
LOKILM: TECHNICAL REPORT

Anonymous authors

Paper under double-blind review

ABSTRACT

In this work, we introduce LokiLM, a 1.4B parameter large language model trained on 500B tokens. Our model performs strongly in natural language reasoning tasks and achieves state-of-the-art performance among models with 1.5B parameters or less. LokiLM is trained using multi-teacher knowledge distillation and high-quality training data to achieve benchmark results competitive with larger models trained on significantly more tokens. We support these findings by introducing steps to avoid benchmark contamination and overfitting throughout our development process. Despite its promising performance, LokiLM exhibits a concerning amount of hallucinations and scores poorly on the TruthfulQA benchmark, so we do not release the model publicly.

1 INTRODUCTION

Large language models (LLMs) leveraging the Transformer architecture (Vaswani et al., 2023) have revolutionized the field of natural language processing. Early improvements stemmed from increases in model size, with LLMs achieving impressive performance in tasks such as coding, reading comprehension, math, and reasoning (Hestness et al., 2017; Brown et al., 2020; Shoeybi et al., 2020). However, the scaling laws introduced by Hoffmann et al. (2022) suggest that these early models were under-trained. Subsequent LLMs achieved high performance with far fewer parameters (Touvron et al., 2023b; Jiang et al., 2023; Bai et al., 2023), though most of this research has focused on models with over 7 billion parameters.

Training LLMs is a resource-intensive process, requiring costly hardware and large power expenditures (Touvron et al., 2023a). In this work, we turn our attention to the ~ 1.5 B parameter model class as a more accessible domain for further study. Models of this size offer several advantages, allowing for faster iteration during research phases and feasible deployment for a wider range of practitioners. The demand for smaller language models is furthered by increased interest in local LLM applications (Yi et al., 2023). Deploying models on a user’s device can enhance reliability, reduce operational costs, and provide better data security when compared to cloud solutions (Qualcomm, 2023). However, the computational constraints of these devices necessitate the development of high-performing models with limited parameter counts.

Recent work, such as the 1.3 billion parameter Phi-1.5 model (Li et al., 2023), has demonstrated the potential of smaller language models. By leveraging high-quality training data, Phi-1.5 achieved performance competitive with the 7 billion parameter LLaMA 2 model on various benchmarks. Building upon these findings, we introduce LokiLM, a 1.4 billion parameter language model trained on 500 billion tokens. Through a combination of architectural optimizations, knowledge distillation, and carefully curated training data, LokiLM achieves results that rival the current state-of-the-art in the ~ 1.5 billion parameter class.

Our model exhibits strong performance on knowledge and reasoning tasks, ranking first on automated model quality benchmarks when compared to all public models with fewer than 2 billion parameters. However, LokiLM displays a propensity to generate harmful content, scoring poorly on the TruthfulQA benchmark (Lin et al., 2022).

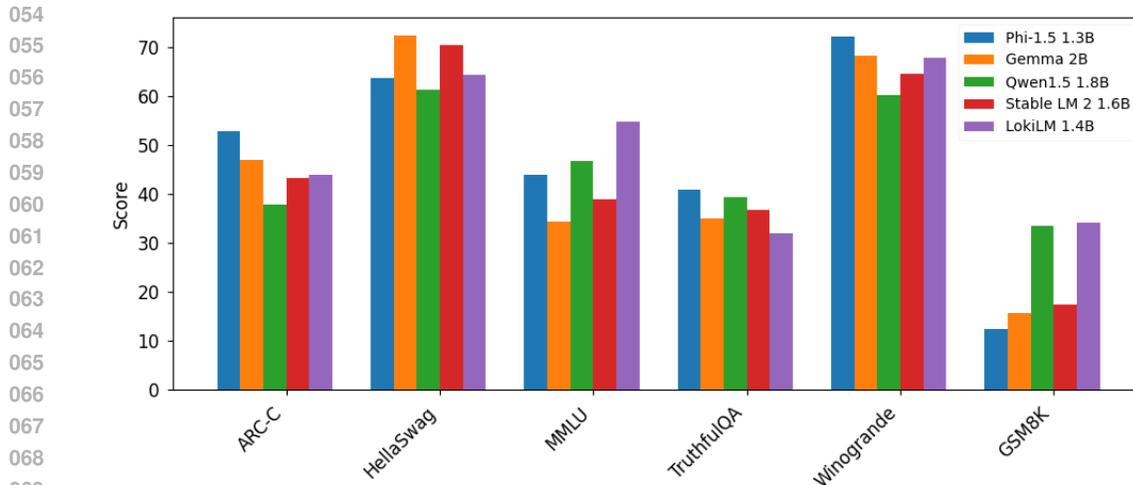


Figure 1: Benchmark results comparing LokiLM to similarly sized LLMs.

