that the goal is to predict the sentiment expressed in the news story towards a country. The dataset consists of English news stories from Bloomberg, premium, and web content. The stories are annotated "positive", "negative", or "neutral" if the news story alludes to the growth, shrinkage, or status quo of that country's economy.

Table 10 shows that across the four internal aspect-specific sentiment tasks BloombergGPT performs better than all the other tested models, by a wide margin. The only task in which the models perform similarly is the social media sentiment task, while BloombergGPT outperforms the other models by at least 25 and up to over 60 points in the other three.

5.3.3 Exploratory Task: NER

Even though NER is a well-established NLP task with state-of-the-art results using BERT (Wu and Dredze, 2019; Luoma and Pyysalo, 2020) and T5 (Liu et al., 2022) style models, NER is largely an unexplored task for generative LLMs. NER is not in HELM (Liang et al., 2022), there is a single (Polish) task in BIG-bench (Srivastava et al., 2022), and none of the LLM papers we study report NER performance. Hence, we consider NER as an exploratory task and report preliminary NER results given its importance in the Financial sector.

There are a few reasons for why NER may be a difficult task for generative LLMs. NER is an information extraction task, and a better fit for encoder-decoder or encoder-only architectures. The generative nature of LLMs does not confer an advantage for NER. We find that extensive prompt engineering and a greater number of shots are required to obtain reasonable results for NER than for other tasks. Finance-specific NER has subtleties that make it especially difficult for zero or few-shot learning.

For example, consider the (fabricated) headline "Bloomberg: Mr. Musk adds new features to Twitter and comments on China". Depending on our annotation guidelines and downstream task needs: (a) the reporting news organization "Bloomberg" can be tagged or not, depending on whether we want only salient entities, (b) "Mr. Musk" or just "Musk" is the PER to be tagged, (c) "Twitter" can be tagged as an ORG or a PRD (product) as features are added to the Twitter product and not the organization, and (d) "China" can be tagged ORG or LOC, though the right tag is likely ORG. Without adding extensive annotation guidelines in the prompt, the LLM does not know the intended tagging behavior.

Based on preliminary testing, we determined the following setting to obtain the best performance on the internal NER tasks from all models. First, we restrict the entity types to be predicted to be ORG, PER, and LOC. In all, we filtered out less than 1% of entities. We also remove all documents that contain no entities (i.e., all "O"'s). Both of these modifications are intended to increase the usefulness of the examples seen in few-shot prompting. We expect that further work on prompt engineering for NER could produce better results.

We consider seven Bloomberg internal NER datasets from different domains.

- **BN NER**: This is a named entity recognition task on entities occurring in English long-form Bloomberg news content (the "BN wire") between 2017 to 2020.

- **BFW NER**: Similar to "BN NER" but instead of using the long-form BN wire, we use short-form stories from the "Bloomberg First Word" wire between 2018 to 2020.

- **Filings NER**: The goal of this task is to identify entities that occur in mandatory financial disclosures filed by companies. The dataset contains filings sampled between 2016 and 2019.

- **Headlines NER**: The goal of this task is to identify entities that occur in headlines of English Bloomberg news content. The dataset contains headlines sampled between 2016 and 2020.

- **Premium NER**: The goal of this task is to identify entities that occur in a subset of the third-party English news content ingested by Bloomberg. The dataset contains stories sampled between 2019 and 2021.

- **Transcripts NER**: The goal of this task is to identify entities that occur in transcripts of company press conferences. The dataset contains transcripts from 2019.

	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}
NER				
BFW	72.04	71.66	72.53	**76.87**
BN	57.31	52.83	46.87	**59.61**
Filings	58.84	59.26	59.01	**64.88**
Headlines	**53.61**	47.70	46.21	52.17
Premium	60.49	59.39	57.56	**61.61**
Transcripts	75.50	70.62	72.53	**77.80**
Social Media	60.60	56.80	51.93	**60.88**
All Tasks *(avg)*	62.63	59.75	58.09	**64.83**
All Tasks *(WR)*	0.57	0.29	0.19	**0.95**
NER+NED				
BFW	**55.29**	34.92	36.73	39.36
BN	**60.09**	44.71	54.60	49.85
Filings	**66.67**	31.70	65.63	42.93
Headlines	**67.17**	36.46	56.46	42.93
Premium	**64.11**	40.84	57.06	42.11
Transcripts	**73.15**	23.65	70.44	34.87
Social Media	67.34	62.57	**70.57**	65.94
All Tasks *(avg)*	**64.83**	39.26	58.79	45.43
All Tasks *(WR)*	**0.95**	0.00	0.67	0.38

Table 12: Results on internal NER and NED datasets. On NER, while the much larger BLOOM_{176B} model outperforms all other models, results from all models are relatively close, with BloombergGPT outperforming the other two models. On NER+NED, BloombergGPT outperforms all other models by a large margin.

- **Social Media NER**: The goal of this task is to identify entities that occur in English financially-relevant social media content. The dataset contains social media content sampled between 2009 and 2020.

As our datasets are substantive, we randomly sample 4,000 training and 500 testing examples from each filtered internal dataset. We utilize 20-shot prompts and evaluate using F1. The results from the internal NER tasks are mixed (Table 12). The much larger BLOOM_{176B} wins most of the NER tasks. Of the like-sized models, BloombergGPT performs the best placing first once (Headlines), second four times (BN, Premium, Transcripts, Social media), third once (BFW), and last once (Filings).

Exploratory Task: NER+NED Named entity disambiguation (NED) links entity mentions to known entities in knowledge bases or other structured information sources. Within the financial world, we seek to link text mentions of companies to their ticker symbols, an abbreviation that uniquely identifies publicly traded shares of a particular stock on a particular stock market.

We directly test the ability of an LLM to complete this task by evaluating a joint NER+NED task: identify the stock tickers of companies mentioned in a document. This requires the model to first identify company mentions and then generate the corresponding stock ticker. For example, given "AAPL announced that they will stop using Intel chips in future products." the correct NER output would be "AAPL, Intel" while the correct NER+NED output would be "AAPL, INTC".

One of the advantages of this task is that it is robust to variations in extracting the exact text span. While NER evaluation requires exact matches, tickers may be successfully produced without first identifying spans. Furthermore, it evaluates a model's knowledge of companies, their various surface forms, and company to ticker mappings.

We create evaluation data with linked tickers for this task by running a state-of-the-art entity linking system for companies in financial data over the Bloomberg internal NER annotated documents from each domain. We remove documents with no linked tickers. Following our NER evaluations, we randomly sample 4,000 training and 500 testing examples from each filtered internal dataset. We utilize 20-shot prompts and evaluate using F1.

Table 12 shows that BloombergGPT outperforms all other models by a large margin, except on social media data where it comes in second behind BLOOM_{176B}. In our social media data, companies are often referenced by their tickers, removing the requirement of the model to link the mention and reverting the task to NER. These results further underscore the advantage of BloombergGPT for financial tasks.

5.4 BIG-bench Hard

We now turn to evaluate BloombergGPT on standard, general-purpose NLP tasks. While the focus of our model is on financial tasks, our inclusion of general-purpose training data may help improve not only the financial tasks, but also allow our model to perform well on more standard NLP datasets. We start with BIG-bench Hard (Suzgun et al., 2022), a subset of the most challenging tasks in BIG-bench (Srivastava et al., 2022). It only includes tasks in which the best available model at construction was unable to achieve a performance higher than the average human rater via standard prompting techniques.

Results for each task are shown in Table 13. Overall, while BloombergGPT falls behind the much larger PaLM_{540B} (10x parameters) and BLOOM_{176B} (3.5x parameters), it is the best-performing among similarly sized models. In fact, its performance is closer to BLOOM_{176B} than it is to either GPT-NeoX or OPT_{66B}. It further achieves the best performance of all models in date understanding, hyperbaton (ordering of adjectives), and tracking shuffled objects. In sum, according to this benchmark, we find that developing finance-specific BloombergGPT did not come at the expense of its general-purpose abilities.

5.5 Knowledge Assessments

We next assess knowledge, which we define as the ability to recall information seen during model training, via scenarios that have the model answer questions without providing additional context or resources (closed-book question answering). This includes multiple-choice questions, and we report accuracy. We follow the template of Brown et al. (2020). The list of scenarios is as follows:

- **ARC** (Clark et al., 2018): Multiple-choice questions collected from 3rd to 9th grade science exams, includes easy and challenging splits.

BIG-bench Hard Task	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}	PaLM_{540B}
Boolean Expressions$^{\lambda}$	62.40	**71.20**	48.40	69.20	**83.2**
Causal Judgement	49.73	**52.41**	51.87	51.87	**61.0**
Date Understanding	**54.80**	45.60	49.60	50.00	53.6
Disambiguation QA	34.00	**40.80**	40.40	40.40	**60.8**
Dyck Languages$^{\lambda}$	15.60	26.00	14.80	**42.00**	28.4
Formal Fallacies	50.80	52.80	**54.00**	52.80	53.6
Geometric Shapes$^{\lambda}$	15.20	8.00	11.60	**22.40**	**37.6**
Hyperbaton	**92.00**	**92.00**	91.60	**92.00**	70.8
Logical Deduction$^{\lambda}$ *(avg)*	**34.53**	30.93	31.87	34.00	**60.4**
Movie Recommendation	90.40	86.40	**91.20**	**91.20**	87.2
Multi-Step Arithmetic$^{\lambda}$ [Two]	**1.20**	0.40	0.40	0.00	**1.6**
Navigate$^{\lambda}$	42.00	45.20	42.00	**50.00**	**62.4**
Object Counting$^{\lambda}$	33.20	21.20	26.00	**36.80**	**51.2**
Penguins in a Table	37.67	33.56	28.08	**40.41**	**44.5**
Reasoning about Colored Objects	34.80	26.00	31.20	**36.80**	**38.0**
Ruin Names	**56.00**	54.00	52.80	54.80	**76.0**
Salient Translation Error Detection	20.00	20.40	16.40	**23.60**	**48.8**
Snarks	69.66	62.36	69.66	**72.47**	**78.1**
Sports Understanding	**62.80**	53.20	54.40	53.20	**80.4**
Temporal Sequences$^{\lambda}$	29.20	21.20	23.60	**36.80**	**39.6**
Tracking Shuffled Objects$^{\lambda}$ *(avg)*	**25.33**	24.53	24.00	23.47	19.6
Web of Lies$^{\lambda}$	49.20	52.40	**54.00**	51.20	51.2
Word Sorting$^{\lambda}$	4.80	5.20	2.40	**7.60**	**32.0**
NLP Task *(avg)*	54.39	51.63	52.60	54.96	**62.7**
Algorithmic Task$^{\lambda}$ *(avg)*	28.42	27.84	25.37	**33.95**	**40.9**
All Tasks *(avg)*	41.97	40.25	39.58	**44.91**	**52.3**
All Tasks *(WR)*	0.57	0.45	0.39	**0.75**	-

Table 13: BIG-bench hard results using standard 3-shot prompting. Following the convention from Suzgun et al. (2022), we denote algorithmic tasks with the superscript $^{\lambda}$, and present averages for NLP and algorithmic categories. The baseline numbers from PaLM_{540B} (Chowdhery et al., 2022) are taken from the original BBH paper.

- **CommonsenseQA** (Talmor et al., 2019): Multiple-choice QA dataset that requires different types of commonsense knowledge.
- **MMLU** (Hendrycks et al., 2021): Manually collected multiple-choice knowledge questions in 57 subjects.
- **PhysicalQA** (PiQA, Bisk et al., 2020): Questions about how the physical world works.

BloombergGPT achieves the highest performance among BLOOM_{176B}, GPT-NeoX, and OPT_{66B} in one task, and comes second in the other three (Table 14). Similar to the previous section, it outperforms models of similar size while almost being on par with the much larger models. The Massive Multitask Language Understanding (MMLU, Hendrycks et al., 2021) covers 57 different subjects and thus has a much wider coverage than the tasks described above. The aggregated results in Table 15 paint a more consistent picture and follow the insights seen in BIG-bench hard. BloombergGPT consistently outperforms OPT_{66B}, which in turn outperforms GPT-NeoX, while GPT-3 performs best. In contrast

Task	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}	GPT-3
ARC (easy)	73.99	70.79	71.25	**75.93**	71.2
ARC (challenging)	48.63	45.39	44.54	**50.85**	**53.2**
CommonsenseQA	65.52	60.36	**66.42**	64.21	-
PiQA	**77.86**	75.84	77.58	77.04	**80.5**
All Tasks *(avg)*	66.50	63.10	64.95	**67.01**	-
All Tasks *(WR)*	**0.75**	0.08	0.33	0.67	-

Table 14: Knowledge tasks 1-shot results. The baseline numbers from GPT-3 are taken from Brown et al. (2020). Among all models, BloombergGPT achieves the highest win rate among the models we ran ourselves, and performs second best on average.

Model	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}	GPT-3
Humanities	**36.26**	32.75	33.28	34.05	**40.8**
STEM	35.12	33.43	30.72	**36.75**	36.7
Social Sciences	40.04	36.63	38.32	**41.50**	**50.4**
Other	46.36	42.29	42.63	**46.48**	**48.8**
Average	**39.18**	35.95	35.99	39.13	**43.9**

Table 15: Results (5-shot) on the MMLU (Hendrycks et al., 2021) benchmark. The baseline numbers from GPT-3 are taken from Hendrycks et al. (2021). While BloombergGPT lacks behind BLOOM_{176B} on three of the categories, its average is the highest among all models we evaluated ourselves. The gap to GPT-3 is largest on social sciences while the performance in other categories is close.

to the previous sections, BloombergGPT also outperforms BLOOM_{176B} in this category, although by a slim margin. It falls behind the reported performance of GPT-3, especially in the social science category. The gap to GPT-3 is closest in the STEM and “Other” domains which include finance and accounting-related questions.

5.6 Reading Comprehension

We define reading comprehension benchmarks as tasks in which the model can generate the correct response based on information contained in the presented input text. Our grouping includes open-book QA tasks, as opposed to Brown et al. (2020), who separate them into a different categories. We follow the template of Brown et al. (2020), and report accuracy. We include the following tasks:

- **BoolQ** (Clark et al., 2019): Yes/No questions about a passage from Wikipedia.
- **OpenBookQA** (Mihaylov et al., 2018): Multiple-choice elementary-level science questions, given a book of science facts, applied to new situations.
- **RACE** (Lai et al., 2017): A multiple choice dataset of middle and high school English examinations.

