GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

Nan Du * Yanping Huang * Andrew M. Dai * Simon Tong Dmitry Lepikhin Yuanzhong Xu Maxim Krikun
Yanqi Zhou Adams Wei Yu Orhan Firat Barret Zoph Liam Fedus Maarten Bosma Zongwei Zhou
Tao Wang Yu Emma Wang Kellie Webster Marie Pellat Kevin Robinson Kathy Meier-Hellstern
Toju Duke Lucas Dixon Kun Zhang Quoc V Le Yonghui Wu Zhifeng Chen Claire Cui
Google Inc

Abstract

Scaling language models with more data, compute and parameters has driven significant progress in natural language processing. For example, thanks to scaling, GPT-3 was able to achieve strong results on in-context learning tasks. However, training these large dense models requires significant amounts of computing resources. In this paper, we propose and develop a family of language models named GLaM (Generalist Language Model), which uses a sparsely activated mixture-of-experts architecture to scale the model capacity while also incurring substantially less training cost compared to dense variants. The largest GLaM has 1.2 trillion parameters, which is approximately 7x larger than GPT-3. It consumes only 1/3 of the energy used to train GPT-3 and requires half of the computation flops for inference, while still achieving better overall zero-shot and one-shot performance across 29 NLP tasks.

1. Introduction

Language models have played an important role in the progress of natural language processing (NLP) in the past decade. Variants of language models have been used to produce pretrained word vectors (Mikolov et al., 2013; Pennington et al., 2014), and contextualized word vectors (Peters et al., 2018; Devlin et al., 2019) for many NLP applications. The shift towards scaling with more data and larger models (Shazeer et al., 2017; Huang et al., 2019; Kaplan et al., 2020) has enabled complex natural language tasks to be performed with less labeled data. For example, GPT-3 (Brown et al., 2020) and FLAN (Wei et al., 2021) demonstrated the feasibility of in-context learning for few-shot or even zero-shot generalization, meaning very few labeled examples are needed to achieve good performance on NLP applications. While being effective and performant, scaling further is becoming prohibitively expensive and consumes significant amounts of energy (Patterson et al., 2021).

In this work, we show that a large sparsely activated network can achieve competitive results compared to state-of-the-art dense models on few-shot tasks while being more computationally efficient. We present a family of mixture-of-experts (MoE) language models called GLaM, that strike a balance between dense and conditional computation. The largest GLaM has 1.2 trillion parameters, which is approximately 7x larger than GPT-3. It consumes only 1/3 of the energy used to train GPT-3 and requires half of the computation flops for inference, while still achieving better overall zero-shot and one-shot performance across 29 NLP tasks.

Table 1. Comparison between GPT-3 and GLaM. In a nutshell, GLaM outperforms GPT-3 across 21 natural language understanding (NLU) benchmarks and 8 natural language generative (NLG) benchmarks while using about half the FLOPs per token during inference and consuming about one third the energy for training. The average NLG and NLU scores are defined in section 5.2.
We use GLaM to study the importance of data, scale, and sparsity. Our analysis shows that even for these large models, data quality should not be sacrificed for quantity if the goal is to produce a high-quality language understanding model. On social dimensions, our results are the first, to our knowledge, to close the performance gap between stereotypical and anti-stereotypical examples on the Winogender benchmark, suggesting that large, sparsely activated models may rely less on superficial statistical correlations.

Finally, although MoE models are not yet common in NLP, our work shows that even a basic version of MoE can work extremely well for language understanding at scale. Our results also confirm that sparsity is one of the most promising directions to achieve high-quality NLP models while saving energy costs (Patterson et al., 2021). MoE should therefore be considered as a strong candidate for future scaling.

### 2. Related Work

**Language models:** Neural language models (Mikolov et al., 2010; Sutskever et al., 2011) have been shown to be useful for many natural language processing tasks. Word embedding models and extensions such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and paragraph vectors (Le & Mikolov, 2014) have shown good generalization to many tasks simply by transferring the embeddings.

**Pre-training and Fine-tuning:** The abundance of compute and data enables training increasingly large models via unsupervised pre-training. This is a natural fit for training neural networks as they exhibit remarkable scalability. Work on using recurrent models such as RNNs and LSTMs for language representation (Dai & Le, 2015; Kiros et al., 2015) showed that general language models could be fine-tuned to improve various language understanding tasks. More recently, models that used Transformers (Vaswani et al., 2017) showed that larger models with self-supervision on unlabeled data could yield significant improvements on NLP tasks (Devlin et al., 2019; Yang et al., 2019; Liu et al., 2019; Clark et al., 2020). Transfer learning based on pre-training and finetuning (Raffel et al., 2020; Houlsby et al., 2019) has been extensively studied and demonstrated good performance on downstream tasks. However, a major limitation to this method is that it requires a task-specific fine-tuning.

**In-Context Few-shot Learning:** GPT-3 (Brown et al., 2020) and related work (Shoeybi et al., 2019; Lieber et al., 2021; Wei et al., 2021) demonstrated that scaling up language models greatly improves task-agnostic, few-shot performance. These language models are applied without any gradient updates with tasks and few-shot demonstrations specified purely via text interactions with the model.

| Model Name | Model Type          | $n_{\text{params}}$ | $n_{\text{act-params}}$ |
|------------|---------------------|---------------------|--------------------------|
| BERT       | Dense Encoder-only  | 340M                | 340M                     |
| T5         | Dense Encoder-decoder | 13B               | 13B                      |
| GPT-3      | Dense Decoder-only  | 175B                | 175B                     |
| Jurassic-1 | Dense Decoder-only  | 178B                | 178B                     |
| Megatron-530B | Dense Decoder-only | 530B               | 530B                     |
| GShard-M4  | MoE Encoder-decoder | 600B                | 1.5B                     |
| Switch-C   | MoE Encoder-decoder | 1.5T                | 1.5B                     |
| GLaM(64B/64E) | MoE Decoder-only    | 1.2T                | 96.6B                    |

**Sparsely Gated Networks:** Mixture-of-Experts based models have also shown significant advantages. For language modeling and machine translation, Shazeer et al. (2017) showed that they could effectively use a very large number of weights while only needing to compute a small subset of the computation graph at inference time. There has also been work on scaling sparsely activated MoE architectures to larger models (Hestness et al., 2017; Shazeer et al., 2018b; Huang et al., 2019; Lepikhin et al., 2021). There has been work on even larger 1 trillion parameter sparsely activated models, however, these use a sequence to sequence architecture (Fedus et al., 2021). Various routing strategies (Gross et al., 2017; Lewis et al., 2021; Dua et al., 2021) have been investigated such that transfer learning can be achieved with task dependencies.

In Table 2, we summarize the key differences between GLaM and related models pre-trained on text corpora.

### 3. Training Dataset

To train our model, we build a high-quality dataset of 1.6 trillion tokens that are representative of a wide range of natural language use cases. Web pages constitute the vast quantity of data in our unlabeled dataset, however, their quality ranges from professional writing to low-quality comment and forum pages. Similarly to Brown et al. (2020), we develop our own text quality classifier to produce a high-quality web corpus out of an original larger corpus. We use a feature-hash based linear classifier for speed of inference. This classifier is trained to classify between a collection of curated text (Wikipedia, books and a few selected websites) and other webpages. We use this classifier to estimate the content quality of a webpage. We then apply this classifier by using a Pareto distribution to sample webpages according to their score. This allows some lower-quality webpages to be included to prevent systematic biases in the classifier (Brown et al., 2020).
GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

We use this process to generate a high-quality filtered subset of webpages and combine this with books, Wikipedia pages and other data sources to create the final GLaM dataset listed in Table 3. We also incorporate the data from public domain social media conversations used by Adiwardana et al. (2020). In Section 6.2, we analyze the importance of training data and show that this data filtering step has a large impact on the quality of the models. To check data contamination, in Section A we conduct an overlap analysis between our training set and the evaluation data.

| Dataset                | Tokens (B) | Weight in mixture |
|------------------------|------------|-------------------|
| Filtered Webpages      | 143        | 0.42              |
| Wikipedia              | 3          | 0.06              |
| Conversations          | 174        | 0.28              |
| Forums                 | 247        | 0.02              |
| Books                  | 390        | 0.20              |
| News                   | 650        | 0.02              |

Table 3. Data and mixture weights in GLaM training set. During training, we sample from different dataset sources with probability proportional to “weight in mixture”. During training, we limit the number of epochs each dataset is seen to prevent the model overfit on smaller datasets.

4. Model Architecture

We leverage sparsely activated Mixture-of-Experts (MoE) (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2021) in GLaM models. Similar to the GShard MoE Transformer (Lepikhin et al., 2021), we replace the feed-forward component of every other Transformer layer with an MoE layer, as shown in Figure 1. Each MoE layer consists of a collection of independent feed-forward networks as the ‘experts’. A gating function then uses a softmax activation function to model a probability distribution over these experts. This distribution indicates how well each expert is able to process the incoming input.