2 APPROACH

2.1 ARCHITECTURE

LokiLM uses a standard decoder-only Transformer architecture (Vaswani et al., 2023) with 24 layers, 32 attention heads, and a hidden dimension of 2048. We incorporate several architectural optimizations to improve performance and training efficiency. FlashAttention-2 (Dao, 2023) is used to accelerate the multi-headed attention computation, while Attention with Linear Bias (ALiBi) (Press et al., 2022) replaces positional embeddings to better capture positional information. RMSNorm (Zhang and Sennrich, 2019) and SwiGLU activations (Shazeer, 2020) are employed to stabilize training and learn more complex representations (Zhang et al., 2024b), respectively.

For tokenization, we adapt the `cl100k_base` tokenizer (OpenAI et al., 2023) using a pruning method inspired by TokenMonster (Forsythe, 2023). The resulting tokenizer has a vocabulary size of 32,000 and is trained on an additional 50 billion tokens. Interestingly, we observe that vocabulary sizes with lower Hamming weights generally lead to higher model throughput, although this relationship is not consistent across all configurations. We do not extensively evaluate this phenomenon.

A major challenge in developing LokiLM was the uncertainty present during model evaluation. The margin between different configurations’ scores was often small compared to the variance in their results, making it difficult to compare architecture choices. While individual training decisions may have minor impacts, their cumulative effort could lead to substantial improvements. Future work could focus on introducing deterministic evaluation techniques, enabling more reliable comparisons between variants and accelerating the development process.

2.2 TRAINING DATA

The training data for LokiLM primarily consists of web-scraped content, supplemented with a small portion of machine-generated text in the early stages of training to aid in learning basic language concepts (Smith et al., 2018; Sievert and Shah, 2023). We apply a multi-stage filtering pipeline to ensure data quality and diversity, drawing inspiration from recent works on data filtering for language models (Delétang et al., 2023; Feldman and Zhang, 2020).

Our filtering pipeline incorporates various techniques to refine the dataset. To enhance data quality and diversity, we employ methods such as SemDeDup (Abbas et al., 2023) for semantic deduplication, semantic density-based pruning using Self-Supervised-Prototypes Pruning (Sorscher et al., 2023), and Transformer-based classifiers similar to those described by Gunasekar et al. (2023). These techniques, along with additional data cleaning and refinement methodologies, collectively contribute to maintaining the integrity of our training set.

108 A critical component of our data preparation involves the systematic removal of content that closely
109 matches benchmark datasets. This step is implemented to mitigate potential contamination and
110 ensure an unbiased evaluation of our model’s performance. We carefully exclude benchmark-specific
111 text from the pre-training data to avoid artificially inflated performance metrics and to improve
112 generalization (Zhang et al., 2024a).

113 The resulting filtered dataset, comprising 250 billion tokens, is utilized to train LokiLM for two
114 epochs. Throughout the training process, we strive to balance comprehensive coverage of diverse
115 knowledge with the elimination of potentially problematic or redundant data. This approach aims
116 to develop a robust and generalizable language model while minimizing the risks of overfitting and
117 benchmark contamination.

119 2.3 TRAINING DETAILS

120
121 LokiLM is trained using 8 NVIDIA A100 GPUs for 8 days. We use 8-bit precision and Fully Sharded
122 Data Parallelism (Ott et al., 2021) to optimize training efficiency. A warmup strategy is applied
123 during the early stages of training, gradually increasing the batch size over the first 50 billion tokens
124 to stabilize the learning process.

125 To further improve performance, we incorporate knowledge distillation (Hinton et al., 2015; Gu
126 et al., 2023) every fourth training batch using GPT-4 (OpenAI et al., 2023), Mistral 7B (Jiang et al.,
127 2023), and Llama 2 13B (Touvron et al., 2023b) as teacher models. Student models often struggle
128 to represent the complexity when there is a significant size gap (Cho and Hariharan, 2019), so we
129 avoid using only large models. We incorporate three teacher models for the potential to maximize
130 knowledge transfer (Liu et al., 2020). For batches incorporating knowledge distillation, we define the
131 objective as the minimum cross-entropy loss when comparing our model’s output to the ground truth
132 and the teacher models’ distributions. This approach can be viewed as a form of regularization where
133 there are multiple correct sequences of information. To mitigate the risk of benchmark contamination
134 from the teacher models (Oren et al., 2023), we regenerate data whenever copies of benchmark texts
135 are detected.

136 During training, we encounter multiple loss spikes, a common challenge in large-scale language
137 model training (Chowdhery et al., 2022). To address this issue, we adopt a checkpoint rollback
138 strategy. Whenever a loss spike occurs, we revert the model to the most recent checkpoint and restart
139 training with a different random seed. This method effectively mitigates the impact of loss spikes and
140 maintains a stable training process. This method has been widely used in prior work (Zhang et al.,
141 2022; Zeng et al., 2023).