RC Scenario	BloombergGPT	GPT-NeoX	OPT_{66B}	BLOOM_{176B}	GPT-3
BoolQ	**74.59**	46.36	57.46	52.94	**76.7**
OpenBookQA	51.60	44.20	**58.00**	47.20	**58.8**
RACE (middle)	**54.32**	41.23	47.42	52.30	**57.4**
RACE (high)	**41.74**	34.33	37.02	39.14	**45.9**
MultiRC	**62.29**	22.86	18.80	26.65	**72.9**
ReCoRD	**82.79**	67.86	82.53	78.01	**90.2**
All Tasks *(avg)*	**61.22**	42.81	50.21	49.37	**67.0**
All Tasks *(WR)*	**0.94**	0.06	0.50	0.50	-

Table 16: Reading Comprehension Results (1-shot). The baseline numbers from GPT-3 are taken from Brown et al. (2020). BloombergGPT far outclasses the models we evaluated ourselves, and is slightly behind GPT-3.

- **Multi-Sentence Reading Comprehension** (MultiRC, Khashabi et al., 2018): Short paragraphs and multi-sentence questions.
- **Reading Comprehension with Commonsense Reasoning** (ReCoRD, Zhang et al., 2018): Automatically generated questions about CNN and Daily Mail news articles.

Table 16 reflects a similar ranking as in the above evaluations: While GPT-3 has the highest performance, BloombergGPT is a close second. Except for OpenBookQA, The performance of BloombergGPT is the highest among BLOOM_{176B}, GPT-NeoX, and OPT_{66B}. Surprisingly, BLOOM_{176B} falls behind significantly in this category.

5.7 Linguistic Tasks

We define as linguistic tasks those scenarios that are not directly connected to user-facing applications. These include tasks that evaluate disambiguation, grammar, or entailment. These tasks are designed to directly assess a model's ability to understand language. We follow the template of Brown et al. (2020), and report accuracy. The list of tasks is as follows:

- **Recognizing Textual Entailment** (RTE, Dagan et al., 2007; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009): Given two text fragments, identify whether the meaning of one text is entailed.
- **Adversarial NLI** (ANLI, Nie et al., 2020): Adversarially constructed entailment detection.
- **CommitmentBank** (CB, De Marneffe et al., 2019): Naturally occurring discourses whose final sentence contains a clause-embedding predicate.
- **Choice of Plausible Alternatives** (COPA, Gordon et al., 2011): Premise and two alternatives, where the task is to select the alternative that more plausibly has a causal relation with the premise.
- **Words in Context** (WIC Pilehvar and Camacho-Collados, 2019): Determine if a word is being used with the same meaning in two sentences.

Linguistic Scenario	BloombergGPT	GPT-NeoX	OPT_{66B}	$\text{BLOOM}_{\text{176B}}$	GPT-3
RTE	**69.31**	53.79	54.87	57.40	**70.4**
ANLI Round 1	32.90	32.60	33.10	**33.60**	32.0
ANLI Round 2	**34.40**	33.80	34.20	33.80	33.9
ANLI Round 3	**37.33**	36.17	34.92	35.17	35.1
CB	**53.57**	48.21	44.64	48.21	**64.3**
COPA	86.00	**88.00**	86.00	84.00	**87.0**
WIC	**52.51**	50.00	**52.51**	50.16	48.6
WinoGrad	80.95	79.12	**82.78**	78.02	**89.7**
WinoGrande	64.09	60.62	66.14	**67.01**	**73.2**
HellaSWAG	**73.92**	68.37	73.47	73.21	**78.1**
StoryCloze	80.87	78.30	**81.83**	80.28	**84.7**
All Tasks *(avg)*	**60.63**	57.18	58.59	58.26	**63.4**
All Tasks *(WR)*	**0.85**	0.27	0.58	0.42	-

Table 17: Results on the Linguistic Scenarios (1-shot). The baseline numbers from GPT-3 are taken from Brown et al. (2020). Win rates and averages are computed only based on accuracy numbers. BloombergGPT consistently scores highest among the models we evaluate, achieving an 85% win rate.

- **Winograd** (Levesque et al., 2011): Determine which word a pronoun refers to when it is semantically unambiguous.
- **Winogrande** (Sakaguchi et al., 2019): Adversarially mined challenging Winograd examples.
- **HellaSWAG** (Zellers et al., 2019): Pick the best ending to a story or set of instructions.
- **StoryCloze** (Mostafazadeh et al., 2016): Select the correct ending sentence for five-sentence long stories.

The results (Table 17) for linguistic tasks follow a similar trend to the knowledge category. BloombergGPT falls slightly behind GPT-3 and outperforms the other models. Similar to the reading comprehension category, $\text{BLOOM}_{\text{176B}}$ falls behind BloombergGPT.

5.8 Summary

Across dozens of tasks in many benchmarks a clear picture emerges. Among the models with tens of billions of parameters that we compare to, BloombergGPT performs the best. Furthermore, in some cases, it is competitive or exceeds the performance of much larger models (hundreds of billions of parameters). While our goal for BloombergGPT was to be a best-in-class model for financial tasks, and we included general-purpose training data to support domain-specific training, the model has still attained abilities on general-purpose data that exceed similarly sized models, and in some cases match or outperform much larger models.

Input: Get me the last price and market cap for Apple
Output: `get(px_last,cur_mkt_cap) for(['AAPL US Equity'])`

Input: Tesla price
Output: `get(px_last) for(['TSLA US Equity'])`

Input: Get the yield and spread for EC527035 Corp and AL580550 Corp
Output: `get(yield,spread) for(['EC527035 Corp','AL580550 Corp'])`

Input: apple and ibm market cap and eps
Output: `get(cur_mkt_cap,is_eps) for(['AAPL US Equity','IBM US Equity'])`

Input: industry subgroup of ibm apple microsoft google
Output: `get(industry_subgroup()) for(['AAPL US Equity','IBM US Equity', 'MSFT US Equity','GOOGL US Equity'])`

Figure 4: Using BloombergGPT to generate valid Bloomberg Query Language. Using only a few examples in a few-shot setting, the model can utilize its knowledge about stock tickers and financial terms to compose valid queries to retrieve the data, given a request in natural language. In each case, the model is given 3 examples (not shown) followed by the 'Input" and a prompt of "Output:".

6 Qualitative Samples

We now share qualitative examples from our model that highlight the benefits of our domain specialization.

Generation of Bloomberg Query Language. One use case for BloombergGPT is to make interactions with financial data more natural. An existing way to retrieve data is via the Bloomberg Query Language (BQL). BQL can be used to interact with different classes of securities, each with its own fields, functions, and parameters. BQL is an incredibly powerful but complex tool. As we show in Figure 4, BloombergGPT can be utilized to make BQL more accessible by transforming natural language queries into valid BQL.

Suggestion of News Headlines. Other use cases that are well supported are in the news space. Since it is trained on many news articles, it can be used for many news applications and assist journalists in their day-to-day work. For example, when constructing newsletters, journalists may have to write short headlines for each new section. While a dedicated model to help with this task may be too expensive to maintain, BloombergGPT performs well out of the box (Figure 5).

Financial Question Answering. Due to the financial domain training data, we are able to query BloombergGPT for knowledge relevant to the financial world. For example, it performs well at identifying the CEO of a company. Figure 6 shows several examples including output from other models. While BloombergGPT correctly identifies the CEOs, GPT-NeoX does not, and FLAN-T5-XXL completely fails, consistently ignoring the com-

Input: The US housing market shrank in value by $2.3 trillion, or 4.9%, in the second half of 2022, according to Redfin. That's the largest drop in percentage terms since the 2008 housing crisis, when values slumped 5.8% during the same period.
Output: Home Prices See Biggest Drop in 15 Years

Input: The global economy is in a better place today than many predicted months ago, Janet Yellen said at the G20. At home, she pointed to a resilient US economy, where headline inflation has moderated and the labor market is strong. She also called for the IMF to move swiftly toward a fully-financed program for Ukraine.
Output: Yellen Sees Global Economy More Resilient Than Expected

Input: Google was sued by the US and eight states seeking the breakup of its ad-tech business for allegedly monopolizing the digital advertising market. The lawsuit is the Biden administration's first big challenge to a tech titan and one of the rare times since 1982 that the DOJ has sought to cleave up a major company.
Output: Google Sued for Monopoly in Online Ad Market

Figure 5: Using BloombergGPT to generate short headline suggestions in a three-shot setting. Bloomberg News sends many newsletters a day that requires these headlines. BloombergGPT could help with the editing process by suggesting initial headlines from the text.

pany and instead predicting the CEO at Cirrus Logic who was included in the prompt. While BloombergGPT does not perfectly solve this task and makes mistakes, we were not able to find any example where the other models solved the task while BloombergGPT did not.

7 Related Work

Language Models. Language modeling has a long history in the NLP community. The idea of training a probabilistic language model for scoring word sequences was likely first introduced by Jelinek (1976). N-gram models were popular for decades (Brown et al., 1992), and were trained on corpora up to 2 trillion tokens (Brants et al., 2007). Research on training language models accelerated over the last decade due to innovations in machine learning, data availability, and compute. Early work in autoregressive language modeling (e.g., Mikolov et al., 2010; Sutskever et al., 2011) used recurrent neural networks, but these were small models trained on small datasets. The introduction of the transformer architecture (Vaswani et al., 2017) facilitated the scaling of these models in terms of data, compute, and the number of parameters.

The process of developing models that could better approximate the distribution of language over large corpora led to the discovery that the representations these models produce are useful starting points for many downstream tasks. This was demonstrated by Radford et al. (2018) and Howard and Ruder (2018) who showed that generative pretraining

Input: CEO of Assicurazioni Generali?
BloombergGPT: Philippe Donnet
GPT-NeoX: Antonio De Lorenzo, Simone Gambarini, Enrico Zanetti
FLAN-T5-XXL: John M Forsyth, Christopher K Peters, {empty string}

Input: CEO of Silicon Valley Bank?
BloombergGPT: Greg Becker
GPT-NeoX: John Hodor, Mike Fridgen, Peter R Briger
FLAN-T5-XXL: Christian Klein, John M Forsyth, John M Forsyth

Input: CEO of IHS Markit Ltd?
BloombergGPT: Lance Uggla
GPT-NeoX: John F Lechleiter, David Steward, Niall O Broin
FLAN-T5-XXL: William M Hawkins, John M Forsyth, John M Forsyth

Input: CEO of Walgreens Boots Alliance?
BloombergGPT: Rosalind Brewer
GPT-NeoX: Mark V Dei, Stefano Pessina, Stefano Pessina
FLAN-T5-XXL: Christian Klein, John M Forsyth, John M Forsyth

Input: CEO of Citigroup Inc?
BloombergGPT: Jane Fraser
GPT-NeoX: Michael L Corbat, Michael L Corbat, Michael L Corbat*
FLAN-T5-XXL: Christian Sewing, John M Forsyth, John M Forsyth

Figure 6: Testing the ability of BloombergGPT, GPT-NeoX, and FLAN-T5-XXL to recall the names of CEOs of companies. Each model is run in a 10-shot setting. We sample up to three answers and present all of them if they are incorrect. *Michael Corbat was CEO of Citigroup until 2021, highlighting the importance of an up-to-date model.

with an autoregressive language modeling objective achieves strong performance in transfer learning. Radford et al. (2019) further showed scaling the model size and training data led to autoregressive language models that perform well in different downstream tasks without any additional supervised fine-tuning.

Brown et al. (2020) showed that further scaling the models led to the emergence of new model capabilities and increased model robustness. Since the release of GPT-3 by Brown et al. (2020), many other researchers built large language models to study data quantity, data quality, network architecture, parameter scaling, data scaling, tokenization, and open-sourcing strategies (Raffel et al., 2020; Zhang et al., 2022a; Black et al., 2022; Rae et al., 2021; Hoffmann et al., 2022; Chowdhery et al., 2022; Lieber et al., 2021; Zeng et al., 2022; Tafjord and Clark, 2021; Smith et al., 2022; Scao et al., 2022; Taylor et al., 2022; Lin et al., 2022; Soltan et al., 2022; Thoppilan et al., 2022; Bao et al., 2022; Sanh et al., 2022; Roller et al., 2021; Glaese et al., 2022; Wang et al., 2021; Peng et al., 2022, among many others).

Domain-Specific Large Language Models. The value of domain-specific training for masked (encoder only) language models is well established. Commonly accepted approaches are to train BERT models (Devlin et al., 2019) from scratch on domain-specific data or to continue pretraining an existing model on new domain-specific data (Gururangan et al., 2020). Following these strategies, BioBERT (Lee et al., 2020) adapts BERT to the biomedical domain and SciBERT is trained on scientific publications (Beltagy et al., 2019). The results of these papers showed that in-domain training allows models to outperform previous state-of-the-art models in a variety of biomedical text mining tasks. Further examples of this paradigm are ClinicalBERT for the clinical domain (Huang et al., 2019), BioMed-RoBERTa for scientific biomedical papers (Gururangan et al., 2020), and BERTweet and Bernice for Twitter data (Nguyen et al., 2020; DeLucia et al., 2022).

Since the training of auto-regressive—decoder-only—language models of more than 10B parameters is significantly more costly than training masked LMs under 1B parameters, there have been much fewer examples of domain-specific autoregressive models. However, existing approaches follow the same two strategies. Adapting an existing model, medPaLM (Singhal et al., 2022) adapted PaLM to the biomedical domain and Minerva (Lewkowycz et al., 2022) to mathematical reasoning tasks.

Recently, several examples of from-scratch trained decoder-only models for domain-specific data have emerged. One popular domain is protein sequences since they can be represented using language-like sequences but are not covered by natural language models (e.g., Lin et al., 2022; Xiao et al., 2021; Nijkamp et al., 2022). However, there can be benefits even for models in natural language domains. Galactica is trained exclusively on a large collection of scientific datasets, and includes special processing to handle scientific notations (Taylor et al., 2022). While performing very well on scientific tasks, Galactica also surprisingly also performs well on more standard NLP tasks. BioGPT (Luo et al., 2022) and BioMedLM (Bolton et al., 2023) are both smaller GPT-style models trained on biomedical data. Lehman et al. (2023) compares encoder/decoder models trained exclusively on domain-specific data, versus those adapted from general-purpose training. Researchers working on large generative language dialog models have reached similar conclusions about the benefits of using domain-specific training data (Zhang et al., 2020; Roller et al., 2021; Thoppilan et al., 2022).