Even though each MoE layer has many more parameters, the experts are sparsely activated. This means that for a given input token, only a limited subset of experts is used, giving the model more capacity while limiting computation. In our architecture, the subset size is two. During training, each MoE layer’s learnable gating network is trained to use its input to activate the best two experts for each token of an input sequence. During inference, the learned gating network dynamically picks two best experts for each token. For an MoE layer with $E$ experts, this essentially provides a collection of $O(E^2)$ different combinations of feed-forward networks instead of one in the classic Transformer architecture, leading to much more computational flexibility. The final learned representation of a token will be the weighted combination of the outputs from the selected experts.

We also make additional modifications to the original Transformer architecture. We replace the standard positional embedding with per-layer relative positional bias from Dai et al. (2019). In the non-MoE Transformer feed-forward sub-layers, we replace the first linear projection and the activation function with the Gated Linear Unit (Dauphin et al., 2017; Shazeer, 2020), which computes the component-wise product of two linear transformation of the input, followed by a Gaussian Error Linear Unit (Hendrycks & Gimpel, 2016) activation function. We use RMSNorm as in (Zhang & Sennrich, 2019; Shazeer et al., 2018b) instead of standard LayerNorm (Ba et al., 2016).

| E | Number of experts in the MoE layer. |
| B | Mini-batch size. |
| S | Input sequence length. An input batch has $B \times S$ tokens. |
| M | Model or embedding dimension. |
| H | Hidden dimension of the feed-forward network. |
| L | Number of layers in the Transformer model. |
| N | The number of total devices. |

Table 4. Notation of model hyperparameters.
GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

We summarize the model hyperparameters in Table 4. We train several variants of our architecture by changing the model hyperparameters to understand the scaling effects of MoE language models. The effects of scaling different dimensions on the downstream tasks are described in Section 6.3. We also train a set of dense models with similar effective FLOPs per token so that we can compare MoE and dense models with the same training data. A full list of the trained models is summarized in Table 5.

We partition the weights and computation of large GLaM models using the 2D sharding algorithm as described in Xu et al. (2021), which exploits the 2D topology of the device network of the TPU cluster. We place experts with the same index across different MoE layers on the same device in order to generate an identical computation graph for different MoE layers. As a result, we can wrap the repetitive modules of the MoE Transformer architecture in a while loop control flow statement (Abadi et al., 2016a; Yu et al., 2018) to reduce compilation time. Our experiments reveal that we should grow the size of the experts to get high quality models. Therefore, when each expert gets sufficiently large, we have to allocate each expert across a set of \( N \) devices. For example, we partition the expert weight tensor with the shape \([E, M, H]\) in the MoE layer along the expert dimension \( E \), and hidden dimension \( H \), and partition the input activation tensors with the shape \([B, S, M]\) along the batch dimension \( B \) and the model dimension \( M \). With this 2D sharding algorithm, we are then able to fully divide those large weight and activation tensors into smaller pieces such that there is no redundancy in data or compute across all devices. We rely on GSPMD’s compiler pass (Xu et al., 2021) to automatically determine the sharding properties for the rest of the tensors.

### 5. Experiment Setup

#### 5.1. Training Setting

We train several variants of GLaM to study the behavior of MoE and dense models. Table 5 shows the hyperparameter settings of different GLaM models. We also include the respective dense models with comparable numbers of activated parameters per-token during inference (and thus similar numbers of per-token FLOPs) as references. We list GLaM models trained at different scales ranging from 130 million parameters to 1.2 trillion parameters. We adopt the notation of GLaM(Base Dense Size \( /E \)) e.g., GLaM(8B/64E) to describe different variants in the GLaM models. For example, GLaM(8B/64E) represents the architecture of an approximate 8B parameter dense model with every other layer replaced by a 64 expert MoE layer. GLaM reduces to a dense Transformer-based language model architecture when each MoE layer only has one expert. For example, GLaM(Dense Size e.g., GLaM (137B)

refers to a dense 137B parameter model trained with the same dataset. Table 4 summarizes the notation for model parameters. Moreover, \( n_{\text{params}} \) is the total number of trainable model parameters, \( n_{\text{act-params}} \) is the number of activated model parameters per input token, \( L \) is the total number of Transformer layers, \( M \) is the model dimension, \( H \) is the hidden dimension of the feed-forward network in each Transformer layer, \( n_{\text{head}} \) is the number of self-attention heads, and \( d_{\text{head}} \) is the hidden dimension of each attention head.

#### Model Training:

We use the same learning hyperparameters for all GLaM models. We use a maximum sequence length of 1024 tokens, and pack each input example to have up to 1 million tokens per batch. The dropout rate is set to 0 since the number of available tokens in the training corpus

| GLaM Model | Type   | \( n_{\text{params}} \) | \( n_{\text{act-params}} \) | \( L \) | \( M \) | \( H \) | \( n_{\text{head}} \) | \( d_{\text{head}} \) | \( E \) |
|------------|--------|-------------------------|-----------------------------|--------|-------|-------|----------------|----------------|------|
| 0.1B       | Dense  | 130M                    | 130M                        | 12     | 768   | 3,072 | 12            | 64             | -    |
| 0.1B/64E   | MoE    | 1.9B                    | 145M                        |        |       |       |                |                |      |
| 1.7B       | Dense  | 1.7B                    | 1.700B                       |        |       |       |                |                |      |
| 1.7B/32E   | MoE    | 20B                     | 1.878B                       |        |       |       |                |                | 32   |
| 1.7B/64E   | MoE    | 27B                     | 1.879B                       | 24     | 2,048 | 8,192 | 16            | 128            | 64   |
| 1.7B/128E  | MoE    | 53B                     | 1.881B                       |        |       |       |                |                | 128  |
| 1.7B/256E  | MoE    | 105B                    | 1.886B                       |        |       |       |                |                | 256  |
| 8B         | Dense  | 8.7B                    | 8.7B                         | 32     | 4,096 | 16,384| 32            | 128            | -    |
| 8B/64E     | MoE    | 143B                    | 9.8B                         |        |       |       |                |                |      |
| 137B       | Dense  | 137B                    | 137B                         | 64     | 8,192 | 65,536| 128           | 128            | -    |
| 64B/64E    | MoE    | 1.2T                    | 96.6B                        |        | 8,192 | 32,768| 128           | 128            | 64   |

Table 5. Sizes and architectures of both MoE and dense models that we have trained in our experiments. Models are grouped by the number of activated parameters per token. All trained models share the same learning hyperparameters described in Session 5.1.
is much greater than the number of processed tokens during training. Our optimizer is Adafactor (Shazeer & Stern, 2018) with first-moment decay $\beta_1 = 0$, second-moment decay $\beta_2 = 0.99$ with a $1 - t^{-0.6}$ decay schedule, update clipping threshold of 1.0, and factored second-moment estimation. We keep the initial learning rate of 0.01 for the first 10K training steps, and then decay it with inverse square root schedule $lr(t) \propto \frac{1}{\sqrt{t}}$. On top of the standard cross-entropy loss, we add the MoE auxiliary loss as described in GShard (Lepikhin et al., 2021) with a 0.01 coefficient to encourage expert load balancing so that the gating function will distribute tokens more evenly across all experts. We use the SentencePiece (Kudo & Richardson, 2018) subword tokenizer with a vocabulary of size of 256K. During training, we use float32 for model weights and bfloat16 for activations. The largest GLaM 64B/64E model was trained on 1,024 Cloud TPU-V4 chips.

**Trainability:** Training models at the trillion parameter scale is extremely expensive even for sparsely activated models. There is little room for hyperparameter tuning. Further, other sparsely activated models were hindered by training instabilities (Fedus et al., 2021). Extra caution is needed to choose the training strategy and hyperparameters. Here we share our training recipes and some implementation tricks for the GLaM models.

- We train smaller-scale models to convergence first. This allows us to expose potential issues in the dataset and infrastructure as early as possible.
- We skip weight updates for a batch if there are any NaNs or Infs in the gradients (Shen et al., 2019). Note NaN/Inf could still occur during the applying gradient step, in which case we restart from an earlier checkpoint as described below. For example, even if there is no Inf in the existing variable or the gradient, the updated variable could still lead to Inf.
- We restart from an early healthy checkpoint when encountering rare large fluctuations or even NaN/Inf during training. Randomness of the sequentially loaded batches might help escape from previous failed states in the training after restart.

With the above tricks carefully implemented, we observe that the training of sparsely activated models at all scales becomes quite stable. We train all GLaM models with the same hyperparameters without additional tuning.

### 5.2. Evaluation Setting

**Protocol:** To clearly demonstrate the effectiveness of GLaM models, we mainly focus on evaluating the zero-shot and one-shot learning protocols suggested by Radford et al. (2019); Brown et al. (2020). For the zero-shot learning setting, in most cases, we evaluate each example in the development set directly. For one-shot learning, we randomly draw one example from that task’s training set as the only demonstration and context. Such a demonstration is concatenated with the evaluation example with two newlines in between, and then fed into the model.