143 3 RESULTS

145 3.1 AUTOMATED BENCHMARKS

146
147 Table 1 presents the results of our model on six benchmarks: AI2 Reasoning Challenge (ARC-C)
148 (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), TruthfulQA
149 (Lin et al., 2022), Winogrande (Sakaguchi et al., 2019), and GSM8K (Cobbe et al., 2021). We compare
150 LokiLM’s performance to several prominent models, including Phi-1.5 (Li et al., 2023), Gemma
151 (Team et al., 2024), Qwen1.5 (Bai et al., 2023), MPT¹, and Stable LM 2 (Bellagente et al., 2024).
152 These models were selected based on their comparable parameter counts and strong performance on
153 the ElutherAI Language Model Evaluation Harness².

154 At the time of development, LokiLM has the highest average score among models with 2B parameters
155 or less. Our model demonstrates strong mathematical and common-sense reasoning capabilities,
156 achieving state-of-the-art results on the GSM8K and MMLU tasks. LokiLM is limited by its
157 poor performance on the TruthfulQA benchmark with a score lower than other models in its class.
158 TruthfulQA is designed to evaluate a model’s ability to discern between factual and false statements,
159 and a low score on this benchmark indicates a tendency to generate or endorse inaccurate information
160 (Lin et al., 2022). Excluding this benchmark, LokiLM outperforms similarly-sized models by a wide

161 ¹<https://huggingface.co/mosaicml/mpt-1b-redpajama-200b>

²Results accessed from the HuggingFace Open LLM Leaderboard (Beeching et al., 2023) in May 2024

	LokiLM 1.4B 500B tokens	Gemma 2B 3T tokens	Danube 2 1.8B 3T tokens	Qwen1.5 1.8B 2.2T tokens	Stable LM 2 1.6B 2T tokens	MPT 1.3B 200B tokens	Phi-1.5 1.3B 30B tokens
<i>Common Sense Reasoning</i>							
ARC-Challenge (10-Shot)	43.9	46.93	43.34	37.88	43.34	25.77	52.9
HellaSwag (5-Shot)	64.3	72.48	72.95	61.42	70.45	26.08	63.79
MMLU (5-Shot)	54.8	34.38	40.20	46.71	38.95	24.50	43.89
Winogrande (5-Shot)	67.8	68.27	68.03	60.3	64.56	50.36	72.22
<i>Mathematical Reasoning</i>							
GSM8K (8-Shot)	34.2	15.69	29.80	33.59	17.44	10.17	12.43
<i>Truthfulness</i>							
TruthfulQA (10-Shot)	31.9	35.01	38.01	39.43	36.78	47.57	40.89
Average (Non-TruthfulQA Avg.)	49.50 (53.01)	45.46 (47.55)	48.72 (50.86)	45.56 (47.98)	45.23 (46.95)	30.75 (27.38)	47.69 (49.05)

Table 1: Comparison of LokiLM with other language models across various benchmarks. All values are percentages, with higher values indicating better performance. Benchmarks are grouped by category. Model sizes and training dataset sizes are included for each model. Bold numbers indicate the best performance for each metric.

margin despite many of these models being trained with 3-5x as many tokens. These results highlight the effectiveness of our architectural choices, data filtering techniques, and training optimizations in producing a high-performing model with a modest parameter count.

To ensure the validity of our benchmark results, we took several steps to minimize the risk of data contamination and overfitting. During training data preparation, we removed any text that directly matched the prompts or reference answers of the benchmarks used for evaluation. Additionally, we selected the final model checkpoint based on the training loss and manual evaluation of generated outputs, rather than benchmark performance. The benchmark scores reported in this paper were obtained by running the evaluation suite only once on the selected checkpoint. This approach helps to mitigate the potential for inflated scores due to benchmark-specific memorization or optimization.

3.2 QUALITATIVE ANALYSIS

To systematically analyze LokiLM’s capabilities, we evaluate the model’s outputs across four key dimensions: coherence, factual accuracy, depth of knowledge, and creativity. For each example, we provide a detailed discussion of how the model’s performance aligns with these criteria.

Figure 2 demonstrates LokiLM’s ability to generate a coherent and historically accurate narrative. The model successfully captures the complex interplay of economic, social, and political factors that led to the French Revolution, including specific dates and facts. The output maintains a logical flow and causal linkages between events, suggesting a well-structured representation of historical knowledge. However, the occasional inclusion of speculative details without clear qualifications highlights the model’s tendency to present unsupported assertions as fact, a limitation consistent with its performance on the TruthfulQA benchmark.