These findings highlight the advantages of in-domain pretraining, especially if sufficient data is available, as it is in our case. Inspired by the general capabilities of Galactica, we augment our private data with public data with the goal of investigating whether a model can gain in-domain capabilities without sacrificing general-domain performance.

Training Data. Large corpora of raw text data are critical for training LLMs. As a result, there are now several corpora available that cover a wide range of sources.

The Colossal Clean Crawled Corpus (C4, Raffel et al., 2020) draws from Common Crawl to create a processed training corpus. The Pile is a carefully curated corpus that contains a wide range of data sources (Gao et al., 2021). These datasets are built on or include web crawls (OpenWebText2) augmented with an array of data from high-quality sources (Pubmed, Arxiv). Various efforts aim to clean datasets, especially web data, by removing unwanted or harmful text (Touvron et al., 2023; Rae et al., 2020). BLOOM (Scao et al., 2022) carefully selected data sources and included various filtering mechanisms (Jernite et al., 2022).

While web data is an effective strategy for obtaining large amounts of diverse data, robust cleaning efforts still result in data artifacts, duplicates (Carlini et al., 2020), various types of toxic language (Welbl et al., 2021), and it can lead to unintended marginalization of minority voices (Xu et al., 2021). Dodge et al. (2021) studied C4 to better understand the metadata, and the included and excluded data. Their findings suggest that C4 contains machine-generated text, is biased due to exclusion filters and might contain examples drawn from evaluation datasets for NLP tasks. A similar effort was undertaken by Zeng et al. (2022) to document the pre-processing they undertook to train their Chinese large language model.

Lee et al. (2022a) investigated the impact of deduplication on model performance for several datasets and found that deduplication reduces the emission of memorized training data, allows better estimation of the generalization error, and improves training time and cost without impacting performance. These insights highlight the importance and challenges of constructing high-quality training corpora. As discussed in §2, Bloomberg's core business curates and provides access to datasets, which we use to construct a high-quality dataset FinPile to train BloombergGPT, resulting in best-in-class financial performance.

Evaluation. The tasks addressed by language models have vastly increased and require a very different evaluation process from traditional task-specific systems. There have been two paradigms for LLM evaluation: The first is to evaluate a model in many different scenarios via automatic evaluation (Liang et al., 2022; Srivastava et al., 2022) and the second is to perform extrinsic and task-specific evaluations by integrating them into user workflows (e.g., Lee et al., 2022b; Goyal et al., 2022).

While the second strategy is necessary for assessing deployments of models in products, it is infeasible to run these human evaluations at a scale of the first strategy and it is thus standard to follow the first strategy when introducing new models. In our case, we combine multiple general-purpose evaluations from multiple existing benchmarks that have different goals. Srivastava et al. (2022) aim for maximum coverage by soliciting tasks from the entire research community, while HELM (Liang et al., 2022) suggests evaluation in various "scenarios" that are represented through specific datasets. Earlier language model papers developed their own evaluation schemata (Brown et al., 2020). While these benchmarks allow for a side-by-side comparison between models, it is challenging to ensure that all experimental parameters (prompts, decoding strategies, few-shot examples, etc.) are the same. For that reason, we differentiate between reported and verified numbers in our evaluation (§5).

Beyond the general-purpose evaluation, we also require a targeted domain evaluation. Prior domain-specific models like Galactica (Taylor et al., 2022) chose a set of tasks that the model is likely to perform well on. In their case, these were various scientific tasks. However, there exists no standard benchmark for the financial NLP domain. While the recent work on FLUE (Shah et al., 2022) aims to provide such a benchmark, it has limited coverage of relevant tasks, no suggested evaluation strategy for few-shot learning, and the quality of some annotations is low. To provide externally comparable results, we developed a few-shot strategy for FLUE, but also decided to augment the publicly available evaluation tasks with company-internal benchmarks.

Model Size. Large language model training remains expensive in terms of the computational cost and human effort to assemble data and train the model. Determining the optimal amount of training data and model shape and size for the best utilization of resources becomes important.

Kaplan et al. (2020) first studied the dependence of language model performance on architecture, parameter size, compute power, and dataset size. They reported that the number of model parameters, the dataset size, and the amount of compute improves performance on the autoregressive language modeling objective smoothly according to the power law. A similar investigation by Hernandez et al. (2021) into data transfer for differing distributions found that this also follows a power law. Moving beyond studying the effect on loss, Rae et al. (2021) analyzed the effect of scale on undesirable properties such as bias and toxicity by training a wide range of model sizes.

Comparing model architectures, Levine et al. (2020) studied the scaling of models that use self-attention and derived guidelines for depth-to-width allocation. Tay et al. (2021) reported that model shape (depth-width ratio) impacted performance on downstream tasks even if it had minimal impact on the pretraining objective. Tay et al. (2022a) further studied the effect of scaling for different model architectures and showed that architecture choice is pertinent when scaling and that the vanilla transformer architecture scales best.

Of particular importance to this work is the study of Hoffmann et al. (2022), who investigated the effect of model size and the number of training tokens on the performance of a model given a fixed compute budget. They posited that existing large language models were undertrained and that model size and the number of training tokens should be scaled equally. They demonstrated this hypothesis through Chinchilla, a model significantly smaller, yet higher performing, than most of the largest LLMs. These findings opened the door for "Chinchilla optimal" training of smaller models that achieve strong performance, and for which inference can be run much more efficiently than for their larger counterparts. These findings led us to consider a nearly Chinchilla-optimal model using a standard architecture.

Tokenization. Tokenization and vocabulary choice play a critical role in model performance as they can help the model learn meaningful representations and generalize to unseen words. Byte-Pair encoding (BPE) (Sennrich et al., 2016) learns a greedy bottom-up vocabulary by repeatedly merging the most frequent sequence pairs in the training set till a predetermined vocabulary size is reached. Radford et al. (2018) adapted BPE by limiting the base vocabulary to be all possible bytes as opposed to all Unicode characters. Wordpiece tokenization (Schuster and Nakajima, 2012) also learns a greedy bottom-up vocabulary by repeatedly merging the sequence-pair that maximizes the likelihood of the training data, which is a slight deviation from the method in Sennrich et al. (2016).

In contrast to BPE and Wordpiece, the Unigram tokenizer (Kudo, 2018) learns a top-down vocabulary by first initializing a large vocabulary and repeatedly discarding those vocabulary items that increase loss (e.g., log-likelihood of the training data) the least. By construction, the Unigram model can tokenize an input text in several different ways. That is, the Unigram model saves probabilities allowing for smarter tokenization at inference time.

Finally, SentencePiece (Kudo and Richardson, 2018) adapts the schemes mentioned above to handle languages that are not space separated. Beltagy et al. (2019) constructed a vocabulary specific to scientific text and observed that their domain-specific trained vocabulary only had a 42% overlap with the non-domain-specific BERT vocabulary trained on general domain text. Similarly, Lewis et al. (2020) showed that a dedicated biomedical vocabulary improved performance on sequence labeling tasks consistently. Lieber et al. (2021) constructed a larger vocabulary to ensure token efficiency, which the authors claim resulted in reduced training time and better semantic representation. These findings demonstrate the importance of selecting a tokenizer and accompanying vocabulary that best reflects that training domain. For those reasons, we decided to train our own unigram tokenizer instead of relying on existing public ones.

Positional Embeddings. Transformer-based models rely on positional embeddings to encode position and location information of words in a text. Encoding the sequence position and the effect of this choice on model performance have been studied extensively. These include sinusoidal embeddings (Vaswani et al., 2017), rotary position embeddings (Su et al., 2021a), adding relative position bias (Raffel et al., 2020), and adding linear biases to attention heads (Press et al., 2022). A side-effect of the strategy in Press et al. (2022) is that one can train on shorter sequences without loss in performance on longer sequences. This has two benefits: first, models learn to generalize (extrapolate) to longer sequences and second, models can be trained on shorter sequences reducing training time.

8 Ethics, Limitations, and Implications

The rapid development and adoption of large language models have been accompanied by a rigorous conversation about the ethics, uses, and limitations of these models. For a more complete treatment of these topics, we direct the reader to Bommasani et al. (2021); Bender et al. (2021); Birhane et al. (2022); Weidinger et al. (2021, 2022). We discuss issues that are directly relevant to the development of BloombergGPT.

8.1 Ethical Use

Finance is a sensitive area for technology, and ensuring accurate, factual information is crucial for our products, our clients, and the firm's reputation in the marketplace. On the other hand, our clients are also eager to adopt state-of-the-art technology to support their workflows. To provide natural language applications to the financial community, we have developed a rigorous risk and testing assessment process. This process includes careful annotation guidelines (Tseng et al., 2020), pre-launch review at multiple levels by the central risk and compliance organizations, and by the product leaders (e.g., the newsroom) as applicable, and post-launch monitoring. Moreover, we conduct our research, development, and deployment of NLP and AI systems in accordance with all applicable regulations.

Similarly, toxicity and bias are areas where, as a company, we take extraordinary care with any content we produce, whether from humans or machines. Since the measurement of toxicity and bias in our model depends on its application areas, quantifying the potential for the generation of harmful language remains an open question. We are particularly interested in studying whether FinPile, which is cleaner and contains fewer examples of overtly biased or toxic language (e.g., Press Releases), reduces the proclivity of the model to generate inappropriate content. As we move to develop products built on this technology, we will apply existing testing procedures, as well as risk and compliance controls, to ensure safe use.

8.2 Openness

An ongoing debate in the community concerns how LLMs should be released, if at all. While models that are not publicly available cannot be fully evaluated by the community, distributing models can lead to nefarious purposes. Especially for a model like BloombergGPT, which is trained on a significant amount of press releases, news articles, and filings, a release carries a high risk for abuse through imitation.

We have witnessed many different strategies to mitigate risks associated with the release of LLMs. One strategy is to freely and openly share trained models (Scao et al., 2022), and rely on a license that dictates how the model should and should not be used. Another requires individuals to apply for access to the trained model parameters (Zhang et al., 2022a; Touvron et al., 2023). A more restrictive approach is to provide API access to models, but no access to the underlying model parameters or detailed information on the data the model was trained on (Brown et al., 2020). Finally, some have provided no access to the model (Chowdhery et al., 2022; Hoffmann et al., 2022). Each decision reflects a combination of factors, including model use, potential harms, and business decisions.

One of Bloomberg's core business propositions is around providing access to data that has been collected over the course of decades. As is well known, LLMs are susceptible to data leakage attacks and it is possible to extract significant segments of text given model weights (Carlini et al., 2020, 2022). Moreover, even giving selective access to researchers isn't a guarantee that the model cannot be leaked. Without strong privacy guarantees, we must be concerned that providing access to model weights entails giving access to FinPile. For this reason, we err on the side of caution and follow the practice of other LLM developers in not releasing our model.

Nevertheless, our insights and experiences in training and evaluating BloombergGPT contribute to the developing understanding of these models. In particular, our experience may be useful to those building domain-specific models. During the process of developing BloombergGPT, we found the OPT chronicles, experiences of the BLOOM team, as well as work of non-open models like GPT-3, PaLM, Chinchilla, and Gopher, to be crucial enablers of our work. In support of this tradition, we include our Training Chronicles (Appendix C).

9 Conclusion

We have presented BloombergGPT, a best-in-class LLM for financial NLP.

Our model contributes to the ongoing dialog on effective ways to train domain-specific models. Our training strategy of mixing domain-specific and general-purpose data results in a model that balances performance in both domains. Additionally, our work offers another data point on selecting Chinchilla optimal-sized models. Finally, we hope that our model training logs will provide a guide for those training their own LLMs.

We have several interesting directions to pursue. First, task fine-tuning has yielded significant improvements in LLMs, and we plan to consider what unique opportunities exist for model alignment in the financial domain (Wei et al., 2021; Ouyang et al., 2022). Second, by training on data in FinPile, we are selecting data that may exhibit less toxic and biased language. The effects of this on the final model are unknown as yet, which we plan to test. Third, we seek to understand how our tokenization strategy changes the resulting model. These are a few of the new research directions we hope to pursue with BloombergGPT.

We achieve strong results on general LLM benchmarks and outperform comparable models on financial tasks. We attribute this, in decreasing order of impact, to 1. a well-curated internal dataset, 2. our unique choice in tokenizer, and 3. an up-to-date architecture. We will continue to develop financial applications with BloombergGPT to further explore the benefits of these modeling choices.

Acknowledgments and Disclosure of Funding

We would like to acknowledge the people who helped us, including Emmanuel Scoullos (NVIDIA) and Can Karakus (Amazon Web Services).

References

Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. *arXiV preprint arXiV:1908.10063*, 2019.

Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhihua Wu, Zhen Guo, Hua Lu, Xinxian Huang, Xin Tian, Xinchao Xu, Yingzhan Lin, and Zheng-Yu Niu. PLATO-XL: Exploring the large-scale pre-training of dialogue generation. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 107–118, Online only, November 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-aacl.10.

Iz Beltagy, Kyle Lo, and Arman 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, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1371. URL https://aclanthology.org/D19-1371.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST, 2009. URL https://tac.nist.gov/publications/2009/additional.papers/RTE5_overview.proceedings.pdf.

Abeba Birhane, Pratyusha Kalluri, Dallas Card, William Agnew, Ravit Dotan, and Michelle Bao. The values encoded in machine learning research. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 173–184, 2022.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6239`.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL `https://doi.org/10.5281/zenodo.5297715`. If you use this software, please cite it using these metadata.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.bigscience-1.9. URL `https://aclanthology.org/2022.bigscience-1.9`.

Elliot Bolton, David Hall, Michihiro Yasunaga, Tony Lee, Chris Manning, and Percy Liang. BioMedLM. `https://github.com/stanford-crfm/BioMedLM`, 2023.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel J. Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *ArXiV*, abs/2108.07258, 2021.

Kaj Bostrom and Greg Durrett. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.414. URL `https://aclanthology.org/2020.findings-emnlp.414`.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `https://aclanthology.org/D07-1090`.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18 (4):467–480, 1992.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Xiaodong Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security Symposium*, 2020.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models, 2022. URL `https://arxiv.org/abs/2202.07646`.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W.

Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, S. Arun Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiV*, abs/2107.03374, 2021a.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiV preprint arXiV:1604.06174*, 2016.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.300. URL `https://aclanthology.org/2021.emnlp-main.300`.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. ConvFinQA: Exploring the chain of numerical reasoning in conversational finance question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6292, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.emnlp-main.421`.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *arXiV*, abs/2204.02311, 2022.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL `https://aclanthology.org/N19-1300`.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiV*, abs/1803.05457, 2018.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, 2007.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, pages 107–124, 2019.

Alexandra DeLucia, Shijie Wu, Aaron Mueller, Carlos Aguirre, Philip Resnik, and Mark Dredze. Bernice: A multilingual pre-trained encoder for Twitter. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6191–6205, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.emnlp-main.415`.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. In *International Conference on Learning Representations*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.98. URL `https://aclanthology.org/2021.emnlp-main.98`.

Mark Dredze, Prabhanjan Kambadur, Gary Kazantsev, Gideon Mann, and Miles Osborne. How twitter is changing the nature of financial news discovery. In *proceedings of the second international workshop on data science for macro-modeling*, pages 1–5, 2016.

Ingrid E Fisher, Margaret R Garnsey, and Mark E Hughes. Natural language processing in accounting, auditing and finance: A synthesis of the literature with a roadmap for future research. *Intelligent Systems in Accounting, Finance and Management*, 23(3):157–214, 2016.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2021. URL `https://arxiv.org/abs/2101.00027`.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text, 2022. URL `https://arxiv.org/abs/2202.06935`.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, June 2007. Association for Computational Linguistics. URL `https://aclanthology.org/W07-1401`.

Amelia Glaese, Nat McAleese, Maja Trebacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Soňa Mokrá, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William Isaac, John Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks, and Geoffrey Irving. Improving alignment of dialogue agents via targeted human judgements, 2022. URL `https://arxiv.org/abs/2209.14375`.

Andrew S. Gordon, Zornitsa Kozareva, and Melissa Roemmele. Semeval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *International Workshop on Semantic Evaluation*, 2011.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. News summarization and evaluation in the era of gpt-3, 2022. URL `https://arxiv.org/abs/2209.12356`.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah 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. doi: 10.18653/v1/2020.acl-main.740. URL `https://aclanthology.org/2020.acl-main.740`.

R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, volume 7, 2006.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiV preprint arXiV:1606.08415*, 2016.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=d7KBjmI3GmQ`.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4246–4253, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.379. URL https://aclanthology.org/2020.findings-emnlp.379.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiV preprint arXiV:2102.01293*, 2021.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack William Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=iBBcRUlOAPR.

Jeremy Howard and Sebastian 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. doi: 10.18653/v1/P18-1031. URL https://aclanthology.org/P18-1031.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiV*, 4 2019. URL http://arxiv.org/abs/1904.05342.

Frederick Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976.

Yacine Jernite, Huu Nguyen, Stella Biderman, Anna Rogers, Maraim Masoud, Valentin Danchev, Samson Tan, Alexandra Sasha Luccioni, Nishant Subramani, Isaac Johnson, Gerard Dupont, Jesse Dodge, Kyle Lo, Zeerak Talat, Dragomir Radev, Aaron Gokaslan, Somaieh Nikpoor, Peter Henderson, Rishi Bommasani, and Margaret Mitchell. Data governance in the age of large-scale data-driven language technology. In *2022 ACM Conference on Fairness, Accountability, and Transparency*. ACM, jun 2022. doi: 10.1145/3531146.3534637. URL https://doi.org/10.1145%2F3531146.3534637.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiV*, 1 2020. URL http://arxiv.org/abs/2001.08361.

Can Karakus, Rahul Huilgol, Fei Wu, Anirudh Subramanian, Cade Daniel, Derya Cavdar, Teng Xu, Haohan Chen, Arash Rahnama, and Luis Quintela. Amazon sagemaker model parallelism: A general and flexible framework for large model training. *arXiV preprint arXiV:2111.05972*, 2021.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. 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 252–262, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1023. URL `https://aclanthology.org/N18-1023`.

Vijay Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. Reducing activation recomputation in large transformer models, 2022. URL `https://arxiv.org/abs/2205.05198`.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL `https://aclanthology.org/P18-1007`.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL `https://aclanthology.org/D18-2012`.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1082. URL `https://aclanthology.org/D17-1082`.

Teven Le Scao, Thomas Wang, Daniel Hesslow, Stas Bekman, M Saiful Bari, Stella Biderman, Hady Elsahar, Niklas Muennighoff, Jason Phang, Ofir Press, Colin Raffel, Victor Sanh, Sheng Shen, Lintang Sutawika, Jaesung Tae, Zheng Xin Yong, Julien Launay, and Iz 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, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.54`.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36:1234–1240, 2 2020. ISSN 14602059. doi: 10.1093/bioinformatics/btz682.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445, Dublin, Ireland, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.577. URL `https://aclanthology.org/2022.acl-long.577`.

Mina Lee, Megha Srivastava, Amelia Hardy, John Thickstun, Esin Durmus, Ashwin Paranjape, Ines Gerard-Ursin, Xiang Lisa Li, Faisal Ladhak, Frieda Rong, Rose E. Wang, Minae Kwon, Joon Sung Park, Hancheng Cao, Tony Lee, Rishi Bommasani, Michael S. Bernstein, and Percy Liang. Evaluating human-language model interaction. *CoRR*, abs/2212.09746, 2022b. doi: 10.48550/arXiv.2212.09746. URL `https://doi.org/10.48550/arXiv.2212.09746`.

Eric Lehman, Evan Hernandez, Diwakar Mahajan, Jonas Wulff, Micah J. Smith, Zachary Ziegler, Daniel Nadler, Peter Szolovits, Alistair Johnson, and Emily Alsentzer. Do we still need clinical language models?, 2023. URL `https://arxiv.org/abs/2302.08091`.

Hector J. Levesque, Ernest Davis, and L. Morgenstern. The winograd schema challenge. In *International Conference on Principles of Knowledge Representation and Reasoning*, 2011.

Yoav Levine, Noam Wies, Or Sharir, Hofit Bata, and Amnon Shashua. Limits to depth efficiencies of self-attention. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22640–22651. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/ff4dfdf5904e920ce52b48c1cef97829-Paper.pdf`.

Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. Pretrained language models for biomedical and clinical tasks: Understanding and extending the state-of-the-art. In *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, pages 146–157, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.clinicalnlp-1.17. URL `https://aclanthology.org/2020.clinicalnlp-1.17`.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL `https://arxiv.org/abs/2206.14858`.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel J. Orr, Lucia Zheng, Mert Yüksekgönül, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *CoRR*, abs/2211.09110, 2022. doi: 10.48550/arXiv.2211.09110. URL `https://doi.org/10.48550/arXiv.2211.09110`.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. Jurassic-1: Technical details and evaluation. *White Paper. AI21 Labs*, 1, 2021.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, and Alexander Rives. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022. doi: 10.1101/2022.07.20.500902. URL `https://www.biorxiv.org/content/early/2022/07/21/2022.07.20.500902`.

Tianyu Liu, Yuchen Eleanor Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. Autoregressive structured prediction with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.70`.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. BioGPT: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6), sep 2022. doi: 10.1093/bib/bbac409. URL `https://doi.org/10.1093%2Fbib%2Fbbac409`.

Jouni Luoma and Sampo Pyysalo. Exploring cross-sentence contexts for named entity recognition with BERT. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 904–914, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.78. URL `https://aclanthology.org/2020.coling-main.78`.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. Www'18 open challenge: Financial opinion mining and question answering. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1941–1942. ACM, 2018. doi: 10.1145/3184558.3192301. URL `https://doi.org/10.1145/3184558.3192301`.

Pekka Malo, Ankur Sinha, Pekka J. Korhonen, Jyrki Wallenius, and Pyry Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *J. Assoc. Inf. Sci. Technol.*, 65(4):782–796, 2014. doi: 10.1002/asi.23062. URL `https://doi.org/10.1002/asi.23062`.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp, 2021. URL `https://arxiv.org/abs/2112.10508`.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages

2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, pages 1045–1048. Makuhari, 2010.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1098. URL https://aclanthology.org/N16-1098.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. BERTweet: A pre-trained language model for English tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.2. URL https://aclanthology.org/2020.emnlp-demos.2.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.441. URL https://aclanthology.org/2020.acl-main.441.

Erik Nijkamp, Jeffrey Ruffolo, Eli N. Weinstein, Nikhil Naik, and Ali Madani. Progen2: Exploring the boundaries of protein language models. *CoRR*, abs/2206.13517, 2022. doi: 10.48550/arXiv.2206.13517. URL https://doi.org/10.48550/arXiv.2206.13517.

NVIDIA. Train with mixed precision, 2023. URL https://docs.nvidia.com/deeplearning/performance/mixed-precision-training/index.html.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=TG8KACxEON.

Baolin Peng, Michel Galley, Pengcheng He, Chris Brockett, Lars Liden, Elnaz Nouri, Zhou Yu, Bill Dolan, and Jianfeng Gao. Godel: Large-scale pre-training for goal-directed dialog. *arXiV preprint arXiV:2206.11309*, 2022.

Mohammad Taher Pilehvar and Jose 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. doi: 10.18653/v1/N19-1128. URL `https://aclanthology.org/N19-1128`.

Ofir Press, Noah Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=R8sQPpGCv0`.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. URL `https://gluebenchmark.com/leaderboard`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. URL `https://github.com/codelucas/newspaper`.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=SylKikSYDH`.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis & insights from training gopher. *arXiV*, 12 2021. URL `http://arxiv.org/abs/2112.11446`.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140): 1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference* *for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.24. URL `https://aclanthology.org/2021.eacl-main.24`.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WINOGRANDE: An adversarial winograd schema challenge at scale. *Commun. ACM*, 64: 99–106, 2019.

Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. Domain adaption of named entity recognition to support credit risk assessment. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 84–90, Parramatta, Australia, December 2015. URL `https://aclanthology.org/U15-1010`.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=9Vrb9D0WI4`.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank

Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán,

Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model. *arXiV*, 11 2022. URL `http://arxiv.org/abs/2211.05100`.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL `https://aclanthology.org/P16-1162`.

Raj Shah, Kunal Chawla, Dheeraj Eidnani, Agam Shah, Wendi Du, Sudheer Chava, Natraj Raman, Charese Smiley, Jiaao Chen, and Diyi Yang. When FLUE meets FLANG: Benchmarks and large pretrained language model for financial domain. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2322–2335, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.emnlp-main.148`.

Noam Shazeer. GLU variants improve transformer. *arXiV preprint arXiV:2002.05202*, 2020.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiV preprint arXiV:1909.08053*, 2019.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. Large language models encode clinical knowledge, 2022. URL `https://arxiv.org/abs/2212.13138`.

Ankur Sinha and Tanmay Khandait. Impact of news on the commodity market: Dataset and results. *CoRR*, abs/2009.04202, 2020. URL `https://arxiv.org/abs/2009.04202`.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model, 2022. URL `https://arxiv.org/abs/2201.11990`.

Saleh Soltan, Shankar Ananthakrishnan, Jack G. M. FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith S. Peris, Stephen Rawls, Andrew Rosenbaum, Anna Rumshisky, Chandan Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Premkumar Natarajan. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *arXiV*, abs/2208.01448, 2022.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Annasaheb Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Santilli, Andreas Stuhlmuller, Andrew M. Dai, Andrew D. La, Andrew Kyle Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakacs, Bridget R. Roberts, Bao Sheng Loe, Barret Zoph, Bartlomiej Bojanowski, Batuhan Ozyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Stephen Howald, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, C'esar Ferri Ram'irez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Tatiana Ramirez, Clara Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Daniel H Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Gonz'alez, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, D. Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth P. Donoway, Ellie Pavlick, Emanuele Rodolà, Emma FC Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan J. Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fan Xia, Fatemeh Siar, Fernando Mart'inez-Plumed, Francesca Happ'e, François Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-L'opez, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Han Sol Kim, Hannah Rashkin, Hanna Ha-

jishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hubert Wong, Ian Aik-Soon Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, John Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, J. Brooker Simon, James Koppel, James Zheng, James Zou, Jan Koco'n, Jana Thompson, Jared Kaplan, Jarema Radom, Jascha Narain Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jenni Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Oluwadara Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Jane W Waweru, John Burden, John Miller, John U. Balis, Jonathan Berant, Jorg Frohberg, Jos Rozen, José Hernández-Orallo, Joseph Boudeman, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Ochieng' Omondi, Kory Wallace Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Luca Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Col'on, Luke Metz, Lutfi Kerem cSenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Madotto Andrea, Maheen Saleem Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, M Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew Leavitt, Matthias Hagen, M'aty'as Schubert, Medina Baitemirova, Melissa Arnaud, Melvin Andrew McElrath, Michael A. Yee, Michael Cohen, Mi Gu, Michael I. Ivanitskiy, Michael Starritt, Michael Strube, Michal Swkedrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Monica Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, T MukundVarma, Nanyun Peng, Nathan Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas S. Roberts, Nicholas Doiron, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter W. Chang, Peter Eckersley, Phu Mon Htut, Pi-Bei Hwang, P. Milkowski, Piyush S. Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, QING LYU, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ram'on Risco Delgado, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan Le Bras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib J. Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Sam Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi S. Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo hwan Lee, Spencer Bradley Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Rose Biderman, Stephanie C. Lin, S. Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Mish-

erghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq A. Ali, Tatsuo Hashimoto, Te-Lin Wu, Theo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, T. N. Kornev, Timothy Telleen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler O'Brien Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Venkatesh Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, W Vossen, Xiang Ren, Xiaoyu Tong, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yang Song, Yasaman Bahri, Ye Ji Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yu Hou, Yuntao Bai, Zachary Seid, Zhao Xinran, Zhuoye Zhao, Zi Fu Wang, Zijie J. Wang, Zirui Wang, Ziyi Wu, Sahib Singh, and Uri Shaham. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv*, abs/2206.04615, 2022.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiV preprint arXiV:2104.09864*, 2021a.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864, 2021b. URL `https://arxiv.org/abs/2104.09864`.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1017–1024, 2011.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *CoRR*, abs/2210.09261, 2022. doi: 10.48550/arXiv.2210.09261. URL `https://doi.org/10.48550/arXiv.2210.09261`.