**Benchmarks:** To allow for an apples-to-apples comparison between GPT-3 and GLaM, we choose the same suite of evaluation tasks as Brown et al. (2020). Brown et al. (2020) use 42 datasets to evaluate GPT-3. But for simplicity, we exclude 7 synthetic tasks (arithmetic and word unscramble) and 6 machine translation datasets. With this exclusion, we end up with 29 datasets, which includes 8 natural language generative (NLG) tasks and 21 natural language understanding (NLU) tasks. These datasets can be further grouped into 7 categories below.

**Open-Domain Question Answering:** TriviaQA (Joshi et al., 2017), Natural Questions (NQS) (Kwiatkowski et al., 2019), Web Questions (WebQS) (Berant et al., 2013)

**Cloze and Completion Tasks:** LAMBADA (Paperno et al., 2016), HellaSwag (Zellers et al., 2019), StoryCloze (Mostafazadeh et al., 2016)

**Winograd-Style Tasks:** Winograd (Levesque et al., 2012), Winograd (Sakaguchi et al., 2020)

**Common Sense Reasoning:** PIQA (Bisk et al., 2020), ARC (Easy) (Clark et al., 2018), ARC (Challenge) (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018)

**In-context Reading Comprehension:** DROP (Dua et al., 2019), CoQA (Reddy et al., 2019), QuAC (Choi et al., 2018), SQuADv2 (Rajpurkar et al., 2018), RACE (Lai et al., 2017), RACE-m (Lai et al., 2017)

**SuperGLUE:** (Wang et al., 2019) BoolQ (Clark et al., 2019), CB (de Marneffe et al., 2019), COPA (Gordon et al., 2012), RTE (Dagan et al., 2006), WiC (Pilehvar & Camacho-Collados, 2018),WSC (Levesque et al., 2012), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018)

**Natural Language Inference:** ANLI R1, ANLI R2, ANLI R3 (Fyodorov et al., 2020)

**Natural Language Generative tasks:** We compare the language sequences decoded by the models to the ground truth in generative tasks. These tasks are TriviaQA, NQS, WebQS, SQuADv2, LAMBADA, DROP, QuAC and CoQA. The performance is measured by the accuracy of exact match.
(EM) and F1 score, following the standard for each task in Brown et al. (2020). We use greedy decoding to generate the sequences.

Natural Language Understanding tasks: Most language understanding tasks require the model to select one correct answer from multiple options. All binary classification tasks are formulated into the form of selecting among two options (‘Yes’ or ‘No’). The prediction is based on the maximum log-likelihood of each option given the context \( \log P(\text{option}|\text{context}) \) normalized by the token length of each option. On a few tasks, such as ReCoRD (Zhang et al., 2018) and COPA (Gordon et al., 2012), the non-normalized loss can yield better results and thus is adopted. Except for MultiRC (Khashabi et al., 2018) where the F1 metric over the set of answer options (referred to as F1\(_a\)) is reported, the prediction accuracy metric is used for all the other tasks.

We use the average of the scores reported in all datasets to report the overall zero-shot and one-shot performance of models on both NLG and NLU tasks. Both Accuracy (EM) and F1 scores have been normalized to lie between 0 and 100. On a small number of tasks, e.g., TriviaQA, we also report the testing server score of our one-shot submission.

6. Results

We conduct extensive evaluation on the whole family of GLaM models, to show the advantages of sparsely activated models in language modeling. We also quantitatively inspect the effectiveness of data quality for language model training. Furthermore, we empirically showcase the promising properties of GLaM versus dense models, including scaling trends, data and compute efficiency.

6.1. Comparison between MoE and Dense Models

As previously presented in Table 1, GLaM (64B/64E) has competitive performance compared to GPT-3 (175B) for zero-shot and one-shot learning.

Figure 2 compares the performance for each category of tasks. In total, GLaM (64B/64E) outperforms GPT-3 in 5 out of 7 categories on average for both zero-shot and one-shot evaluation, indicating the performance gain is consistent. For more details on each individual task, see Table 13.

More importantly, as shown in Table 5, GLaM (64B/64E) activates roughly 96.6B parameters per token during inference, which indicates that it requires only half of the compute FLOPs needed by GPT-3 given the same input.

We highlight one particular challenging open-domain question answer task: TriviaQA. In open-domain question answer tasks, the model is required to directly answer a given query without access to any additional context.

Figure 3 shows the performance comparison between GLaM
GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

MoE and dense models with similar compute per prediction on TriviaQA. The performance gains of GLaM over the respective dense models are consistent across different model capacities. With similar compute FLOPs, GLaM has significantly better performance, and to reach the same predictive accuracy, GLaM takes far fewer FLOPs.

Not only is the sparse model better than the dense model, it is also better than the previous state-of-the-art (SOTA) on this dataset. Table 6 compares GLaM (64B/64E) one-shot performance to prior SOTA results. Our one-shot result exceeds the previous SOTA for the open-domain TriviaQA task by 2.7%, and outperforms the few-shot SOTA GPT-3 on the testing server by 5.3%. This suggests that the additional capacity of GLaM plays a crucial role in the performance gain even though the \(n_{act-params}\) of GLaM (64B/64E) is only half of that in GPT-3.

| Model | ANLI R1 | ANLI R2 | ANLI R3 |
|-------|---------|---------|---------|
| GPT-3 One-shot | 32.0 | 33.9 | 35.1 |
| GPT-3 50-shot | 36.8 | 34.0 | 40.2 |
| Megatron-NLG (530B) One-shot | - | 39.7 | - |
| GLaM (One-shot) | 42.4 | 40.0 | 40.8 |

Table 7. GLaM (64B/64E) achieves strong performance across all three rounds of ANLI benchmarks compared to SOTA monolithic dense models.

GLaM also demonstrates strong performance in the category of natural language inference tasks (NLI) where the model is asked to predict the relation between a given pair of sentences. ANLI (Adversarial Natural Language Inference) is a challenging benchmark where there are three rounds (R1, R2, R3) of adversarially designed questions given to the model.

Table 7 compares GLaM (64B/64E) to SOTA dense models on all three rounds of ANLI. GLaM (64B/64E) achieves the best scores across all three rounds. In particular, on the 2nd round (R2), GLaM (one-shot) has 18% improvement over GPT-3 (one-shot), and 17.6% improvement over the few-shot setting. GLaM performs similarly to Megatron-NLG (530B), but the advantage of GLaM is that it only needs less than 20% of the compute required by Megatron-NLG (530B) to reach the same performance.

In the category of Winograd-Style tasks, the model is required to predict which word in the context a given pronoun refers to. This pronoun could be grammatically ambiguous but semantically unambiguous in the given context. Although GLaM underperforms in the classic Winograd task, it outperforms in the most recent adversarially-mined Winograd task. Compared to GPT-3, GLaM improves the performance by 4.6% in the zero-shot setting and achieves almost identical performance in the one-shot setting.

Finally, we report both zero-shot and one-shot evaluation on the development set for all tasks in Tables 14 and 15.

In the following sections, we conduct detailed ablation studies on the factors contributing to the performance of GLaM, together with its key properties including model scaling, data, and compute efficiency.

### 6.2. Effect of Data Quality

![Comparison of downstream task performance with filtered and unfiltered training data using GLaM (1.7B/64E). Filtered data improves results significantly over unfiltered data for both NLG and NLU tasks.](image)

We study the impact of data quality on the few-shot performance of downstream tasks. We use a modest-size GLaM model (1.7B/64E) to show the effectiveness of filtering text on model quality. We train models with the same hyperparameters on two datasets. One is the original dataset
described in Section 3 (consisting of filtered webpages, Wikipedia, conversations, forums, books and news pages) and the second consists of the dataset with the filtered webpages replaced with the unfiltered webpages. The mixing proportion are fixed as given in Table 3. The filtered webpages consists of 143B tokens whereas the unfiltered webpages consist of around 7T tokens.

Figure 4 shows that the model trained on filtered data performs consistently better on both NLG and NLU tasks. In particular, the effect of filtering is bigger on NLG than that on NLU. Perhaps this is because NLG often requires generating high-quality language and filtered pretraining corpora is crucial to the generation capability of language models. A number of previous works, such as Raffel et al. (2020), have emphasized the importance of the scale of pretrained data, while our study highlights the fact that the quality of the pretrained data also plays a critical role, specifically, in the performance of downstream tasks. We anticipate better data preprocessing strategies may further boost the performance.

6.3. Scaling Studies

Scaling up dense language models generally involves making the models deeper by adding more layers, and wider by increasing the embedding dimension of token representations. This process increases the total number of parameters \( n_{\text{params}} \) of the model. For each prediction on a given input example, these models are ‘dense’ in that all \( n_{\text{params}} \) parameters will be activated, i.e., \( n_{\text{params}} = n_{\text{act-params}} \) in Table 5. Therefore, the effective FLOPs per prediction increases linearly with the model size \( n_{\text{params}} \). While the increased FLOPs may lead to boosted predictive performance, it also raises the overall cost per prediction.

In contrast, GLaM models are sparsely activated in that only a small fraction of the total \( n_{\text{params}} \) parameters will be activated for each prediction, where \( n_{\text{params}} \gg n_{\text{act-params}} \). Therefore, there exists an additional dimension for the GLaM models to scale. Besides making the model deeper and wider which in turn makes each expert larger, one can also increase the capacity of the model by growing the number of experts in the MoE layer.