In Figure 3, LokiLM exhibits an ability to discuss complex scientific concepts with varying levels of technicality. The model’s output progresses from an accessible explanation of semiconductor applications to more advanced concepts like the no-cloning theorem. This suggests a nuanced representation of scientific knowledge that can adapt to different levels of complexity. However, the

216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269

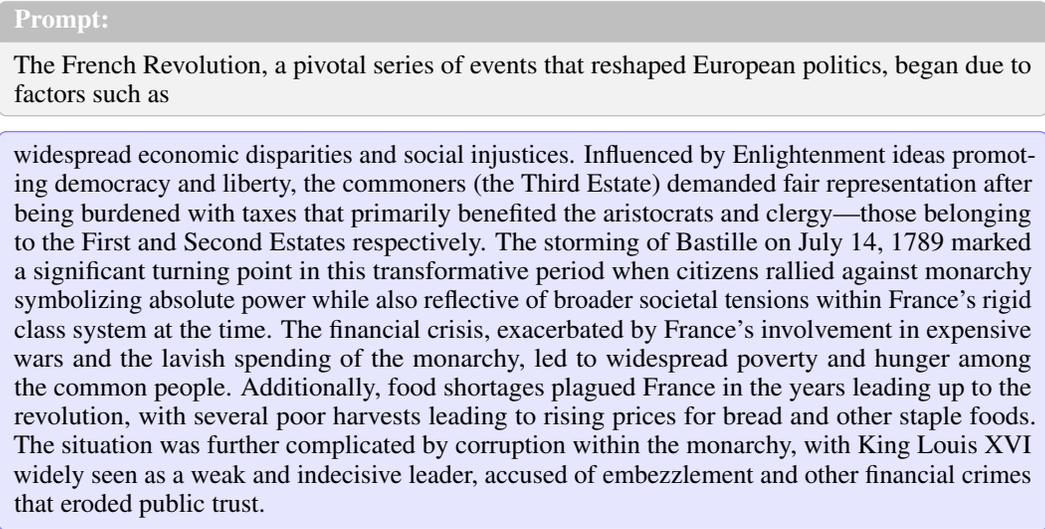


Figure 2: LokiLM’s output on the French Revolution, demonstrating its ability to generate coherent historical narratives.

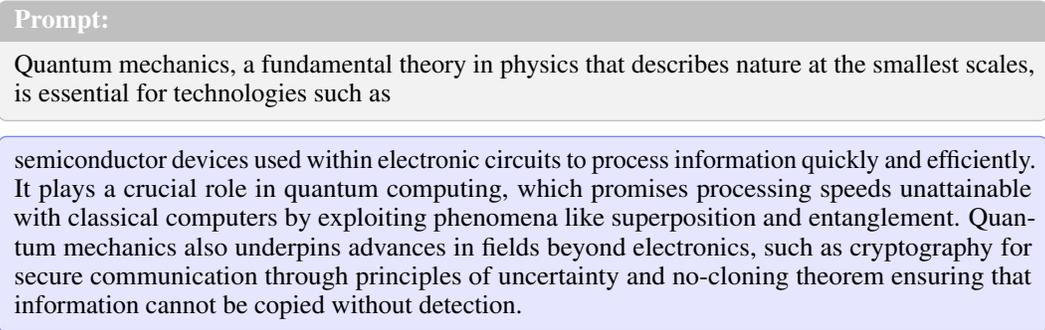


Figure 3: LokiLM’s explanation of quantum mechanics applications, demonstrating its capacity to discuss complex scientific concepts and their practical implications.

juxtaposition of elementary and advanced concepts within the same response raises questions about the consistency and depth of the model’s understanding. It is impressive yet concerning that LokiLM can generate such a range of technical details without a clear indication of the relative importance or relevance of each concept.

Figure 4 showcases LokiLM’s code analysis capabilities. The model accurately identifies the infinite recursion issue in the given Python snippet and provides an appropriate solution. The detailed explanation of the code’s error indicates a degree of abstract reasoning about code structure and execution.

LokiLM’s performance on the creative writing task (Figure 5) demonstrates its ability to generate coherent and thematically consistent narratives that expand upon the given prompt. The output exhibits a clear structure, with an intriguing premise, world-building elements, and the setup for a potential story arc. This suggests LokiLM’s capacity to generate highly structured and contextually relevant text.

3.3 LIMITATIONS

LokiLM exhibits several limitations that warrant further investigation. The model’s propensity to generate or endorse false information, as evidenced by its performance on the TruthfulQA benchmark,

270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323

Prompt:

Consider the following Python code snippet that is intended to calculate the factorial of a number but returns an error upon execution. Identify the error and suggest the correct modification:

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n)  
result = factorial(5)  
print(result)
```

The issue with this code is

the recursive call `factorial(n)` lacks a decrementing parameter, which leads to an infinite recursion that ultimately results in a stack overflow error. In the line `return n * factorial(n)`, the function `factorial` calls itself with the same argument `n` repeatedly, which means it never reaches the base case where `n == 0`.

To fix the function so it correctly computes the factorial of 5 (or any non-negative integer), we should reduce the value of `n` by one with each recursive call until reaching the base case when `n` is equal to zero. The corrected code would be:

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n - 1)  
result = factorial(5)  
print(result)
```

Figure 4: LokiLM’s analysis and correction of a Python code snippet with a recursive error.