Oyvind Tafjord and Peter Clark. General-purpose question-answering with macaw. *arXiV preprint arXiV:2109.02593*, 2021.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL `https://aclanthology.org/N19-1421`.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale efficiently: Insights from pre-training and fine-tuning transformers. *arXiV preprint arXiV:2109.10686*, 2021.

Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Won Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Q Tran, Dani Yogatama, and Donald Metzler. Scaling laws vs model architectures: How does inductive bias influence scaling? *arXiV preprint arXiV:2207.10551*, 2022a.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. Ul2: Unifying language learning paradigms, 2022b. URL `https://arxiv.org/abs/2205.05131`.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *arXiV*, 11 2022. URL `http://arxiv.org/abs/2211.09085`.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. Lamda: Language models for dialog applications, 2022. URL `https://arxiv.org/abs/2201.08239`.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003. URL `https://aclanthology.org/W03-0419`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models, 2023. URL `https://arxiv.org/abs/2302.13971`.

Tina Tseng, Amanda Stent, and Domenic Maida. Best practices for managing data annotation projects, 2020. URL `http://rgdoi.net/10.13140/RG.2.2.34497.58727`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von 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. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

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

Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.685. URL `https://aclanthology.org/2021.emnlp-main.685`.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2021. URL `https://arxiv.org/abs/2109.01652`.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research (TMLR)*, 2022a. doi: 10.48550/ARXIV.2206.07682. URL `https://arxiv.org/abs/2206.07682`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022b. URL `https://openreview.net/forum?id=_VjQlMeSB_J`.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Ethical and social risks of harm from language models, 2021. URL `https://arxiv.org/abs/2112.04359`.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John F. J. Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sande Minnich Brown, Zachary Kenton, William T. Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William S. Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Taxonomy of risks posed by language models. *2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022.

Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. Challenges in detoxifying language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2447–2469, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.210. URL `https://aclanthology.org/2021.findings-emnlp.210`.

Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. 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 833–844, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1077. URL `https://aclanthology.org/D19-1077`.

Yonghui Wu, Mike Schuster, Z. Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason R. Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiV*, abs/1609.08144, 2016.

Yijia Xiao, Jiezhong Qiu, Ziang Li, Chang-Yu Hsieh, and Jie Tang. Modeling protein using large-scale pretrain language model. *CoRR*, abs/2108.07435, 2021. URL `https://arxiv.org/abs/2108.07435`.

Frank Z Xing, Erik Cambria, and Roy E Welsch. Natural language based financial forecasting: a survey. *Artificial Intelligence Review*, 50(1):49–73, 2018.

Albert Xu, Eshaan Pathak, Eric Wallace, Suchin Gururangan, Maarten Sap, and Dan Klein. Detoxifying language models risks marginalizing minority voices. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2390–2397, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.190. URL `https://aclanthology.org/2021.naacl-main.190`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL `https://aclanthology.org/P19-1472`.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Peng Zhang, Yuxiao Dong, and Jie Tang. Glm-130b: An open bilingual pre-trained model. *arXiV*, 10 2022. URL `http://arxiv.org/abs/2210.02414`.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiV*, abs/1810.12885, 2018.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models. *arXiV*, 5 2022a. URL `http://arxiv.org/abs/2205.01068`.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.30. URL `https://aclanthology.org/2020.acl-demos.30`.

Zhen Zhang, Shuai Zheng, Yida Wang, Justin Chiu, George Karypis, Trishul Chilimbi, Mu Li, and Xin Jin. Mics: Near-linear scaling for training gigantic model on public cloud, 2022b. URL `https://arxiv.org/abs/2205.00119`.

Appendix A. Architecture

A.0 Notation

Styling. Unstyled variables denote scalars, bold lower-case variables represent [column] vectors, and bold capitalized variables represent matrices. For instance, $h_{i,j}$ could be an element in the vector $\boldsymbol{h_j}$, which could in turn be the j-th column of matrix $\boldsymbol{H}$.

Named functions are typed in non-italicized regular typeface, such as $\mathrm{softmax}(\cdot)$ and $\mathrm{FFN}(\cdot)$.

Red color is used to denote trainable parameters, or functions that are parametrized by trainable parameters, such as $\boldsymbol{W}$ or $\mathrm{FFN}(\cdot)$.

Sequences. A sequence $(x_1, \ldots, x_n)$ of n elements is denoted by $\{x_i\}_{i=1}^n$. We treat a sequence of (column) vectors as a matrix, i.e. $\boldsymbol{X} = \{\boldsymbol{x_i}\}_{i=1}^n \in \mathbb{R}^{m \times n}$, where each $\boldsymbol{x_i} \in \mathbb{R}^m$.

Operators.

- $f : \mathbb{R}^n \to \mathbb{R}^n$: A function on vectors, that is, $\boldsymbol{y} = f(\boldsymbol{x})$ where $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are n-dimensional real valued vectors. Whenever such a function is applied to a matrix, it is applied column-wise: $f(\boldsymbol{X}) = \{f(\boldsymbol{x_j})\}_{j=1}^m$, $\boldsymbol{X} \in \mathbb{R}^{n \times m}$.
- $\boldsymbol{A} \odot \boldsymbol{B}$: Element-wise (or Hadamard) product of matrices or vectors $\boldsymbol{A}$ and $\boldsymbol{B}$ (of the same shape).
- $\mathbb{1}(P)$: Indicator function that returns 1 if the predicate P is true and 0 otherwise.
- $[n]$: For integer n, the set of all positive integers up to (including) n, i.e. $\{1, \ldots, n\}$.
- $\boldsymbol{A} + \boldsymbol{b}$: Adding a vector to a matrix is defined as repeated addition to each column. That is, $\boldsymbol{A} + \boldsymbol{b} = \{\boldsymbol{a_i} + \boldsymbol{b}\}_{i=1}^n$.
- Softmax: $\mathrm{softmax}(\boldsymbol{x}) = \dfrac{\exp(\boldsymbol{x})}{\sum_i^n \exp(x_i)}$ where $\exp(\cdot)$ is applied element-wise to a vector.
- Dropout: $\mathrm{drop}^p(\boldsymbol{x}) = \dfrac{1}{1-p} \cdot \boldsymbol{m} \odot \boldsymbol{x}$ where, $\boldsymbol{m} = [m_i]_{i=1}^{n}{}^\top$, and $m_i \sim \mathrm{Bernoulli}(1-p)$. Random variables m_i are drawn independently for each presentation of an example.

A.1 Full Architecture

Embedding. Let $(x_1, \ldots, x_t) = \{x_t\}_{t=1}^T \in \mathcal{V}^T$ denote an input sequence of length T, where each element x_t denotes an integer identifier of a token from the vocabulary $\mathcal{V} = [|\mathcal{V}|]$.

Initial input representations $\boldsymbol{H^0} = \{\boldsymbol{h_t^0}\}_{t=1}^T$ are obtained by

$$\boldsymbol{\bar{h}_t^0} = \boldsymbol{W}^{em} \boldsymbol{e_{x_t}} \qquad \forall t \qquad (1)$$

$$\boldsymbol{h_t^0} = \mathrm{LN}^{em}(\boldsymbol{\bar{h}_t^0}) \qquad \forall t \qquad (2)$$

where $\boldsymbol{W}^{em} \in \mathbb{R}^{D \times |\mathcal{V}|}$ is the token embedding matrix, $\boldsymbol{e_{x_t}} \in \mathbb{R}^{|\mathcal{V}|}$ is the x_t-th standard basis vector, and LN^{em} is the *em*bedding LayerNorm function, to be defined in the following subsections.

Observe that no positional embedding is applied here due to how ALiBi works.

Layers. Layer representations $\boldsymbol{H^\ell} \in \mathbb{R}^{D\times T}$ for each layer $\ell = 1, \ldots, L$ can be sequentially defined as follows (this computation is sometimes referred to as a "block"):

$$\bar{\boldsymbol{H}}^{\boldsymbol{\ell}} = \boldsymbol{H^{\ell-1}} + \mathrm{SA}_\ell(\mathrm{LN}_\ell^{in}(\boldsymbol{H^{\ell-1}})) \qquad \forall \ell \tag{3}$$

$$\boldsymbol{H^\ell} = \bar{\boldsymbol{H}}^{\boldsymbol{\ell}} + \mathrm{FFN}_\ell(\mathrm{LN}_\ell^{at}(\bar{\boldsymbol{H}}^{\boldsymbol{\ell}})) \qquad \forall \ell \tag{4}$$

where SA_ℓ, FFN_ℓ, and $\mathrm{LN}_\ell^\bullet$ denote SelfAttention, FeedForwardNetwork, and LayerNorm functions at layer ℓ, respectively, as defined in the following subsections. The red color indicates that the functions depend on trainable parameters. $\mathrm{LN}_\ell^\bullet$ is further parametrized by an indication of what the function is applied to, such as LN_ℓ^{in} when applied to the block *in*put and LN_ℓ^{at} when applied to the *at*tention output. We designate these separately since they use different (i.e. untied) trainable parameters.

Logits. Given the final layer representation $\boldsymbol{H^L}$, logits $\boldsymbol{Y} \in \mathbb{R}^{|\mathcal{V}|\times T}$ are obtained as:

$$\boldsymbol{Y} = \boldsymbol{W^{em}}^\top \mathrm{LN}^f(\boldsymbol{H^L}) \tag{5}$$

where $\boldsymbol{W^{em}} \in \mathbb{R}^{D\times|\mathcal{V}|}$ is the same embedding matrix we used in the **embedding** part and LN^f is the *f*inal LayerNorm application. We follow the PaLM approach in omitting a bias term.

The token distribution for position $j+1$, conditioned on the prefix $(x_1, \ldots, x_j)$, is given by

$$\mathbb{P}(x_{j+1} = w | \{x_t\}_{t=1}^j) = \mathrm{softmax}(\boldsymbol{y_j})_w \tag{6}$$

where $\boldsymbol{y_j}$ is the j'th column of $\boldsymbol{Y}$.

A.2 SelfAttention with ALiBi (SA)

SelfAttention with ALiBi at layer ℓ, $\mathrm{SA}_\ell : \mathbb{R}^{D\times T} \to \mathbb{R}^{D\times T}$ is defined as follows.

Let $n \in \{1, \ldots, N\}$ denote an attention head where N is the total number of heads. Let D^n denote the dimensionality of each head. Let $\boldsymbol{A^n}, \boldsymbol{M} \in \mathbb{R}^{T\times T}$ denote the ALiBi matrix and the attention mask, respectively, which will be defined later.

Then, $\boldsymbol{Y} = \mathrm{SA}_\ell(\boldsymbol{X})$ such that:

$$\boldsymbol{Q^n} = \boldsymbol{W_\ell^{n,q}} \boldsymbol{X} + \boldsymbol{b_\ell^{n,q}} \qquad \forall n \tag{7}$$

$$\boldsymbol{K^n} = \boldsymbol{W_\ell^{n,k}} \boldsymbol{X} + \boldsymbol{b_\ell^{n,k}} \qquad \forall n \tag{8}$$

$$\boldsymbol{V^n} = \boldsymbol{W_\ell^{n,v}} \boldsymbol{X} + \boldsymbol{b_\ell^{n,v}} \qquad \forall n \tag{9}$$

$$\bar{\boldsymbol{S}}^{\boldsymbol{n}} = \boldsymbol{A^n} + \frac{\boldsymbol{K^n}^\top \boldsymbol{Q^n}}{\sqrt{D^n}} \qquad \in \mathbb{R}^{T\times T} \qquad \forall n \tag{10}$$

$$\boldsymbol{S^n} = \mathrm{drop}^{p_{at}}(\mathrm{softmax}(\bar{\boldsymbol{S}}^{\boldsymbol{n}} \odot \boldsymbol{M})) \qquad \in \mathbb{R}^{T\times T} \qquad \forall n \tag{11}$$

$$\bar{\boldsymbol{Y}}^{\boldsymbol{n}} = \boldsymbol{V^n} \boldsymbol{S^n} \qquad \in \mathbb{R}^{D^n\times T} \qquad \forall n \tag{12}$$

$$\boldsymbol{Y} = \mathrm{drop}^{p_h}((\sum_{n=1}^N \boldsymbol{U_\ell^n} \bar{\boldsymbol{Y}}^{\boldsymbol{n}}) + \boldsymbol{c_\ell}) \qquad \in \mathbb{R}^{D\times T} \tag{13}$$

where $\boldsymbol{W}_\ell^{n,q}, \boldsymbol{W}_\ell^{n,k}, \boldsymbol{W}_\ell^{n,v} \in \mathbb{R}^{D^n \times D}$, $\boldsymbol{U}_\ell^{n} \in \mathbb{R}^{D \times D^n}$, $\forall n$ are the trainable weight parameters, $\boldsymbol{b}_\ell^{n,q}, \boldsymbol{b}_\ell^{n,k}, \boldsymbol{b}_\ell^{n,v} \in \mathbb{R}^{D^n}$, $\forall n$, $\boldsymbol{c}_\ell \in \mathbb{R}^D$, are the trainable bias parameters, and $p_{at}, p_h \in [0,1)$ are the *at*tention and *h*idden unit dropout probabilities.

The ALiBi matrix $\boldsymbol{A^n} = [a^n_{i,j}]_{i,j} \in \mathbb{R}^{T \times T}$ is constructed as:

$$\tilde{N} = 2^{\lfloor \log_2(N) \rfloor} \tag{14}$$

$$\tilde{n} = 1 + ((n-1) \bmod \tilde{N}) - 0.5 \left\lfloor \frac{n-1}{\tilde{N}} \right\rfloor \tag{15}$$

$$a^n_{i,j} = 2^{-\frac{8}{N}\tilde{n}} \cdot (i-j) \cdot \mathbb{1}(i<j) \qquad \forall i,j \in [T], n \in [N] \tag{16}$$

and the attention mask $\boldsymbol{M} = [m^n_{i,j}]_{i,j} \in \mathbb{R}^{T \times T}$ is constructed as:

$$m_{i,j} = \mathbb{1}(i \le j) - \infty \cdot \mathbb{1}(i > j) \qquad \forall i,j \in [T] \tag{17}$$

where we follow the convention that $\infty \cdot 0 = 0$.