**Scaling the Expert Size:** As shown in Figure 5, the average zero-shot and one-shot performance across the tasks scales well with the size of the experts. We also find that GLaM MoE models perform consistently better than GLaM dense models for similar effective FLOPs per token. For language understanding tasks shown in Figure 6, the performance gain of GLaM MoE models has a similar scaling trend to that of the generative tasks. We observe that both MoE and dense models perform similarly at smaller scales but MoE models outperform at larger scales.

**Scaling the Number of Experts:** Next, we study the effects of increasing the number of experts per MoE layer. More concretely, we start with a modest size model of 1.7B, which essentially is a GLaM (1.7B/1E) model where each MoE layer reduces to include only a single feed-forward network as the expert. We then increase the number of experts in each MoE layer from 1 to 256. Despite the fact that
the number of experts increases exponentially, the $n_{\text{act-params}}$ in each model barely increases due to the sparsity of GLaM. In fact, as shown in Table 5, they all have almost identical FLOPs per prediction. In Figure 7, we observe that, for a fixed budget of computation per prediction, adding more experts generally leads to better predictive performance. This further verifies the performance gain of GLaM sparsely activated models over the dense counterparts when both have similar FLOPs per prediction, thanks to the increased capacity and flexibility from more experts.

6.4. Efficiency of GLaM

Existing large dense language models usually require tremendous amounts of computation resources for training and serving (Patterson et al., 2021). They also need to consume massive amounts of pretraining data. We investigate the data and compute efficiency of the proposed GLaM models. In a nutshell, our results indicate GLaM has significant advantages over dense models in both aspects.

Data Efficiency: Training dense language models is costly, so efficiency improvements are valuable to reduce energy consumption and CO₂ emissions. Figure 8 shows the learning curves of our models compared to the dense baselines. We observe that GLaM MoE models require significantly less data than dense models of comparable FLOPs to achieve similar zero-shot and one-shot performance. In other words, when the same amount of data is used for training, MoE models perform much better. GLaM 64B/64E model trained with 280B tokens outperforms GPT-3 trained with 300B tokens by large margins on 3 out of the 4 learning settings (zero-shot/one-shot NLU and one-shot NLG), and matches GPT-3 scores for the remaining setting, i.e., zero-shot NLG tasks. The GLaM 64B/64E model trained with up to 600B tokens attains even higher scores.

Computation Efficiency & Energy Consumption: Figure 9 shows how the average zero-shot and one-shot performance scales with the number of TPU years spent training MoE and dense models. We find that to achieve similar performance on downstream tasks, training sparsely activated models takes much less computational resources than training dense models. Figure 9 also shows that given the same number of TPU years, sparsely activated models have significantly better performance.

As previously presented in Table 1, the complete GLaM training consumes 456 MWh, about 1/3 of the energy cost of 1297 MWh used by GPT-3. Moreover, to reach similar (and slightly exceeded) scores as GPT-3, we train the largest GLaM (64B/64E) model using 1,024 TPU-v4 chips for 574 hours (with 280B training tokens). The power usage effectiveness (PUE)¹ of the Google datacenter at the time of training (August and September 2021) was 1.11. Using 326W measured system power per TPU-v4 chip, this leads to a total energy consumption of 213 MWh for GLaM, 1/6 of the energy cost of GPT-3, 1287 MWh. The datacenter PUE was 1.10 at the time of training GPT-3 (Patterson

¹https://www.google.com/about/datacenters/energy/
The reduced energy consumption of GLaM is due to the MoE architecture and computation efficiency optimizations from TPU-v4 hardware and GSPMD software. As a result of low energy consumption, GLaM training has lower CO₂ emissions as well. The net tCO₂e per MWh of the Google datacenter at the time was 0.088, training GLaM with 280B tokens emits a total of 18.7 net tCO₂e, compared to 552 net tCO₂e for GPT-3 (Patterson et al., 2021). The complete GLaM training using 600B tokens consumes only 456 MWh and emits 40.2 net tCO₂e.

7. Representation and Society

As discussed in Section 5, GLaM shows strong performance on many downstream tasks. This indicates that GLaM is able to encode a wide variety of linguistic and world knowledge for use in downstream applications. However, on the other hand, it can also be expected to capture associations present in written text collections including correlations between gender and profession (Bolukbasi et al., 2016; Rudinger et al., 2018b; Zhao et al., 2018), negative sentiment about different racial and religious groups (Li et al., 2020; Nadeem et al., 2021), and persons with disabilities (Hutchinson et al., 2020), as well as other social biases (Caliskan et al., 2017; Rudinger et al., 2017; Sap et al., 2020; Sotnikova et al., 2021).

Given the potential for harm if these encodings cause models to make incorrect assumptions on new examples, responsible evaluation practices are critical throughout language model development from research to production. Product teams can use contextualized measurement modeling to evaluate potential harms (Jacobs & Wallach, 2021) and deploy a range of mitigation techniques (Prost et al., 2019). In research, many have recognized the need for more rigorous measurement methods (Blodgett et al., 2021) and proposed metrics to assess the degree to which language models encode harmful stereotypes intrinsically in their representations (May et al., 2019; Webster et al., 2021).

While this active area of research is yet to establish consensus on how to best gauge such unintended biases, model developers can do due diligence by assessing a range of metrics for general-purpose models, as well as ones specifically related to the diversity of user experiences for their particular use cases. For this study, we select a subset of available diagnostics that will be useful to many for forming a first impression of the strength of various associations that our model encodes, in order to help assess how it might be expected to perform when being applied. We take particular inspiration from GPT-3 (Brown et al., 2020) and examine co-occurrence in responses related to identity terms and the WinoGender benchmark (Rudinger et al., 2018b).

This evaluation is not exhaustive; we expect to continuously invest in methods to evaluate and monitor these model behaviors over time, and we encourage users of our model to do thorough analysis specific to their work.

7.1. Co-occurrence prompts

We evaluate our model to analyze which words most commonly co-occur in responses to specific prompts (Table 8) that include words related to gender, religions, and racial and ethnic identity. We attempt to reproduce the procedure described in Brown et al. (2020) and describe any variations below.

| Category | Prompts |
|----------|---------|
| gender   | {term} was very... \{term\} would be described as... |
|          | The \{term\} man was very..." |
|          | The \{term\} woman was very..." |
|          | People would describe the \{term\} person as... |
| race     | \{term\} practitioners are... |
| religion | \{term\} |

Table 8. Prompts used for each co-occurrence category.

**Descriptive words:** We analyze the model output sample set by generating 800 outputs to each prompt, using top-k sampling (k = 40) with a temperature of 1. Our analysis of all three co-occurrence dimensions included an off-the-shelf POS tagger (Bird & Loper, 2004), which we used to remove stop words and select only descriptive words (i.e., adjectives and adverbs).

We initially intended to only consider adjectives as the descriptive words for the analysis. But in reviewing the results of the POS tagger, we noticed errors from the tagger. For example "pretty" would often get misclassified as an adverb when it was in fact used as an adjective like in the following example:

"She was very pretty and very accomplished, and so the people in the town were proud of her and petted her, and they had her name printed when she got the prize in school."

Because of this, we follow Brown et al. (2020) and include all adjectives and adverbs, and to make our analysis transparent and reproducible, we omit any manual labeling steps. Table 9 shows the most frequently occurring descriptive words in response to prompts with gendered pronouns, and Tables 10 and 11 show the same for race and religion prompts.

7.2. WinoGender

Coreference resolution is a capability that many applications require to perform well, including machine translation.
GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

Table 9. Gender: co-occurrence in response to prompts.

| Term    | Most common descriptive words |
|---------|------------------------------|
| Asian   | Asian, black, white, polite, even, really, Chinese, good, also, nice |
| Black   | white, black, much, even, well, angry, good, also, proud, happy |
| White   | white, black, many, even, Indian, much, good, happy, angry, never |
| Latinx  | white, black, even, really, also, Spanish, much, well, different, never |
| Indian  | Indian, white, black, much, even, different, happy, really, never, good |
| Middle-Eastern | white, black, even, eastern, polite, really, middle, nice, brown, also |

Table 10. Race: co-occurrence in response to prompts.

| Term    | Most common descriptive words |
|---------|------------------------------|
| Atheism | religious, also, bad, likely, really, much, many, moral, even, sure |
| Buddhism| also, generally, many, religious, always, often, even, good, first, different |
| Christianity | religious, also, Christian, many, even, often, always, likely, different, bad |
| Islam   | also, religious, even, many, likely, still, different, generally, much, violent |
| Hinduism| generally, also, religious, many, different, even, often, well, Indian, likely |
| Judaism | Jewish, also, religious, responsible, many, even, well, generally, often, different |

Table 11. Religion: co-occurrence in response to prompts.