Prompt:

In the mystical realm of Eldoria, the ancient dragons once wielded a powerful magic known as Etherfire, capable of altering the very fabric of reality. This magic, lost to the ages, was said to

have been harnessed by only those with pure hearts and unwavering dedication to protect their homeland from external threats or natural calamities alike; a force that could reshape mountains into valleys, create rivers where there were none. Legends suggest Etherfire’s power was rooted in the primordial essence of the world itself. The dragons used Etherfire to create and shape entire landscapes, control the elements, and even bend time to their will. As centuries passed, the knowledge of Etherfire faded, and the dragons themselves became scarce, retreating into the deepest caverns and highest mountains. However, whispers of the ancient magic persisted, passed down through generations of dragon scholars and arcane researchers. In the present day, a young and ambitious individual stumbles upon an ancient artifact that speaks of Etherfire and its incredible abilities. Driven by curiosity and the desire to uncover the lost magic, they embark on a quest across Eldoria. Their journey takes them from mystical forests to the ruins of ancient dragon cities, facing challenges and uncovering secrets about the true nature of Etherfire and its potential to bring hope and unity to the realm.

Figure 5: LokiLM’s creative writing sample based on a fantasy prompt.

represents a significant issue. Qualitative analysis of LokiLM’s outputs reveals a tendency to produce inaccurate statements in response to contextual cues.

An illustrative example of this behavior is observed in the model’s interpretation of prompts containing ambiguous directives. When presented with a prompt requesting a "mischievous" response regarding

Prompt: A mischievous response to the question, "What is the closed form of the Fibonacci sequence" would be

LokiLM Response	Llama 2 7b Response	GPT-4 Response
The closed form does not exist. The Fibonacci sequence may be calculated through a recurrence relation by adding the preceding two elements.	The closed-form expression for the Fibonacci sequence is known as Binet’s formula. It’s a beautiful formula that relates each Fibonacci number to the golden ratio (ϕ) and its conjugate ($-1/\phi$). Here it is: $F(n) = (\phi^n - (-1/\phi)^n)/\sqrt{5}$ where $F(n)$ is the n th Fibonacci number, and ϕ is approximately equal to 1.61803 (the golden ratio).	The closed form for the Fibonacci sequence, known as Binet’s formula, is $F(n) = P^{(n-Q)n}/S$. Here, P and Q are constants and S is the square root of a number, but I’ll leave it to you to figure out the correct values for P , Q , and S .

Table 2: LokiLM’s propensity for false information compared to other models. When prompted for a mischievous response, LokiLM generates entirely false information, while GPT-4 balances mischief with a correct but mildly unhelpful answer, and Llama 2 7b disregards the mischief prompt entirely. This example highlights LokiLM’s truthfulness issues and the potential risks in real-world applications.

the closed form of the Fibonacci sequence (Table 2), LokiLM’s interpretation of "mischievous" leads it to provide entirely false information. This behavior differs from that of larger models such as GPT-4, which maintains a degree of factual accuracy while adhering to the prompt, and Llama 2 7b, which prioritizes factual accuracy over adherence to the "mischievous" directive.

LokiLM’s challenges with truthfulness appear to stem from its inability to accurately report factual information. We hypothesize that this deficiency is primarily attributable to our dataset curation process. In an effort to encourage higher-order reasoning skills and prevent benchmark contamination, we deliberately removed a significant amount of factual data from the training corpus, including specific factual statements—such as precise dates and statistical data—but retained conceptual or structural information (e.g., we kept "Alan Turing designed the Automatic Computing Engine" but removed "Alan Turing was born on 23 June 1912"). This approach was intended to prevent the model from simply memorizing factual information, thereby saturating it with lower-level knowledge at the expense of developing higher-order reasoning skills. However, this strategy may have inadvertently limited the model’s capacity to accurately recall and reproduce essential factual information during inference, making it more prone to hallucinations.

In addition to pruning factual content, we strongly excluded benchmark-related material to prevent the model from memorizing evaluation datasets and artificially inflating its performance on benchmarks. While necessary for fair evaluation, this removal process may have unintentionally eliminated other important factual information closely related to the benchmark content. The boundaries between benchmark data and general factual knowledge can be blurred, and aggressive pruning might have reduced the richness of information available to the model. This deprivation of factual grounding could contribute to the model’s difficulties in generating truthful and accurate responses across various domains.

The lack of factual grounding is evident in LokiLM’s strong performance on complex reasoning tasks, such as those presented in GSM8k and MMLU, juxtaposed with its poor results on truthfulness evaluations like TruthfulQA. The models’ proficiency in higher-order reasoning suggests that it has effectively learned the underlying patterns and structures necessary for such tasks. Nevertheless, its constrained exposure to factual content undermines its overall reliability, particularly in domains requiring precise knowledge retrieval.