A.3 LayerNorm (LN)

LayerNorm, $\mathrm{LN}^\theta : \mathbb{R}^D \to \mathbb{R}^D$, is defined as follows:

$$\boldsymbol{y} = \mathrm{LN}^\theta(\boldsymbol{x}) = \frac{\boldsymbol{x} - \mu(\boldsymbol{x})}{\sqrt{\sigma^2(\boldsymbol{x}) + \epsilon}} \odot \boldsymbol{\gamma^\theta} + \boldsymbol{\beta^\theta} \tag{18}$$

where

$$\mu(\boldsymbol{x}) = \frac{1}{D}\sum_i x_i \qquad \in \mathbb{R} \tag{19}$$

$$\sigma^2(\boldsymbol{x}) = \frac{1}{D}\sum_i (x_i - \mu(\boldsymbol{x}))^2 \qquad \in \mathbb{R} \tag{20}$$

and, $\boldsymbol{\gamma^\theta}, \boldsymbol{\beta^\theta} \in \mathbb{R}^D$ are the trainable gain and bias parameters, and $\epsilon \in \mathbb{R}$ is a small constant.

θ is used as the parametrization variable to emphasize LN^{em}, LN^{f}, and LN^{in}_ℓ, LN^{at}_ℓ, $\forall \ell$ have different (untied) $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ parameters.

A.4 FeedForwardNetwork (FFN)

Feedforward network component $\mathrm{FFN}_\ell : \mathbb{R}^D \to \mathbb{R}^D$ is defined as a simple multilayer perceptron. $\boldsymbol{y} = \mathrm{FFN}_\ell(\boldsymbol{x})$ such that:

$$\boldsymbol{h} = \mathrm{gelu}(\boldsymbol{W}^f_\ell \boldsymbol{x} + \boldsymbol{b}^f_\ell) \qquad \in \mathbb{R}^{D'} \tag{21}$$

$$\boldsymbol{y} = \mathrm{drop}^{p_f}(\boldsymbol{U}^f_\ell \boldsymbol{h} + \boldsymbol{c}^f_\ell) \qquad \in \mathbb{R}^{D} \tag{22}$$

where $\mathrm{gelu}(x) = 0.5 \cdot x \cdot (1 + \tanh(0.79788456 \cdot x \cdot (1 + 0.044715 \cdot x^2)))$ is applied element-wise, $\boldsymbol{W}^f_\ell \in \mathbb{R}^{D' \times D}$, $\boldsymbol{U}^f_\ell \in \mathbb{R}^{D \times D'}$ are the trainable weight parameters, $\boldsymbol{b}^f_\ell \in \mathbb{R}^{D'}$, $\boldsymbol{c}^f_\ell \in \mathbb{R}^{D}$ are the trainable bias parameters, and $p_f \in [0,1)$ denotes the dropout probability at this component.

A.5 List of All Trainable Parameters

List of shape hyperparameters and their values are as follows:

- $L = 70$ (number of layers)
- $N = 40$ (number of heads)
- $|\mathcal{V}| = 131072$ (vocabulary size)
- $D = 7,680$ (hidden dimension)
- $D^n = 192$, $\forall n \in [N]$ (hidden dimension of each head)
- $D' = 4D = 30,720$ (hidden dimension of FFN)

Initialization hyperparameters are as follows:

- $z = 0.006588 \approx 1/\sqrt{3D}$ is the default range (standard deviation).
- $z' = z \cdot (1/\sqrt{2L})$ is the rescaled range for the second layer in FFN and the final linear map in SA.

List of all parameters with their sizes and (element-wise) initialization:

Range	Group	Param	Shape	Size	Total size	Init
		$\boldsymbol{W}^{em}$	$D \times \|\mathcal{V}\|$	1,006,632,960	1,006,632,960	$\sim \mathcal{N}(0, z)$
	LN^{em}	$\boldsymbol{\gamma}^{em}$	D	7,680	7,680	$= 1$
		$\boldsymbol{\beta}^{em}$	D	7,680	7,680	$= 0$
$\ell \in [70]$	LN$^{in}_\ell$	$\boldsymbol{\gamma}^{in}_\ell$	D	7,680	537,600	$= 1$
		$\boldsymbol{\beta}^{in}_\ell$	D	7,680	537,600	$= 0$
$\ell \in [70]$,	SA$_\ell$	$\boldsymbol{W}^{n,q}_\ell$	$D^n \times D$	1,474,560	4,128,768,000	$\sim \mathcal{N}(0, z)$
$n \in [40]$		$\boldsymbol{W}^{n,k}_\ell$	$D^n \times D$	1,474,560	4,128,768,000	$\sim \mathcal{N}(0, z)$
		$\boldsymbol{W}^{n,v}_\ell$	$D^n \times D$	1,474,560	4,128,768,000	$\sim \mathcal{N}(0, z)$
		$\boldsymbol{U}^{n}_\ell$	$D \times D^n$	1,474,560	4,128,768,000	$\sim \mathcal{N}(0, z')$
		$\boldsymbol{b}^{n,q}_\ell$	D^n	192	537,600	$= 0$
		$\boldsymbol{b}^{n,k}_\ell$	D^n	192	537,600	$= 0$
		$\boldsymbol{b}^{n,v}_\ell$	D^n	192	537,600	$= 0$
$\ell \in [70]$	SA$_\ell$	$\boldsymbol{c}_\ell$	D	7,680	537,600	$= 0$
$\ell \in [70]$		$\boldsymbol{\gamma}^{at}_\ell$	D	7,680	537,600	$= 1$
		$\boldsymbol{\beta}^{at}_\ell$	D	7,680	537,600	$= 0$
$\ell \in [70]$	FFN$_\ell$	$\boldsymbol{W}^{f}_\ell$	$D' \times D$	235,929,600	16,515,072,000	$\sim \mathcal{N}(0, z)$
		$\boldsymbol{U}^{f}_\ell$	$D \times D'$	235,929,600	16,515,072,000	$\sim \mathcal{N}(0, z')$
		$\boldsymbol{b}^{f}_\ell$	D'	30,720	2,150,400	$= 0$
		$\boldsymbol{c}^{f}_\ell$	D	7,680	537,600	$= 0$
	LN^{f}	$\boldsymbol{\gamma}^{f}$	D	7,680	7,680	$= 1$
		$\boldsymbol{\beta}^{f}$	D	7,680	7,680	$= 0$
					50,558,868,480	

Tag	Question
price or not	Does the news headline talk about price(?)
price up	Does the news headline talk about price going up(?)
price stable	Does the news headline talk about price staying constant(?)
price down	Does the news headline talk about price going down(?)
past price	Does the news headline talk about price in the past(?)
future price	Does the news headline talk about price in the future(?)
past general	Does the news headline talk about a general event (apart from prices) in the past(?)
future general	Does the news headline talk about a general event (apart from prices) in the future(?)
asset comparison	Does the news headline compare gold with any other asset(?)

Table 18: Official documentation of each tag (Sinha and Khandait, 2020).

Appendix B. Details on external financial tasks

FPB (Malo et al., 2014): The Financial Phrasebank Dataset includes a sentiment classification task on ~5, 000 sentences in the English language taken from financial news about companies listed on OMX Helsinki. Sentiment annotations of positive, negative, neutral are adjudicated from the perspective of an investor: any news that could benefit/hurt an investor is considered positive/negative and neutral otherwise. Each sentence is annotated by 5 to 8 annotators who have sufficient knowledge of finance, whereas the source sentences were written by financial journalists. For example, news about shrinking revenue would be labeled negative and company growth as positive. While there are different configurations of this dataset with each configuration denoting the percentage agreement between annotators ($\geq$50%, $\geq$66%, $\geq$75%, 100%), we choose to use the configuration with $\geq$50%. Since an official train-test split is not available, we create our own random split. Our training split contains 3,876 sentences with 1,086 positive, 488 negative, and 2,302 neutral sentences and our test set contains 970 sentences with 277 positive, 116 negative, and 577 neutral sentences. We choose 5 shots and report F1 score weighted by support.

FiQA SA (Maia et al., 2018): The second sentiment analysis task is to predict the aspect-specific sentiment in English financial news and microblog headlines, which were published as a part of the 2018 challenge on financial question answering and opinion mining. In the original task, sentiment is annotated on a continuous scale of $[-1, +1]$; the details on the annotation task are not readily available. To make this regression dataset amenable for few-shot LLM setup, we convert it into a classification task: Negative ($-1 \leq x < -0.1$), neutral ($-0.1 \leq x < +0.1$), and positive ($+0.1 \leq x \leq +1$), where x is the original sentiment score. We selected this discretization based on a manual examination of the dataset. Like with FPB, we create our own random split combining both microblogs and news. After discretization, our training set contains 938 sentences with 576 positive, 287 negative, and 75 neutral sentences and our test set contains 235 sentences with 141 positive, 76 negative, and 18 neutral sentences. We select 5 shots and report weighted F1.

Headline (Sinha and Khandait, 2020): This is a binary classification task of whether a news headline in the gold commodity domain includes certain information. This human-annotated dataset consists of 11,412 English news headlines from 2000 to 2019 about "gold" scraped from providers such as Reuters, The Hindu, The Economic Times, Bloomberg, and from aggregator sites such as Kitco, and MetalsDaily. Each news article carries a subset of the following tags: “price or not”, “price up”, “price down”, “price stable”, “past price”, “future price”, “past general”, “future general”, “asset comparison”. The dataset is created using annotator consensus and Cohen’s Kappa for each of the categories is $\geq$0.85 indicating a high-quality dataset. Like with FPB, we create our own random split. Our training set contains 9,129 sentences with 7,780, 3,785, 3,392, 414, 7,482, 299, 1,285, 67, 1,696 examples of “price or not”, “price up”, “price down”, “price stable”, “past price”, “future price”, “past general”, “future general”, “asset comparison” classes, respectively. Similarly, the test set contains 2283 sentences with 1,955, 962, 838, 109, 1,873, 82, 313, 15, 454 examples of the same classes. We verbalize each tag into a question using the official documentation on each tag as shown in Table 18. We used 5 shots, and report the average weighted F1 score across all categories.

NER (Salinas Alvarado et al., 2015): This is a named entity recognition task on financial data gathered for credit risk assessment. The dataset consists of 8 documents with ~55,000 words of financial agreements filed with the SEC. The annotated entity types follow the standard CoNLL format (Tjong Kim Sang and De Meulder, 2003) and are annotated with PER, LOC, ORG, and MISC. We use Fin-5 as the training data for context sampling and test on the Fin-3 split. As MISC cannot be defined on its own but “names (that) are not already in the other categories” (Tjong Kim Sang and De Meulder, 2003), we drop all entities with type MISC. Additionally, as it is nontrivial to learn to predict empty output in the few-shot set-up, we drop sentences that do not contain any entity. After preprocessing, our training set contains 504 sentences with 168 PER, 745 LOC, and 241 ORG, and our test set consists of 98 sentences with 39 PER, 216 LOC, and 56 ORG. We found that all the models required more shots to perform well. Hence, we selected 20 shots and report the entity-level F1 score.

ConvFinQA (Chen et al., 2022): Given an input that includes text and at least one table with financial data, the task is to answer conversational questions that require numerical reasoning over the input. The source data is earning reports of S&P 500 companies and consists of 3,892 conversations consisting 14,115 questions. This task requires numerical reasoning, an understanding of structured data and financial concepts, and a model needs to relate follow-up questions to the dialog turns. To solve this task, we use “1 shot” where an entire gold conversation and its context is input to the models. In addition, as each “turn” of the conversation concludes, the “turn” along with the “gold” answer for that turn is appended as context for future turns. Tables are linearized in the context (as suggested by the authors) as Markdown tables, and we replace an empty entry with ”-”. The reported score is the exact match accuracy of the direct answer produced by a model. As test set labels are not publicly available, we report results on the dev set instead. Our training set contains 11,104 conversations and 45,888 questions and our test set contains 1,490 conversations and 5,932 questions.

Appendix C. Training Chronicles

C.0 Still 🌱

Our first training run was called v0. In this run, we experimented with curriculum learning. Data that the model would see in the future would likely be similar to the newer data in our training corpus, so we wanted the model to do better on those future documents. Additionally, since there are facts that change over time, newer information should ideally override the old. Therefore, we temporally ordered the training data by month in FinPile.

Figure 7 shows the learning curve for run v0. We observed a large gap between training and validation losses, which was expected: early stages of training would observe the oldest data (starting from 2007) whereas our validation set was strictly from the future (i.e., 2022). However, one week into training we found the model stuck on both training and validation loss, as seen by the very limited validation progress between steps 15k-20k and almost no progress after step 20k. There was the possibility that the training loss and the divergence of training and validation loss would both resolve themselves as the training data became more and more similar to the validation data as the curriculum progressed. However, we deemed this to be too risky to catch any other potential problems with the training that might require early intervention, since it would mean training for many steps without any diagnostic signal. We thus decided to abandon curriculum learning altogether.

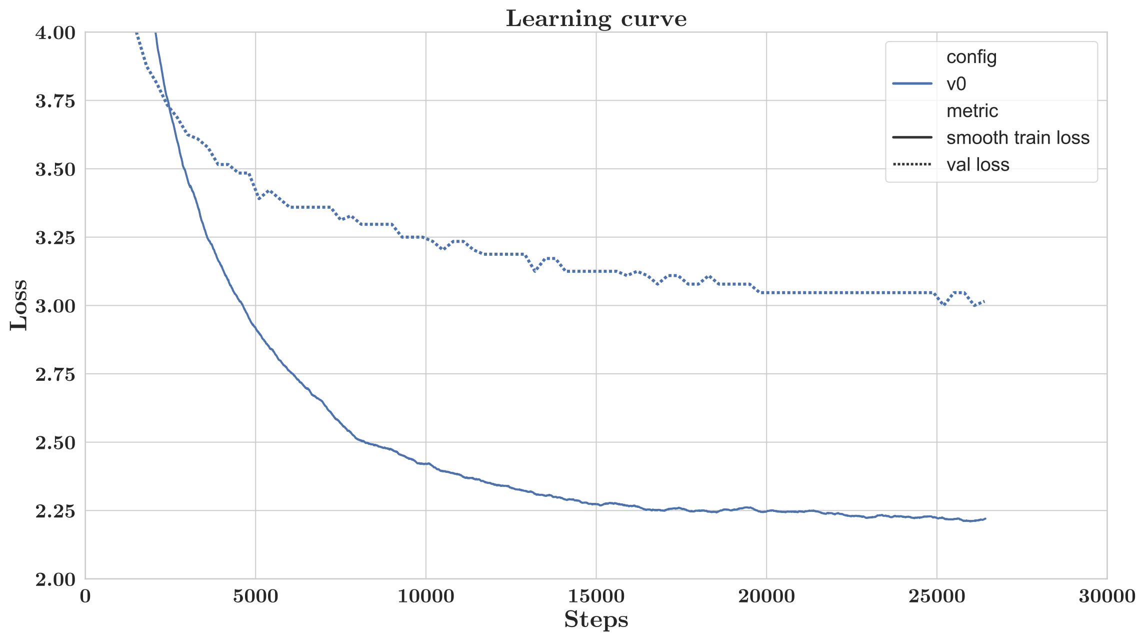

Figure 7: Learning curve of our first training attempt named v0. Observe the large gap between training and validation losses, as well as the flatness of both curves after step 20k. The final 6k steps lasted about ~2.3 days.