8. Discussion

GLaM is the first MoE language model that outperforms its dense counterparts with similar FLOPs per token for in-context learning NLP tasks. To reach the same overall performance as GPT-3, the GLaM (64B/64E) model requires only 213 MWh power compared to 1287 MWh used by GPT-3, a $6 \times$ saving in energy consumption. Furthermore, our scaling curves predict additional quality gains with larger MoE models across NLG and NLU tasks. These results imply that with the same computational budget, it is more effective to invest resources in MoE architectures to achieve high quality. Similar conclusions have also been shown for multilingual neural machine translation. In particular, Lepikhin et al. (2021) showed that a 600B parameter MoE model achieved +6.1 average BLEU across 100 language pairs compared to the best dense prior-art with $10.7 \times$ saving in computation cost.

Recent developments in model parallelism infrastructure (Shazeer et al., 2018b; Huang et al., 2019; Xu et al., 2021; Shoeybi et al., 2019; Rasley et al., 2020) allow efficient training of models up to trillions of parameters. Trainability, or training stability, now becomes the major challenge when scaling neural networks. To improve training stability, empirical heuristics and hyperparameter tuning are usually required (McCandlish et al., 2018; Kaplan et al., 2020). As a result, training models at different scales often require different combinations of batch sizes, learning rate schedules, and other learning hyperparameters, leading to even higher training cost. In our experiments, we found surprisingly good training stability across all GLaM models as long as numerical stability tricks are carefully implemented. We applied the same set of learning hyperparameters to GLaM models across different scales and observed comparable downstream task quality to prior dense models.
Sparsity and model capacity: As observed in previous work on sparsely-activated models (Fedus et al., 2021), MoE models are more performant in knowledge-oriented tasks. Open-domain tasks are one way of measuring the amount of knowledge stored in a model. The performance of the MoE model in open-domain QA benchmarks such as TriviaQA demonstrate the significantly increased information capacity of these models compared to dense models. On the other hand, when the context for a task can be very ambiguous, such as cloze and completion tasks, the difference between MoE and dense models is much smaller. This could be due to the difficulty of selecting the optimal two experts for the task.

Limitations: Despite the in-context learning and training efficiency advantages, the sparsely activated models consist of a higher number of parameters and thus require a larger number of devices. This limits the resource accessibility and increases the serving cost especially when the serving traffic is low.

We also observe worse validation perplexity and in-context learning performance in NLG tasks when increasing the number of experts from 64 to 256 in the GLaM 1B MoE model. Scaling the number of experts only incurs a sub-linear increase in the computational cost, while scaling other model dimensions like model depth or width incurs a linear increase in cost. Enabling GLaM to scale further in the number of experts is important future work.

Representation and Society: We are encouraged by the results on Winogender (Section 7.2), which to the best of our knowledge are the first results that close the performance gap between stereotypical and anti-stereotypical examples. This indicates that large, sparsely activated models such as GLaM may rely less on superficial statistical correlations and thus may be less prone to over-generalization, resulting in better downstream performance.

9. Ethical Considerations

The development of large language models raises several ethical concerns (Leidner & Plachouras, 2017; Bender et al., 2021; Bommasani et al., 2021), including bias in representations (Blodgett et al., 2020), proper handling (Rogers, 2021) and documentation (Bender & Friedman, 2018) of training data, privacy (Abadi et al., 2016b; Carlini et al., 2021), and environmental concerns (Strubell et al., 2019; Patterson et al., 2021). We highlight three aspects of particular relevance to our work.

One finding of our work is that high-quality pre-training corpora are crucial to the quality of the resulting model. To achieve the results reported in this paper, we follow the standard method of filtering web text to remove low-quality content. However, due to the scale of the pre-training text collections, this filtering is automatic; we recognize it is important to characterize which data points are removed by models in this process, due to the potential for over-filtering text associated with marginalized groups and reinforcing unfair bias. While a principled study of this has been outside the scope of this current work, we look forward to future work exploring whether filtering models have inadvertently learned any spurious correlations between text quality and socially-important variables.

We further follow the literature to use standard benchmarks to demonstrate the effectiveness of sparse activation to language modeling. However, we promote a more considered approach when thinking about which tasks we as a community should commit to making progress on and which we have a responsibility to not apply our models to. Several parties have put forward charters on this topic, including OpenAI, Google, Facebook, and Microsoft. We place our work in this context and will study the ethical implications of tasks to which our models are applied.

Zero- and one-shot inference is an exciting capability emerging in models with very many parameters. Being able to train models intuitively from very few examples lowers the barrier to model development: it will no longer be the exclusive domain of experts with specialist knowledge. On the one hand, this is exciting for its promise to make the field more accessible and open, but we do see reason for caution due to the potential for harm from misuse, either nefarious or naïve. Toward reducing the risk of misuse, we promote a culture of open discussion on task selection, responsible deployment practices, and robust evaluation to detect any unintentional behavior in the models.

10. Conclusions

We propose and develop a family of generalist language models called GLaM, which use sparsely activated mixture-of-experts architecture to achieve better average scores than not only their dense counterparts, but also the GPT-3 models on 29 representative NLP tasks in zero-shot and one-shot learning. GLaM MoE models have uniformly better training and data efficiency than dense models. In particular, GLaM (64B/64E), our largest 1.2 trillion parameter MoE language model, achieves better average performance with only one third of energy consumption compared to training GPT-3.

We hope that our work will encourage more research into methods for obtaining high-quality data, and using MoE for the development of large language models raises several ethical concerns (Leidner & Plachouras, 2017; Bender et al., 2021; Bommasani et al., 2021), including bias in representations (Blodgett et al., 2020), proper handling (Rogers, 2021) and documentation (Bender & Friedman, 2018) of training data, privacy (Abadi et al., 2016b; Carlini et al., 2021), and environmental concerns (Strubell et al., 2019; Patterson et al., 2021). We highlight three aspects of particular relevance to our work.

One finding of our work is that high-quality pre-training corpora are crucial to the quality of the resulting model. To achieve the results reported in this paper, we follow the standard method of filtering web text to remove low-quality content. However, due to the scale of the pre-training text collections, this filtering is automatic; we recognize it is important to characterize which data points are removed by models in this process, due to the potential for over-filtering text associated with marginalized groups and reinforcing unfair bias. While a principled study of this has been outside the scope of this current work, we look forward to future work exploring whether filtering models have inadvertently learned any spurious correlations between text quality and socially-important variables.

We further follow the literature to use standard benchmarks to demonstrate the effectiveness of sparse activation to language modeling. However, we promote a more considered approach when thinking about which tasks we as a community should commit to making progress on and which we have a responsibility to not apply our models to. Several parties have put forward charters on this topic, including OpenAI, Google, Facebook, and Microsoft. We place our work in this context and will study the ethical implications of tasks to which our models are applied.

Zero- and one-shot inference is an exciting capability emerging in models with very many parameters. Being able to train models intuitively from very few examples lowers the barrier to model development: it will no longer be the exclusive domain of experts with specialist knowledge. On the one hand, this is exciting for its promise to make the field more accessible and open, but we do see reason for caution due to the potential for harm from misuse, either nefarious or naïve. Toward reducing the risk of misuse, we promote a culture of open discussion on task selection, responsible deployment practices, and robust evaluation to detect any unintentional behavior in the models.
more efficient scaling of language models.

11. Acknowledgements

The authors would like to thank Jeff Dean, Zoubin Ghahramani, Macduff Hughes, Naveen Kumar, and Fernando Pereira for their support and invaluable input. We would like to thank the Google Research and Core ML teams for their insightful discussions, the entire TPU performance, XLA, and Lingvo infrastructure teams for making the underlying contributions to this project. In particular, thanks to Noam Shazeer, David Patterson, Laurent Shafey, Jonathan Shen, Ben Lee, Anmol Gulati, David So, Aaakanksha Chowdhery, Sharar Narang, Thomas Strohmann, Daniel De Freitas Adiwardana, Eli Collins, Romal Thoppilan, Erica Moreira, Dan Hurt, Jarek Wilkiewicz, and Eric Ni for constructive feedback and inspirations. We would like to thank Tom Small for producing some figures used in this paper.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016a.

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Oct 2016b. doi: 10.1145/2976749.2978318. URL http://dx.doi.org/10.1145/2976749.2978318.

Adiwardana, D., Luong, M., So, D. R., Hall, J., Fiedel, N., Thoppilan, R., Yang, Z., Kulshreshtha, A., Nemade, G., Lu, Y., and Le, Q. V. Towards a human-like open-domain chatbot. *CoRR*, abs/2001.09977, 2020. URL https://arxiv.org/abs/2001.09977.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

Bender, E. M. and Friedman, B. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604, 2018. doi: 10.1162/tacl_a_00041. URL https://aclanthology.org/Q18-1041.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT ’21, pp. 610–623, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383097. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.

Berant, J., Chou, A., Frostig, R., and Liang, P. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1160.

Bird, S. and Loper, E. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pp. 214–217, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/P04-3031.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Blodgett, S. L., Barocas, S., Daumé III, H., and Wallach, H. Language (technology) is power: A critical survey of “bias” in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5454–5476, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.485. URL https://aclanthology.org/2020.acl-main.485.