We conducted preliminary experiments where we increased the amount of factual data in the training set, which showed promising results in reducing the generation of inaccurate information. These findings suggest that incorporating an appropriate amount of high-quality factual content is crucial

A {lawyer, secretary} is talking to a {secretary, lawyer}. {He, She} drops a pen. {She, He} picks up the pen, walks to the bathroom, and then returns to {her, his} desk. The person who last went to the bathroom was the

Prompt				Response
lawyer	secretary	He	She/her	secretary
lawyer	secretary	She	He/his	lawyer
secretary	lawyer	He	She/her	secretary
secretary	lawyer	She	He/his	lawyer

Table 3: An ambiguous sample sentence in the form of the WinoBias dataset (Zhao et al., 2018). The orderings of the occupation and pronouns lead to four possible prompts.

for enhancing the model’s truthfulness and reliability. However, we were concerned about potential benchmark contamination, as we had already evaluated the model on these benchmarks and retraining may inadvertently lead to overfitting on evaluation benchmarks.

Furthermore, the incorporation of knowledge distillation during training, while beneficial for overall performance, may have contributed to the amplification of biases, misconceptions, or inconsistencies present in the teacher models (Chvasta et al., 2022). We used multiple teacher models to maximize knowledge transfer, but any biases present in these models may have transferred to LokiLM through distillation, exacerbating its tendency to generate false or misleading information.

In addition to its truthfulness issues, LokiLM also exhibits problematic biases and may generate toxic content in certain contexts. Using a prompt template similar to the one proposed by Kotek et al. (2023), we find that the model often relies on gender stereotypes when resolving ambiguous references (see Table 3). This suggests that LokiLM has learned and amplified biases present in its training data, a common problem among large language models (Shumailov et al., 2023).

These limitations underscore the need for continued research into methods for improving the truthfulness, consistency, and safety of generated outputs. Potential avenues for future work include the development of more robust data filtering techniques, the incorporation of explicit knowledge bases or fact-checking mechanisms, and the exploration of alternative training objectives that prioritize truthfulness alongside other desirable properties.

4 CONCLUSION

In this work, we introduced LokiLM, a 1.4B parameter language model that achieves state-of-the-art performance among models with 2B parameters or less on the Open LLM Leaderboard tasks. Our model demonstrates that through careful data curation, knowledge distillation, and architectural optimizations, it’s possible to create highly capable language models with fewer parameters and significantly less training data than competing models.

LokiLM’s strong performance, particularly on tasks like GSM8K and MMLU, highlights the potential for smaller, more efficient models to rival larger counterparts in specific domains. This advancement is particularly significant for applications requiring edge computing or deployment on consumer hardware, potentially broadening access to high-performing LLMs for research and development.

However, LokiLM’s poor performance on the TruthfulQA benchmark and its tendency to generate false or biased information underscore the ongoing challenges in developing safe and reliable language models. These limitations prevent us from releasing the model publicly and highlight critical areas for future research. Our findings emphasize the need for more robust techniques to ensure truthfulness and reduce hallucination in language models, possibly through improved data filtering or novel training objectives.

A notable challenge in LokiLM’s development was the difficulty in evaluating incremental improvements. Performance differences between architectural configurations often fell within the bounds of statistical variance, complicating the distinction between genuine advancements and random fluctuations. This highlights the need for more deterministic evaluation methodologies in the field.

432 Such techniques could enhance the model iteration process, facilitating more reliable comparisons
433 between variants and potentially accelerating language model development.

434 In conclusion, while LokiLM represents a step forward in efficient language modeling, it also
435 serves as a reminder of the complex challenges facing the field. These challenges span from ethical
436 considerations and model capabilities to the methodologies we use for evaluation and improvement.
437 By addressing these multifaceted issues and continuing to push the boundaries of what’s possible with
438 smaller models, we can work towards language technologies that are both powerful and responsible,
439 developed through increasingly rigorous and reliable processes.

441 REFERENCES

- 442
443 A. Abbas, K. Tirumala, D. Simig, S. Ganguli, and A. S. Morcos. Semdedup: Data-efficient learning
444 at web-scale through semantic deduplication, 2023.
- 445
446 J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji,
447 M. Li, J. Lin, R. Lin, D. Liu, G. Liu, C. Lu, K. Lu, J. Ma, R. Men, X. Ren, X. Ren, C. Tan, S. Tan,
448 J. Tu, P. Wang, S. Wang, W. Wang, S. Wu, B. Xu, J. Xu, A. Yang, H. Yang, J. Yang, S. Yang,
449 Y. Yao, B. Yu, H. Yuan, Z. Yuan, J. Zhang, X. Zhang, Y. Zhang, Z. Zhang, C. Zhou, J. Zhou,
450 X. Zhou, and T. Zhu. Qwen technical report, 2023.
- 451
452 E. Beeching, C. Fourrier, N. Habib, S. Han, N. Lambert, N. Rajani, O. Sanseviero, L. Tunstall, and
453 T. Wolf. Open llm leaderboard. [https://huggingface.co/spaces/HuggingFaceH4/
454 open_llm_leaderboard](https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard), 2023.
- 455
456 M. Bellagente, J. Tow, D. Mahan, D. Phung, M. Zhuravinskyi, R. Adithyan, J. Baicoianu, B. Brooks,
457 N. Cooper, A. Datta, M. Lee, E. Mostaque, M. Pieler, N. Pinnaparju, P. Rocha, H. Saini, H. Teufel,
458 N. Zanichelli, and C. Riquelme. Stable lm 2 1.6b technical report, 2024.
- 459
460 T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam,
461 G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh,
462 D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess,
463 J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models
464 are few-shot learners, 2020.
- 465
466 J. H. Cho and B. Hariharan. On the efficacy of knowledge distillation, 2019.
- 467
468 A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung,
469 C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes,
470 Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin,
471 M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski,
472 X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph,
473 A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat,
474 A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz,
475 O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm:
476 Scaling language modeling with pathways, 2022.
- 477
478 A. Chvasta, A. Lees, J. Sorensen, L. Vasserman, and N. Goyal. Lost in distillation: A case study in
479 toxicity modeling. In K. Narang, A. Mostafazadeh Davani, L. Mathias, B. Vidgen, and Z. Talat,
480 editors, *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, pages 92–
481 101, Seattle, Washington (Hybrid), July 2022. Association for Computational Linguistics. doi:
482 10.18653/v1/2022.woah-1.9. URL <https://aclanthology.org/2022.woah-1.9>.
- 483
484 P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you
485 have solved question answering? try arc, the ai2 reasoning challenge, 2018.
- 486
487 K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton,
488 R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems, 2021.
- 489
490 T. Dao. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023.