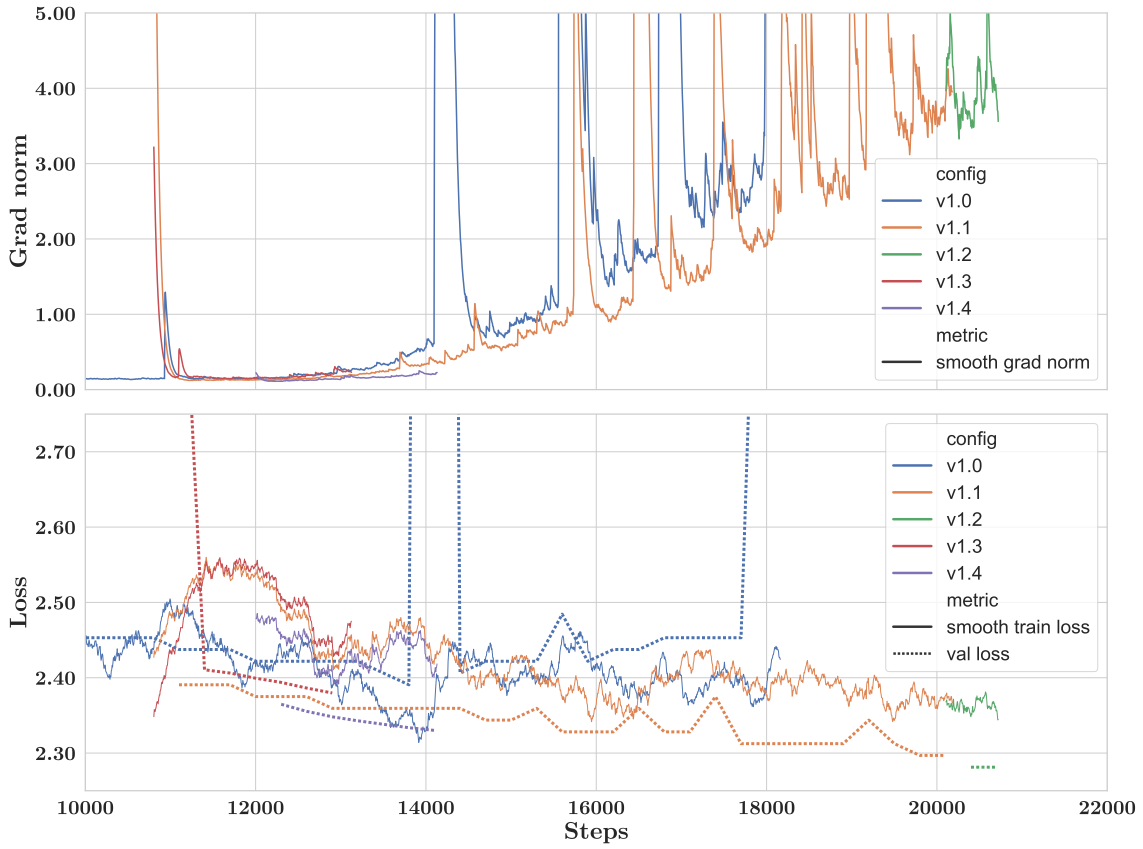

Figure 8: Gradient norms (top) and train & validation loss (bottom) of v1.x runs.

We removed curriculum learning by shuffling all of our training data uniformly on the shard level.[3] We then started a new run (v1.0), which led to much faster improvements in the validation loss. We were unable to ascertain if curriculum learning had a negative impact on training or if the loss plateaued due to other factors, for example, the other discovered issue in v1.x.

C.1 Elbow 💪

During our new run without curriculum learning (v1.0), we observed that the gradient norm showed a steady increase after about 12k steps (~4.5 days of training), with occasional spikes (see Figure 8). This was accompanied by sudden jumps in the validation loss, possibly indicating that the model might be becoming sensitive to small changes in its weights. Training loss seemed to have been plateauing again, as well.

We believed that the gradient norm increases were the cause of the validation loss problems (notice the alignment between sudden validation loss jumps with some of the

3. Instead of loading one shard of data at a time, we load multiple random shards (without replacement) at the same time and shuffle them on the fly.

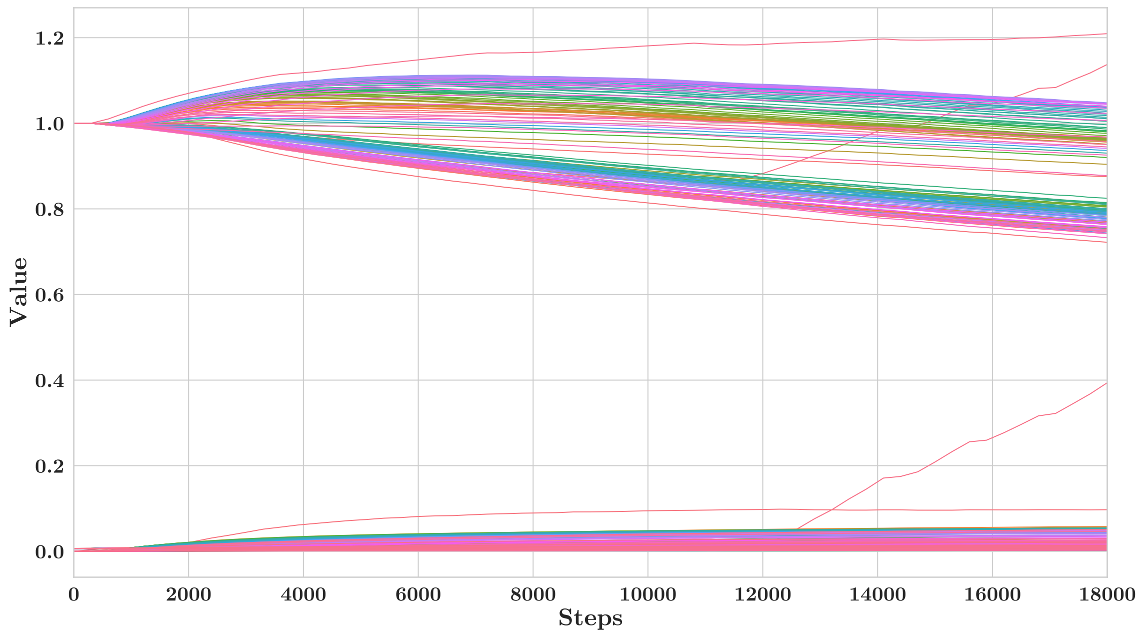

Figure 9: Rescaled norms for each component in v1.0 run. Input LayerNorm at Layer 1 stood out.

sudden gradient norm jumps for v1.0, in Figure 8). We made several attempts across several model runs to fix the gradient norm increases:

Run	**Changes from v1.0 run**
Shared Change	- Fully shuffle any data not seen by the model checkpoint that we chose to start (or restart) from instead of shard-level shuffling
v1.1	- Start from step 10.8k of v1.0, prior to any gradient increase - Reduce max learning rate from 1e-4 to 8e-5
v1.2	- Continue from step 20.1k of v1.1 (most recent checkpoint) - Reduce max learning rate from 1e-4 to 6e-5 - Reduce gradient clip from 1.0 to 0.3
v1.3	- Start from step 10.8k of v1.0, prior to any gradient increase - Use FP32 precision for LM-head computation (prior to softmax)
v1.4	- Start from step 12.0k of v1.3 - Reduce max learning rate from 1e-4 to 6e-5 - Reduce gradient clip from 1.0 to 0.3 - Use FP32 precision for LM-head computation (prior to softmax)

All of these attempted fixes were made after we observed a trend of increasing gradient norms similar to the original run (v1.0), or some early signs of a similar path that we hypothesized would eventually grow more. Since we didn't want to waste training time, we

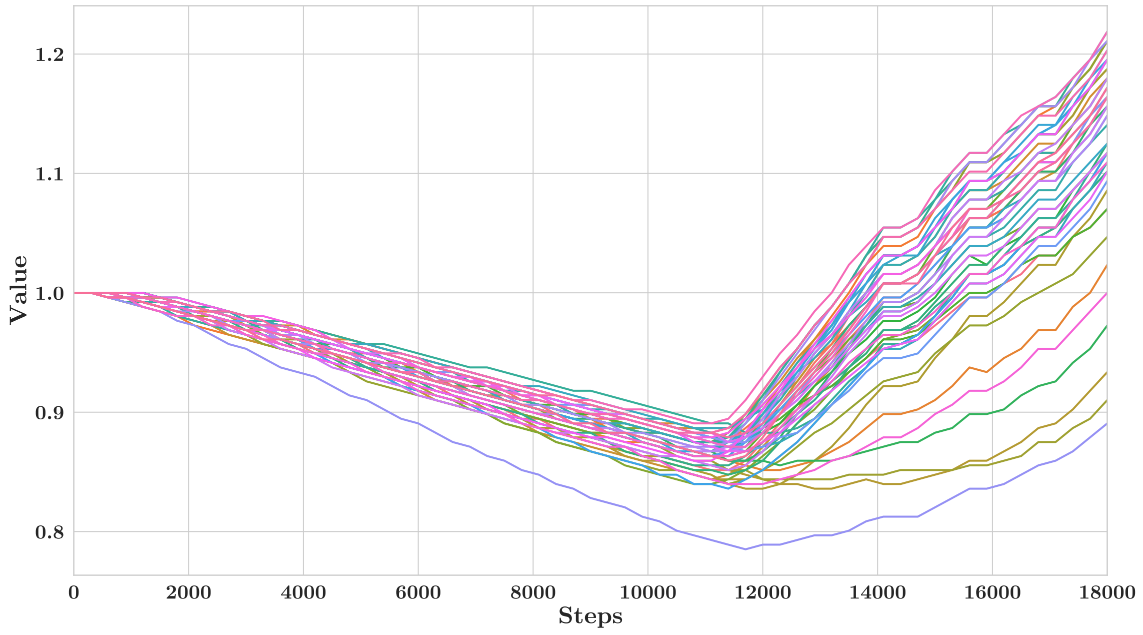

Figure 10: Values for Input LayerNorm at Layer 1 in v1.0 run.

did our best to make decisions early instead of allowing the model to continue down a bad training path.

We investigated the norms of the weights themselves to see if any peculiar trends were aligning with the gradient growth. In particular, we were curious to see if there were particular layers or components that were responsible for the large gradient norms.

Figure 9 plots L2 norms for each component, averaged by the square root of the number of elements (layer norm multipliers start from 1 and all the others start close to zero due to initialization). We observed that all components follow a similar benign trend except one: Input LayerNorm at layer 1 (i.e. LN_1^{in}), which suddenly elbows and starts increasing roughly linearly after step ~12k. This also aligns with the initial growth of the gradient norms.

To take a closer look, we inspected individual values of the multiplier weights γ_1^{in} (there are 60 such values in a single model shard out of 128) in Figure 10. We observed all values contributing to the same trend of shrinking until steps 11-12k and then shifting to move upward instead.

During this investigation, we discovered another bug: Weight decay was applied to all the non-bias parameters, as opposed to skipping the LayerNorm multiplier weight γ since they are initialized at 1. To the best of our knowledge, this practice has been inherited from the BERT implementation[4].

4. `https://github.com/google-research/bert/blob/eedf5716ce1268e56f0a50264a88cafad334ac61/optimization.py#L59-L65`

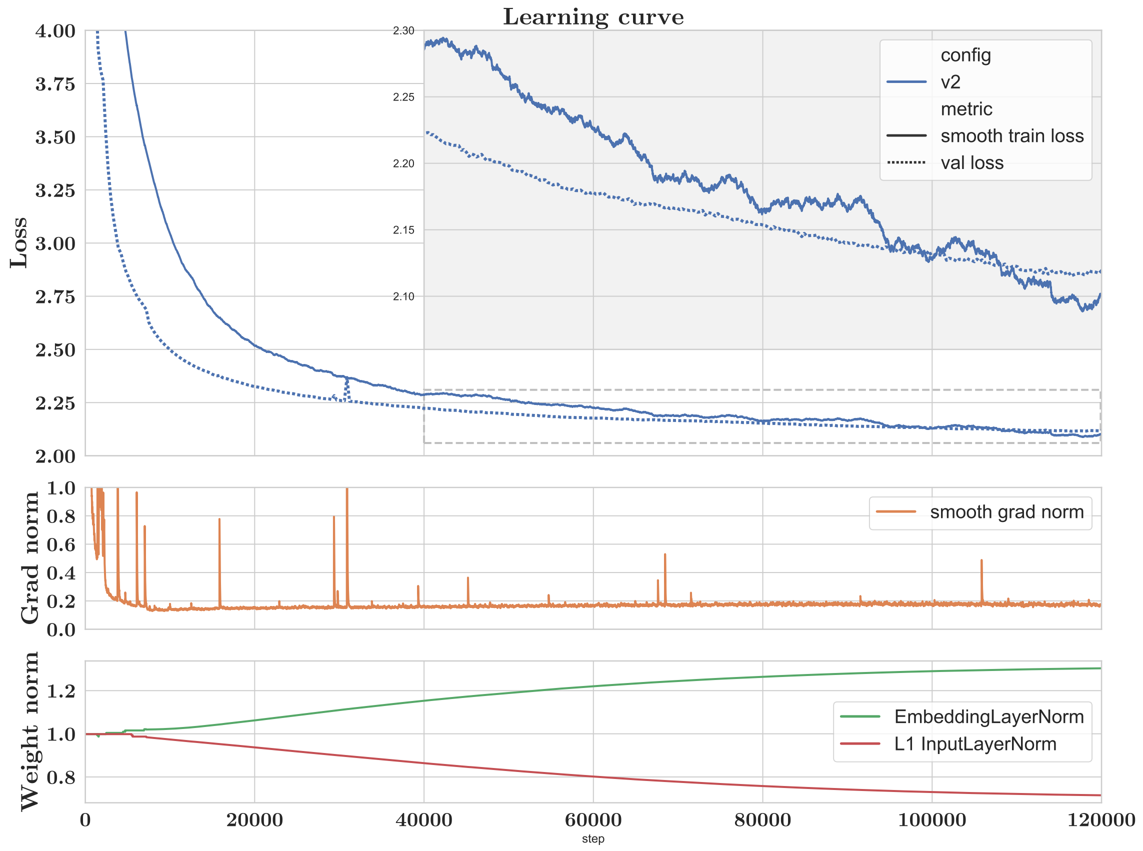

Figure 11: Loss, gradient norm and weight norms for listed components for the v2.0 run.