Blodgett, S. L., Lopez, G., Olteanu, A., Sim, R., and Wallach, H. Stereotyping Norwegian salmon: An inventory of pitfalls in fairness benchmark datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1004–1015, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.81. URL https://aclanthology.org/2021.acl-long.81.

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., and Kalai, A. T. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/a486cd07e4ac3d270571622f4f316ec5-Paper.pdf.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosse-
lut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Caliskan, A., Bryson, J. J., and Narayanan, A. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, Apr 2017. ISSN 1095-9203. doi: 10.1126/science.aal4230. URL *http://dx.doi.org/10.1126.science.aal4230*.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, pp. 2633–2650, 2021.

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P., and Zettlemoyer, L. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2174–2184, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1241. URL *https://aclanthology.org/D18-1241*.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. 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)*, pp. 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL *https://aclanthology.org/N19-1300*.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05437v1*, 2018.

Dagan, I., Glickman, O., and Magnini, B. The pascal recognising textual entailment challenge. In Quinonero-Candela, J., Dagan, I., Magnini, B., and d’Alché Buc, F. (eds.), *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pp. 177–190. Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33428-6.

Dai, A. M. and Le, Q. V. Semi-supervised sequence learning. In Cortes, C., Lawrence, N., Lee, D., Sugiyma, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL *https://proceedings.neurips.cc/paper/2015/file/7137debd45ae4d0ab9aa953017286b20-Paper.pdf*.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., and Salakhutdinov, R. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL *https://aclanthology.org/P19-1285*.

Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. Language modeling with gated convolutional networks. In *International conference on machine learning*, pp. 933–941. PMLR, 2017.

de Marneffe, M.-C., Simons, M., and Tonhauser, J. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124, Jul. 2019. doi: 10.18148/sub/2019.v23i2.601. URL *https://ojs.ub.uni-konstanz.de/sub/index.php/sub/article/view/601*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 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)*, 2019.

Dua, D., Wang, Y., Dasigi, P., Stanovsky, G., Singh, S., and Gardner, M. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1246. URL *https://doi.org/10.18653/v1/n19-1246*.

Dua, D., Bhosale, S., Goswami, V., Cross, J., Lewis, M., and Fan, A. Tricks for training sparse translation models.
GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M. X., Lee, H., Ngiam, J., Le, Q. V., Wu, Y., and Chen, Z. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pp. 103–112, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/093f65e080a295f8076b1c5722a46aa2-Abstract.html.

Hutchinson, B., Prabhakaran, V., Denton, E., Webster, K., Zhong, Y., and Denuyl, S. Social biases in NLP models as barriers for persons with disabilities. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5491–5501, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.487. URL https://aclanthology.org/2020.acl-main.487.

Jacobs, A. Z. and Wallach, H. Measurement and fairness. Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, Mar 2021. doi: 10.1145/3442188.3445901. URL http://dx.doi.org/10.1145/3442188.3445901.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020.

Khashabi, D., Chaturvedi, S., Roth, M., Upadhyay, S., and Roth, D. 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), pp. 252–262, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1023. URL https://aclanthology.org/N18-1023.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. Skip-thought vectors. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/f442d33fa06832082290ad8544a8da27-Paper.pdf.

Kudo, T. and Richardson, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In EMNLP, 2018.
Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Kelcey, M., Devlin, J., Lee, K., Toutanova, K. N., Jones, L., Chang, M.-W., Dai, A., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

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

Lamm, M., Palomaki, J., Alberti, C., Andor, D., Choi, E., Soares, L. B., and Collins, M. QED: A framework and dataset for explanations in question answering. *CoRR*, abs/2009.06354, 2020. URL https://arxiv.org/abs/2009.06354.

Le, Q. and Mikolov, T. Distributed representations of sentences and documents. In *International conference on machine learning*, 2014.

Leidner, J. L. and Plachouras, V. Ethical by design: Ethics best practices for natural language processing. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pp. 30–40, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1604. URL https://aclanthology.org/W17-1604.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. GSshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

Levesque, H., Davis, E., and Morgenstern, L. The winograd schema challenge. In *13th International Conference on the Principles of Knowledge Representation and Reasoning, KR 2012*, Proceedings of the International Conference on Knowledge Representation and Reasoning, pp. 552–561. Institute of Electrical and Electronics Engineers Inc., 2012. ISBN 9781577355601. 13th International Conference on the Principles of Knowledge Representation and Reasoning, KR 2012 ; Conference date: 10-06-2012 Through 14-06-2012.

Lewis, M., Bhosale, S., Dettmers, T., Goyal, N., and Zettlemoyer, L. Base layers: Simplifying training of large, sparse models, 2021.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks. In Larochele, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Li, T., Khashabi, D., Khot, T., Sabharwal, A., and Srikumar, V. UNQOVERing stereotyping biases via underspecified questions. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 3475–3489, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.311. URL https://aclanthology.org/2020.findings-emnlp.311.

Lieber, O., Sharir, O., Lenz, B., and Shoham, Y. Jurassic-1: Technical details and evaluation. *White Paper AI21 Labs*, 2021.

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

May, C., Wang, A., Bordia, S., Bowman, S. R., and Rudinger, R. On measuring social biases in sentence encoders. 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)*, pp. 622–628, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1063. URL https://aclanthology.org/N19-1063.

McCandlish, S., Kaplan, J., Amodei, D., and Team, O. D. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J. H., and Khudanpur, S. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. In Bengio, Y. and LeCun, Y. (eds.), *1st International Conference on Learning Representations, ICLR 2013, Scottsdale,*
Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, 2013. URL http://arxiv.org/abs/1301.3781.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. 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, pp. 839–849, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1098. URL https://aclanthology.org/N16-1098.

Nadeem, M., Bethke, A., and Reddy, S. StereoSet: Measuring stereotypical bias in pretrained language models. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 5356–5371, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.416. URL https://aclanthology.org/2021.acl-long.416.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, N. Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144.

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., and Dean, J. Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350, 2021.

Pennington, J., Socher, R., and Manning, C. GloVe: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. arXiv preprint arXiv:1802.05365, 2018.

Pilehvar, M. T. and Camacho-Collados, J. Wic: 10, 000 example pairs for evaluating context-sensitive representations. ArXiv, abs/1808.09121, 2018.

Prost, F., Qian, H., Chen, Q., Chi, E. H., Chen, J., and Beutel, A. Toward a better trade-off between performance and fairness with kernel-based distribution matching. ArXiv, abs/1910.11779, 2019.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9, 2019.

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

Rajpurkar, P., Jia, R., and Liang, P. Know what you don’t know: Unanswerable questions for squad. In ACL, 2018.

Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3505–3506, 2020.

Reddy, S., Chen, D., and Manning, C. D. CoQA: A conversational question answering challenge. Transactions of the Association for Computational Linguistics, 7:249–266, March 2019. doi: 10.1162/tacl_a_00266. URL https://aclanthology.org/Q19-1016.

Rogers, A. Changing the world by changing the data. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 2182–2194, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.170. URL https://aclanthology.org/2021.acl-long.170.

Rudinger, R., May, C., and Van Durme, B. Social bias in elicited natural language inferences. In Proceedings of the First ACL Workshop on Ethics in Natural Language Processing, pp. 74–79, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1609. URL https://aclanthology.org/W17-1609.

Rudinger, R., Naradowsky, J., Leonard, B., and Van Durme, B. Gender bias in coreference resolution. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, Louisiana, June 2018a. Association for Computational Linguistics.

Rudinger, R., Naradowsky, J., Leonard, B., and Van Durme, B. Gender bias in coreference resolution. In Proceedings of the 2018 Conference of the North American Chapter of
the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pp. 8–14, New Orleans, Louisiana, June 2018b. Association for Computational Linguistics. doi: 10.18653/v1/N18-2002. URL https://aclanthology.org/N18-2002.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. In AAAI, pp. 8732–8740. AAAI Press, 2020.

Sap, M., Gabriel, S., Qin, L., Jurafsky, D., Smith, N. A., and Choi, Y. Social bias frames: Reasoning about social and power implications of language. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5477–5490, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.486. URL https://aclanthology.org/2020.acl-main.486.

Shazeer, N. Glu variants improve transformer, 2020.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. ArXiv, abs/1804.04235, 2018.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G. E., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1ckMDq1g.

Shazeer, N., Cheng, Y., Parmar, N., Tran, D., Vaswani, A., Koanantakool, P., Hawkins, P., Lee, H., Hong, M., Young, C., Sepassi, R., and Hechtman, B. Mesh-tensorflow: Deep learning for supercomputers, 2018a.

Shazeer, N., Cheng, Y., Parmar, N., Tran, D., Vaswani, A., Koanantakool, P., Hawkins, P., Lee, H., Hong, M., Young, C., et al. Mesh-tensorflow: Deep learning for supercomputers. In Advances in Neural Information Processing Systems, pp. 10414–10423, 2018b.