486 G. Delétang, A. Ruoss, P.-A. Duquenne, E. Catt, T. Genewein, C. Mattern, J. Grau-Moya, L. K.
487 Wenliang, M. Aitchison, L. Orseau, M. Hutter, and J. Veness. Language modeling is compression,
488 2023.

489 V. Feldman and C. Zhang. What neural networks memorize and why: Discovering the long tail via
490 influence estimation, 2020.

492 A. Forsythe. Tokenmonster. [https://github.com/alasdairforsythe/
493 tokenmonster](https://github.com/alasdairforsythe/tokenmonster), 2023.

494 Y. Gu, L. Dong, F. Wei, and M. Huang. Knowledge distillation of large language models, 2023.

496 S. Gunasekar, Y. Zhang, J. Anreja, C. C. T. Mendes, A. D. Giorno, S. Gopi, M. Javaheripi, P. Kauff-
497 mann, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, H. S. Behl, X. Wang, S. Bubeck, R. Eldan, A. T.
498 Kalai, Y. T. Lee, and Y. Li. Textbooks are all you need, 2023.

499 D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring
500 massive multitask language understanding, 2021.

502 J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. M. A. Patwary, Y. Yang,
503 and Y. Zhou. Deep learning scaling is predictable, empirically, 2017.

504 G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network, 2015.

506 J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas,
507 L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche,
508 B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre.
509 Training compute-optimal large language models, 2022.

510 A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand,
511 G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril,
512 T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023.

514 H. Kotek, R. Dockum, and D. Sun. Gender bias and stereotypes in large language models. In
515 *Proceedings of The ACM Collective Intelligence Conference*, CI '23. ACM, Nov. 2023. doi:
516 10.1145/3582269.3615599. URL <http://dx.doi.org/10.1145/3582269.3615599>.

517 Y. Li, S. Bubeck, R. Eldan, A. D. Giorno, S. Gunasekar, and Y. T. Lee. Textbooks are all you need ii:
518 phi-1.5 technical report, 2023.

519 S. Lin, J. Hilton, and O. Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2022.

521 Y. Liu, W. Zhang, and J. Wang. Adaptive multi-teacher multi-level knowledge distillation. *Neuro-
522 computing*, 415:106–113, Nov. 2020. ISSN 0925-2312. doi: 10.1016/j.neucom.2020.07.048. URL
523 <http://dx.doi.org/10.1016/j.neucom.2020.07.048>.

524 OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Al-
525 tenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu,
526 H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bog-
527 donoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button,
528 T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis,
529 D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cum-
530 mings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan,
531 S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P.
532 Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh,
533 R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hal-
534 lacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele,
535 B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin,
536 D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S.
537 Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight,
538 D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo,
539 M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin,
T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski,

540 B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan,
541 J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco,
542 E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak,
543 A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O’Keefe, J. Pachocki, A. Paino, J. Palermo,
544 A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman,
545 F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass,
546 V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh,
547 C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli,
548 T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard,
549 T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama,
550 I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang,
551 N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek,
552 J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang,
553 J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Will-
554 ner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo,
555 K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang,
556 W. Zhuk, and B. Zoph. Gpt-4 technical report, 2023.

556 Y. Oren, N. Meister, N. Chatterji, F. Ladhak, and T. B. Hashimoto. Proving test set contamination in
557 black box language models, 2023.

558 M. Ott, S. Shleifer, M. Xu, P. Goyal, Q. Duval, and V. Caggiano. Fully sharded data parallel: Faster
559 ai training with fewer gpus. Jul 2021. URL [https://engineering.fb.com/2021/07/](https://engineering.fb.com/2021/07/15/open-source/fsdp/)
560 [15/open-source/fsdp/](https://engineering.fb.com/2021/07/15/open-source/fsdp/).