This makes the elbow artifact shown in Figure 10 even more confusing: An additional push of weights towards 0 would trivially explain a downward trend but not a sudden shift to growth.

After four failed attempts to fix the run, we considered the possibility of this run being unsalvageable and contemplated starting from scratch to apply a more conservative hyperparameter setting from the beginning. These would include things that we have tried in our attempts such as shrinking the learning rate or gradient clipping. Additionally, because the pathological trend change is isolated to LN_1^{in}, which is topologically very close to the removed LayerNorm at the embedding layer (LN^{em}) we decided to add back LN^{em} as an additional precaution, despite most other LLMs not having this component.

C.2 Slide 🛝

After numerous attempts to fix the elbow issue, we wanted to be as conservative as possible for the hyperparameter choices when starting from scratch to keep the learning dynamics as stable as possible. We started the next run (v2.0) with the following hyperparameters and changes:

- Use FP32 precision in LM-head (softmax in Equation (6))

- Use max learning rate of 6e-5 instead of 1e-4
- Use a gradient clipping value of 0.3 instead of 1.0
- Fully shuffle data
- Use a different seed to ensure different initialization and data order
- Reintroduce LayerNorm at embedding layer (LN^{em})
- Use a longer learning rate warm-up period of 1800 steps
- Remove incorrect use of weight decay on LayerNorm multipliers ($\gamma_{\cdot}^{\cdot}$)
- Use Megatron initialization rescaling (see use of z' in Appendix A.5)
- Apply `query_key_layer_scaling` (Shoeybi et al., 2019)
- Apply a batch size warm-up: Use a batch size of 1024 for 7200 iterations, then increase to 2048

In addition to hyperparameter changes, we also performed additional monitoring to catch issues earlier. Because we observed the pathological behavior at the first LayerNorm component, we started monitoring the norms of the weights γ^{em} and γ_1^{in} (scaled by $1/\sqrt{D}$).

With the aforementioned conservative choice of hyperparameters during v2.0, we observed very smooth and (thankfully!) uneventful training for approximately 42 days (~115,500 iterations). We saw few surprises both in terms of training and validation performance curves (see Figure 11), as well as the norms of the gradients. The only intervention needed during this period was to restart the job after 28 days, due to the underlying platform having a hard limit on the duration of the job.

During this period, we observed a smoothly decreasing validation loss (except a few jumps earlier on) until it started to flatten around 2.116 (y-axis) at the end. Running training loss has a similar trend of overall decrease with the typical jitter and random occasional increments.

We also observed that weight norms for LayerNorm components of the initial layer were smooth and stable without any immediate or long-term trend changes (see Figure 11). This presents some evidence that we were not suffering from the pathological behavior we observed in v1.x in regards to LayerNorm parameters γ^{em}.

Overall, while this set of changes led to a smooth training run in v2.0, we cannot conclude which of these changes was decisive in leading to a successful training run. We defer such investigation to future work.

C.3 Suspense 🚡

About 48 days into training v2.0, we noticed that the validation loss had not improved in a week (from iteration 115,500 to 133,200, see Figure 12, v2.0 curves). During the same period, we also noticed training loss flattening around 2.10 (with the usual jitter). We suspected that the model was no longer learning properly, and decided to intervene.

We considered two options: 1) changing the max learning rate, 2) rolling back to an earlier checkpoint and re-shuffling the remainder of the data to pick up a different optimization path.

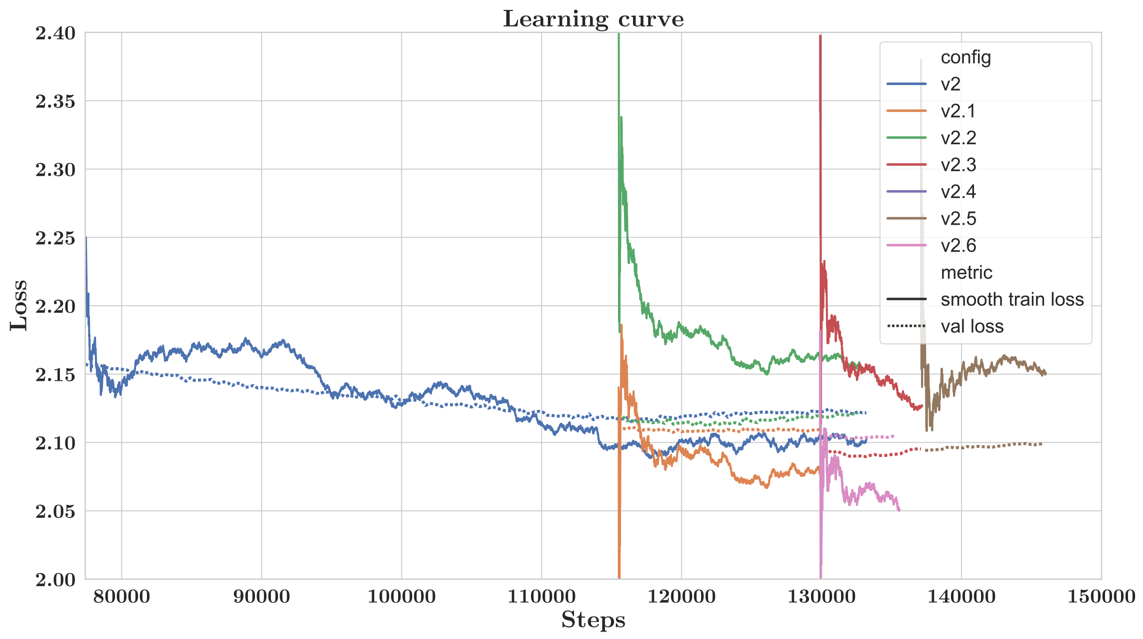

Figure 12: Loss values for various runs at the end of v2. Note v2.4 overlaps with v2.6.

We had two proposed ways in which to change the learning rate. An argument for *increasing* the learning rate was the possibility that we were stuck in a local optimum. Allowing the optimizer to make bigger jumps would allow the model to escape the optimum and continue learning. On the other hand, the argument for *decreasing* the learning rate was based on Zhang et al. (2022a) in which they had observed improvements after shrinking the learning rate after getting stuck. Furthermore, we had spent more steps in the high-learning-rate region of the overall learning rate schedule since, by following the Chinchilla scaling law, we had more total steps compared to models like BLOOM or GPT-3.

The other option was to roll back to an earlier checkpoint, re-shuffle the remainder of the data and continue training. Chowdhery et al. (2022) found that when they saw spikes in the validation loss, they "re-started training from a checkpoint roughly 100 steps before the spike started, and skipped roughly 200–500 data batches." This suggests that data ordering mattered, and backing out of a bad data/gradient path may help. That may have been our issue with curriculum learning (v0.x), although it may have been that the issues were not with curriculum learning but with other issues we fixed in v1.0.

In the end, we decided to *shrink* the learning rate, roll back to the start of the increasing validation loss trend 7 days prior, and also re-shuffle the remaining data.

We also became concerned that our choices were based on a single, albeit large, development set. Our validation set only contained data from July 2022 (val$_{\text{future}}$; roughly 105M tokens), whereas the training set ranged from 2007 to June 2022, meaning that the validation set was slightly out of distribution. We had done this to ensure a future-forward evaluation, and to ensure that the training set didn't have leaked validation data. While this matched our goals, it was possible that a single month of data was not properly reflective of the model's abilities, and thus we were making decisions that overfit the validation data. We created a second validation set from the last 105M tokens of the training set for offline evaluation (val$_{past}$). These tokens were from training but would be unobserved until the model finished training. However, since the model training data was fully shuffled, this validation set was not from a held-out time-period.

To assess whether a lack of progress on validation loss translates into a lack of progress on downstream evaluation performance, we used two popular benchmarks: the multiple-choice subset of BBH (bbh$_{sub}$) and all of MMLU. These provided additional assurance that changes in validation loss were tracking actual model improvements. Note that running a checkpoint on these benchmarks is much more time-consuming than computing the validation loss.

	v2	**v2.1**	**v2**	**v2.1**	**v2**	**v2.1**	**v2**	**v2.1**
step	**val$_{future}$**		**val$_{past}$**		**mmlu**		**bbh$_{sub}$**	
99300	8.43		8.60		37.77		43.57	
115500	8.30		8.43		38.79		43.10	
126600	8.34	8.24	8.40	8.32	37.86	38.09	42.37	42.79
133200	8.35		8.38		37.02		42.26	

Table 19: Preliminary evaluation on in-distribution (val$_{past}$, 105M tokens), and out-of-distribution (val$_{future}$; 105M tokens) validation sets (perplexity), and downstream tasks (accuracy). We report perplexity since we compare models with the same tokenization.

Using our two dev sets and downstream evaluations for guidance, we made several attempts to improve run v2.0 and direct the model to continue learning. A summary of our attempts follows:

Run	**Changes from v2.0 run**
Shared Change	- Re-shuffle future data starting from step 115500
v2.1	- Start from v2.0 step 115500 - Reduce max learning rate from 6e-5 to 4e-5
v2.2	- Start from v2.0 step 115500 - Increase dropout from 0.0 to 0.1
v2.3	- Start from v2.1 step 129900 - Reduce max learning rate from 6e-5 to 2e-5 - Increase dropout from 0.0 to 0.1
v2.4	- Start from v2.1 step 129900 - Reduce max learning rate from 6e-5 to 2e-5
v2.5	- Start from v2.3 step 137100 - Reduce max learning rate from 6e-5 to 1e-5 - Increase dropout from 0.0 to 0.1
v2.6	- Start from v2.1 step 129900 - Reduce max learning rate from 6e-5 to 2e-5 - Reduce weight decay from 0.1 to 0.01

After we lowered the learning rate and rolled back the model (v2.1), we observed an initial sudden (and dramatic) improvement; however, validation loss quickly flattened out. Coupled with the mixed results on downstream evaluation, we decided to enable dropout for the first time with a probability of 0.1.

With dropout, as expected, we observed a larger training loss since dropout is applied during the computation of the loss (v2.2 in Figure 12). However, we observed an initially decreasing validation loss. Still, as the run progressed further, validation loss started creeping back up and met the value of the original run (v2.0, blue).

Based on these observations, we decided that further decreasing the learning rate would give us the best chance to continue learning successfully. We subsequently tried various combinations of smaller values of learning rate and adding dropout. Observe that in v2.3 (red) with 2e-5 max learning rate and in v2.5 (brown) with 1e-5 as its continuation, both with a dropout rate of 0.1, shown in Figure 12. In Table 20, we observed v2.3 led to much better perplexity and slightly better downstream performance, and v2.5 continues to improve downstream performance compared to v2.3 in the beginning, while decreasing perplexity slightly. v2.4 (purple) attempted a max learning rate of 2e-5 as well, without dropout, however. The only odd run during this time is v2.6 (pink), in which we experimented with a smaller weight decay of 0.01 (compared to the original 0.1) with a max learning rate of 2e-5 to investigate the possibility of getting stuck in local minima due to too strong of a pull from the weight decay. However, this yields almost the exact same curve as the original 0.1 weight decay (the difference between v2.4 and v2.6 is only the weight decay, and since they yield the same curve v2.6 completely hides v2.4, rendering it invisible in the plot).

In conclusion, all of the runs (summarized in Figure 12) had the same outcome of eventual flattening of the validation loss and sometimes even increasing the loss. We did not observe that any particular change significantly improved the downstream evaluations and validation loss (Table 20).

At this point, we had used 77% of our training data and were nearing the end of the budget we had set aside for training. Combined with all of these observations and initial promising results on the downstream benchmarks, we decided to end training despite not having gone through all of our training data. Another motivating factor was the possibility of using remaining unseen training data for subsequent runs of different styles of training and finetuning.

Based on this experience, we plan to explore different options in future experiments that have shown the potential to lead to more stable training for longer durations, including SwiGLU activations (Shazeer, 2020), RoPE embeddings (Su et al., 2021b), and normalization for queries and keys in the attention layers (Henry et al., 2020).

	v2	**v2.1**	**v2.2**	**v2.3**	**v2.4**	**v2.5**	**v2**	**v2.1**	**v2.2**	**v2.3**	**v2.4**	**v2.5**
step	**val$_{\text{future}}$**						**val$_{\text{past}}$**					
99300	8.43						8.60					
115500	8.30						8.43					
126600	8.34	8.24	8.31				8.40	8.32	8.56			
131700				8.09	8.20					8.22	8.24	
133200	8.35			8.08			8.38			8.22		
137100				8.13						8.30		
139200						8.14						8.32
143400						8.15						8.33
145800						8.16						8.32
step	**mmlu**						**bbh$_{\text{sub}}$**					
99300	37.77						43.57					
115500	38.79						43.10					
126600	37.86	38.09	38.77				42.37	42.79	42.82			
131700				38.76	38.51					43.02	43.49	
133200	37.02			38.90			42.26			43.46		
137100				38.71						44.02		
139200						39.02						44.20
143400						38.98						43.20
145800						38.80						43.37

Table 20: Preliminary evaluation on in-distribution (val$_{\text{past}}$, 105M tokens), and out-of-distribution (val$_{\text{future}}$; 105M tokens) validation sets (perplexity), and downstream tasks (accuracy). We report perplexity here as we are comparing models with the same tokenization.