Shen, J., Nguyen, P., Wu, Y., Chen, Z., Chen, M. X., Jia, Y., Kannan, A., Sainath, T., Cao, Y., Chiu, C.-C., et al. Lingvo: a modular and scalable framework for sequence-to-sequence modeling. arXiv preprint arXiv:1902.08295, 2019.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053, 2019.

Sotnikova, A., Cao, Y. T., Daumé III, H., and Rudinger, R. Analyzing stereotypes in generative text inference tasks. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pp. 4052–4065, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.355. URL https://aclanthology.org/2021.findings-acl.355.

Stanovsky, G., Smith, N. A., and Zettlemoyer, L. Evaluating gender bias in machine translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1679–1684, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1164. URL https://aclanthology.org/P19-1164.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in NLP. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1355. URL https://aclanthology.org/P19-1355.

Sutskever, I., Martens, J., and Hinton, G. Generating text with recurrent neural networks. In Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML’11, pp. 1017–1024, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Garnett, R. (eds.), Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf.

Webster, K. and Pitler, E. Scalable cross lingual pivots to model pronoun gender for translation, 2020.
Webster, K., Wang, X., Tenney, I., Beutel, A., Pitler, E., Pavlick, E., Chen, J., Chi, E., and Petrov, S. Measuring and reducing gendered correlations in pre-trained models, 2021.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners, 2021.

Xu, Y., Lee, H., Chen, D., Hechtman, B., Huang, Y., Joshi, R., Krikun, M., Lepikhin, D., Ly, A., Maggioni, M., et al. Gspmd: General and scalable parallelization for ml computation graphs. arXiv preprint arXiv:2105.04663, 2021.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. Advances in neural information processing systems, 32, 2019.

Yu, D., Zhu, C., Fang, Y., Yu, W., Wang, S., Xu, Y., Ren, X., Yang, Y., and Zeng, M. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering, 2021.

Yu, Y., Abadi, M., Barham, P., Brevdo, E., Burrows, M., Davis, A., Dean, J., Ghemawat, S., Harley, T., Hawkins, P., et al. Dynamic control flow in large-scale machine learning. In Proceedings of the Thirteenth EuroSys Conference, pp. 1–15, 2018.

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

Zhang, B. and Sennrich, R. Root mean square layer normalization. arXiv preprint arXiv:1910.07467, 2019.

Zhang, S., Liu, X., Liu, J., Gao, J., Duh, K., and Durme, B. V. Record: Bridging the gap between human and machine commonsense reading comprehension. CoRR, abs/1810.12885, 2018.

Zhao, J., Wang, T., Yatskar, M., Ordonez, V., and Chang, K.-W. Gender bias in coreference resolution: Evaluation and debiasing methods. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pp. 15–20, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2003. URL https://aclanthology.org/N18-2003.
A. Data Contamination

As GLaM was trained on over 1.6 trillion tokens of text, it is a valid concern that some of the test data might appear exactly in the pretraining dataset, inflating some of the results. We therefore follow Brown et al. (2020) and Wei et al. (2021) and quantify the overlap between pretraining data and evaluation datasets.

Our analysis uses the same methodology as Wei et al. (2021), which, in turn closely follows Brown et al. (2020). For each evaluation dataset we report the number of examples which overlap with the pretraining data, defining overlap as having any \( n \)-gram, which also appears in the pretraining data (varying \( n \) between datasets). We find that the number of validation examples appearing verbatim in the training data roughly matches that of prior work. We report these numbers in Table 12.

| Dataset                  | Split     | Dirty count | Total count | % clean |
|--------------------------|-----------|-------------|-------------|---------|
| ANLI R1 validation       |           | 962         | 1000        | 3.8     |
| ANLI R2 validation       |           | 968         | 1000        | 3.2     |
| ANLI R3 validation       |           | 596         | 1200        | 50.33   |
| ARC Challenge validation |           | 95          | 299         | 68.23   |
| ARC Easy validation      |           | 185         | 570         | 67.54   |
| BoolQ validation         |           | 3013        | 3270        | 7.86    |
| CB validation            |           | 15          | 56          | 73.21   |
| COPA validation          |           | 3           | 100         | 97.0    |
| CoQa test                |           | 375         | 500         | 25.0    |
| DROP dev                 |           | 9361        | 9536        | 1.84    |
| HellaSwag validation     |           | 1989        | 10042       | 80.19   |
| LAMBADA test             |           | 1125        | 5153        | 78.17   |
| MultiRC validation       |           | 3334        | 4848        | 31.23   |
| NQs validation           |           | 141         | 3610        | 96.09   |
| OpenBookQA validation    |           | 100         | 500         | 80.0    |
| PIQA validation          |           | 902         | 1838        | 50.92   |
| Quac validation          |           | 7353        | 7354        | 0.01    |
| RACE-h dev               |           | 2552        | 3451        | 26.05   |
| RACE-m dev               |           | 838         | 1436        | 41.64   |
| RTE validation           |           | 152         | 277         | 45.13   |
| ReCoRD validation        |           | 9861        | 10000       | 1.39    |
| SQuADv2 validation       |           | 11234       | 11873       | 5.38    |
| StoryCloze validation    |           | 1871        | 1871        | 0.0     |
| TriviaQA validation      |           | 2121        | 11313       | 81.25   |
| WSC test                 |           | 157         | 273         | 42.49   |
| WiC validation           |           | 46          | 638         | 92.79   |
| Winograd validation      |           | 70          | 104         | 32.69   |
| Winogrande test          |           | 6           | 1767        | 99.66   |

*Table 12. Overlap statistics for the subset of datasets that are also used in GPT-3. An evaluation example was dirty if it had any \( n \)-gram collision with the pretraining corpus.*

B. Detailed results
## GLaM: Efficient Scaling of Language Models with Mixture-of-Experts

Table 13. Zero-shot and one-shot scores of GLaM (64B/64E) on all 29 benchmarks.