561 O. Press, N. A. Smith, and M. Lewis. Train short, test long: Attention with linear biases enables
562 input length extrapolation, 2022.

563 Qualcomm. Qualcomm works with meta to enable on-device ai applications using llama 2, 2023.

564 K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. WINOGRANDE: an adversarial winograd
565 schema challenge at scale, 2019.

566 N. Shazeer. Glu variants improve transformer, 2020.

567 M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-lm: Training
568 multi-billion parameter language models using model parallelism, 2020.

569 I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. Anderson. The curse of recursion:
570 Training on generated data makes models forget, 2023.

571 S. Sievert and S. Shah. Improving the convergence of sgd through adaptive batch sizes, 2023.

572 S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le. Don’t decay the learning rate, increase the
573 batch size, 2018.

574 B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. S. Morcos. Beyond neural scaling laws:
575 beating power law scaling via data pruning, 2023.

576 G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S.
577 Kale, J. Love, P. Tafti, L. Hussenot, P. G. Sessa, A. Chowdhery, A. Roberts, A. Barua, A. Botev,
578 A. Castro-Ros, A. Slone, A. Héliou, A. Tacchetti, A. Bulanova, A. Paterson, B. Tsai, B. Shahriari,
579 C. L. Lan, C. A. Choquette-Choo, C. Crepy, D. Cer, D. Ippolito, D. Reid, E. Buchatskaya, E. Ni,
580 E. Noland, G. Yan, G. Tucker, G.-C. Muraru, G. Rozhdestvenskiy, H. Michalewski, I. Tenney,
581 I. Grishchenko, J. Austin, J. Keeling, J. Labanowski, J.-B. Lespiau, J. Stanway, J. Brennan, J. Chen,
582 J. Ferret, J. Chiu, J. Mao-Jones, K. Lee, K. Yu, K. Millican, L. L. Sjoesund, L. Lee, L. Dixon,
583 M. Reid, M. Miłkuła, M. Wirth, M. Sharman, N. Chinaev, N. Thain, O. Bachem, O. Chang,
584 O. Wahltinez, P. Bailey, P. Michel, P. Yotov, R. Chaabouni, R. Comanescu, R. Jana, R. Anil,
585 R. McIlroy, R. Liu, R. Mullins, S. L. Smith, S. Borgeaud, S. Girgin, S. Douglas, S. Pandya,
586 S. Shakeri, S. De, T. Klimenko, T. Hennigan, V. Feinberg, W. Stokowiec, Y. hui Chen, Z. Ahmed,
587 Z. Gong, T. Warkentin, L. Peran, M. Giang, C. Farabet, O. Vinyals, J. Dean, K. Kavukcuoglu,
588 D. Hassabis, Z. Ghahramani, D. Eck, J. Barral, F. Pereira, E. Collins, A. Joulin, N. Fiedel, E. Senter,
589 A. Andreev, and K. Kenealy. Gemma: Open models based on gemini research and technology,
590 2024.

594 H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal,
595 E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient
596 foundation language models, 2023a.

597 H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra,
598 P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu,
599 J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini,
600 R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A.
601 Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra,
602 I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M.
603 Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan,
604 I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and
605 T. Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b.

606 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin.
607 Attention is all you need, 2023.

608 R. Yi, L. Guo, S. Wei, A. Zhou, S. Wang, and M. Xu. Edgemoe: Fast on-device inference of
609 moe-based large language models, 2023.

610 R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. Hellaswag: Can a machine really finish
611 your sentence?, 2019.

612 A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, W. L. Tam,
613 Z. Ma, Y. Xue, J. Zhai, W. Chen, P. Zhang, Y. Dong, and J. Tang. Glm-130b: An open bilingual
614 pre-trained model, 2023.

615 B. Zhang and R. Sennrich. Root mean square layer normalization, 2019.

616 H. Zhang, J. Da, D. Lee, V. Robinson, C. Wu, W. Song, T. Zhao, P. Raja, D. Slack, Q. Lyu, S. Hendryx,
617 R. Kaplan, M. Lunati, and S. Yue. A careful examination of large language model performance on
618 grade school arithmetic, 2024a. URL <https://arxiv.org/abs/2405.00332>.

619 S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V.
620 Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang,
621 and L. Zettlemoyer. Opt: Open pre-trained transformer language models, 2022. URL <https://arxiv.org/abs/2205.01068>.

622 Z. Zhang, Y. Song, G. Yu, X. Han, Y. Lin, C. Xiao, C. Song, Z. Liu, Z. Mi, and M. Sun. Relu² wins:
623 Discovering efficient activation functions for sparse llms, 2024b.

624 J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K.-W. Chang. Gender bias in coreference resolution:
625 Evaluation and debiasing methods, 2018. URL <https://arxiv.org/abs/1804.06876>.

626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647