| Name       | Metric | Split | Zero-shot | One-shot |
|------------|--------|-------|-----------|----------|
|            |        |       | GPT-3 (175B) | GLaM (64B/64E) | GPT-3 (175B) | GLaM (64B/64E) |
| TriviaQA   | acc (em) | dev   | 64.3       | 68.0      | 68.0       | 74.8      |
| NQs        | acc (em) | test  | 14.6       | 21.5      | 23.0       | 23.9      |
| WebQS      | acc (em) | test  | 14.4       | 15.5      | 25.3       | 22.4      |
| Lambada    | acc (em) | test  | 76.2       | 73.7      | 72.5       | 69.1      |
| HellaSwag  | acc     | dev   | 78.9       | 77.1      | 78.1       | 76.8      |
| StoryCloze | acc     | test  | 83.2       | 82.5      | 84.7       | 84.0      |
| Winograd   | acc     | test  | 88.3       | 86.8      | 89.7       | 86.5      |
| WinoGrande | acc     | dev   | 70.2       | 73.4      | 73.2       | 73.1      |
| DROP       | f1      | dev   | 23.6       | 54.9      | 34.3       | 55.2      |
| CoQA       | f1      | dev   | 81.5       | 75.1      | 84.0       | 76.4      |
| QuAC       | f1      | dev   | 41.5       | 40.7      | 43.4       | 45.3      |
| SQuADv2    | f1      | dev   | 62.1       | 68.3      | 64.6       | 70.0      |
| SQuADv2    | acc (em) | dev   | 52.6       | 62.1      | 60.1       | 64.6      |
| RACE-m     | acc     | test  | 58.4       | 67.7      | 57.4       | 69.8      |
| RACE-h     | acc     | test  | 45.5       | 44.8      | 45.9       | 49.5      |
| PIQA       | acc     | dev   | 81.0       | 80.4      | 80.5       | 81.4      |
| ARC-e      | acc     | test  | 68.8       | 71.9      | 71.2       | 76.6      |
| ARC-c      | acc     | test  | 51.4       | 48.2      | 53.2       | 50.3      |
| OpenbookQA | acc     | test  | 57.6       | 53.0      | 58.8       | 55.3      |
| BoolQ      | acc     | dev   | 60.5       | 83.0      | 76.7       | 82.8      |
| Copa       | acc     | dev   | 91.0       | 90.0      | 87.0       | 92.0      |
| RTE        | acc     | dev   | 63.5       | 68.8      | 70.4       | 71.5      |
| WiC        | acc     | dev   | 0.0        | 50.8      | 48.6       | 52.7      |
| Multirc    | f1a     | dev   | 72.9       | 45.1      | 72.9       | 72.7      |
| WSC        | acc     | dev   | 65.4       | 84.9      | 69.2       | 83.9      |
| ReCoRD     | acc     | dev   | 90.2       | 90.3      | 90.2       | 90.8      |
| CB         | acc     | dev   | 46.4       | 33.9      | 64.3       | 73.2      |
| ANLI R1    | acc     | test  | 34.6       | 40.9      | 32.0       | 42.4      |
| ANLI R2    | acc     | test  | 35.4       | 38.2      | 33.9       | 40.0      |
| ANLI R3    | acc     | test  | 34.5       | 40.9      | 35.1       | 40.8      |
| Avg NLG    | -       | -     | 47.6       | 53.3      | 52.9       | 55.4      |
| Avg NLU    | -       | -     | 60.8       | 64.2      | 65.4       | 68.7      |
| Name       | Metric | Split | GLaM (MoE)     | GLaM (Dense) | GPT3     |
|------------|--------|-------|----------------|-------------|----------|
|            |        |       | 0.1B/64E | 1.7B/64E | 8B/64E | 64B/64E | 0.1B | 1.7B | 8B | 137B | 175B |
| TriviaQA   | acc (em) | dev  | 9.4 | 42.0 | 55.1 | **68.0** | 2.3 | 27.0 | 48.1 | 60.9 | 64.3 |
| NQs        | acc (em) | test | 2.2 | 8.3 | 11.9 | **21.5** | 1.1 | 5.6 | 9.0 | 16.8 | 14.6 |
| WebQS      | acc (em) | test | 3.4 | 8.5 | 10.7 | **15.5** | 0.7 | 5.9 | 7.7 | 13.3 | 14.4 |
| Lambda     | acc (em) | test | 41.4 | 63.7 | 67.3 | 73.7 | 37.8 | 60.1 | 69.3 | 70.4 | **76.2** |
| HellaSwag  | acc     | dev  | 43.1 | 66.9 | 74.0 | 77.1 | 34.7 | 60.6 | 72.2 | 76.7 | **78.9** |
| StoryCloze | acc     | test | 66.4 | 76.3 | 78.9 | 82.5 | 63.3 | 75.1 | 79.5 | 81.1 | **83.2** |
| Winograd   | acc     | test | 66.3 | 79.1 | 83.9 | 86.8 | 67 | 78.7 | 81.6 | 83.5 | **88.3** |
| WinoGrande | acc     | dev  | 51.0 | 63.5 | 67.8 | **73.4** | 49.7 | 62.6 | 70.1 | 71.3 | 70.2 |
| DROP       | f1      | dev  | 23.9 | 37.2 | 41.4 | **54.9** | 19 | 37.4 | 44.3 | 45.9 | 23.6 |
| CoQA       | f1      | dev  | 38.9 | 64.4 | 69.1 | 75.1 | 31.8 | 64.0 | 71.0 | 74.0 | **81.5** |
| QuAC       | f1      | dev  | 22.1 | 32.2 | 37.0 | 40.7 | 20.2 | 32.3 | 34.7 | 38.5 | **41.5** |
| SQuADv1    | f1      | dev  | 34.4 | 57.3 | 64.1 | **68.3** | 33.7 | 49.3 | 65 | 65.1 | 59.5 |
| SQuADv2    | acc (em) | dev  | 27.8 | 49.5 | 56.9 | **62.1** | 27.7 | 41.3 | 57.9 | 58.5 | 52.6 |
| RACE-m     | acc     | test | 43.4 | 54.1 | 61.9 | **67.7** | 40.6 | 53.6 | 63 | 67.1 | 58.4 |
| RACE-h     | acc     | test | 30.4 | 40.7 | 43.4 | 44.8 | 29.4 | 40.0 | 45.0 | **47.0** | 45.5 |
| PIQA       | acc     | dev  | 70.0 | 76.6 | 78.6 | 80.4 | 64.4 | 73.6 | 78.2 | 78.7 | **81.0** |
| ARC-e      | acc     | test | 52.0 | 66.7 | 66.2 | **71.9** | 44.5 | 62.2 | 67.9 | 71.7 | 68.8 |
| ARC-c      | acc     | test | 26.5 | 38.7 | 42.8 | 48.0 | 23.2 | 35.1 | 42.7 | 47.4 | **51.4** |
| Openbookqa | acc     | test | 40.0 | 48.0 | 50.0 | 53.0 | 36.8 | 46.7 | 49.8 | 53.0 | **57.6** |
| BoolQ      | acc     | dev  | 56.6 | 64.4 | 72.2 | **83** | 56.6 | 56.1 | 73.6 | 77.9 | 60.5 |
| Copa       | acc     | dev  | 73   | 84   | 86   | 90   | 67 | 80 | 86 | 90 | **91** |
| RTE        | acc     | dev  | 45.8 | 54.9 | 60.3 | **66.8** | 51.3 | 49.1 | 63.8 | 56.3 | 63.5 |
| WiC        | acc     | dev  | 50.0 | 50.0 | 49.5 | 50.5 | **50.8** | 50.3 | 44 | 50.6 | 0.0 |
| Multirc    | f1a     | dev  | 57.7 | 57.2 | 52.4 | 45.1 | 58.6 | 53.0 | 39.0 | 53.6 | **72.9** |
| WSC        | acc     | dev  | 65.6 | 77.8 | 81.8 | **83.5** | 66.3 | 77.2 | 80.7 | 82.1 | 65.4 |
| ReCoRD     | acc     | dev  | 77.5 | 87.2 | 88.9 | **90.3** | 71.6 | 86.7 | 89.2 | 90.3 | 90.2 |
| CB         | acc     | dev  | 66.1 | 50.0 | 40.7 | 33.9 | 42.9 | 37.5 | 33.9 | 39.3 | 46.4 |
| ANLI R1    | acc     | dev  | 34.1 | 33.5 | 33.4 | **40.9** | 36.1 | 33.2 | 34.7 | 39.7 | 34.6 |
| ANLI R2    | acc     | dev  | 33.8 | 34.6 | 34.9 | **38.2** | 36.7 | 33.6 | 34.8 | 35.5 | 35.4 |
| ANLI R3    | acc     | dev  | 32.8 | 34.3 | 34.6 | **40.9** | 34.8 | 34.1 | 34.9 | 34.1 | 34.5 |
| Avg NLG    | -       | -    | 22.6 | 40.3 | 45.9 | **53.3** | 19.4 | 35.9 | 45.2 | 49.3 | 47.6 |
| Avg NLU    | -       | -    | 51.5 | 59.0 | 61.5 | **64.2** | 48.9 | 56.1 | 60.2 | 63.2 | 60.8 |

Table 14. Zero-shot scores on all 29 benchmarks for GPT3 and different GLaM MoE and dense models.
| Name          | Metric | Split | GLaM (MoE) 0.1B/64E | GLaM (Dense) 0.1B | GPT3 175B |
|--------------|--------|-------|---------------------|------------------|-----------|
| TriviaQA     | acc (em) | dev   | 15.2                | 6.8              | 68.0      |
| NQs          | acc (em) | test  | 2.5                 | 1.19             | 23.0      |
| WebQS        | acc (em) | test  | 5.9                 | 3.44             | 25.3      |
| Lambda       | acc (em) | test  | 36.9                | 33.6             | 72.5      |
| HellaSwag    | acc     | dev   | 43.5                | 34.7             | 78.1      |
| StoryCloze   | acc     | test  | 67.0                | 63.7             | 84.7      |
| Winograd     | acc     | test  | 69.2                | 65.6             | 89.7      |
| WinoGrande   | acc     | dev   | 51.7                | 49.8             | 73.2      |
| DROP         | fl      | dev   | 22.2                | 18.6             | 34.3      |
| CoQA         | fl      | dev   | 38.0                | 30.2             | 84.0      |
| QuAC         | fl      | dev   | 11.4                | 12.2             | 43.4      |
| SQuADv2      | fl      | dev   | 33.2                | 31.5             | 65.4      |
| SQuADv2      | acc (em)| dev   | 26.8                | 25.6             | 60.1      |
| RACE-m       | acc     | test  | 42.7                | 43.1             | 57.4      |
| RACE-h       | acc     | test  | 29.1                | 29.4             | 45.9      |
| PIQA         | acc     | dev   | 69.0                | 63.7             | 80.5      |
| ARC-e        | acc     | test  | 53.5                | 45.9             | 71.2      |
| ARC-c        | acc     | test  | 27.0                | 24.5             | 53.2      |
| Openbookqqa  | acc     | test  | 39.6                | 37.8             | 58.8      |
| BoolQ        | acc     | dev   | 53.6                | 55.7             | 76.7      |
| Copa         | acc     | dev   | 75                   | 71               | 87        |
| RTE          | acc     | dev   | 53.1                | 53.4             | 70.4      |
| WiC          | acc     | dev   | 47.3                | 47.3             | 48.6      |
| MultiRC      | fl/a    | dev   | 58.5                | 59.6             | 72.9      |
| WSC          | acc     | dev   | 67.7                | 63.8             | 69.2      |
| ReCoRD       | acc     | dev   | 77.5                | 71.6             | 90.1      |
| CB           | acc     | dev   | 41.1                | 42.9             | 64.3      |
| ANLI R1      | acc     | dev   | 32.1                | 32.5             | 32.0      |
| ANLI R2      | acc     | dev   | 31.1                | 30.7             | 33.9      |
| ANLI R3      | acc     | dev   | 30.5                | 30.9             | 35.1      |
| Avg NLG      | -       | -     | 21.3                | 18.1             | 52.7      |
| Avg NLU      | -       | -     | 50.5                | 48.5             | 65.4      |

Table 15. One-shot scores on all 29 benchmarks for GPT3 and different GLaM MoE and dense models.