Language Models Are Better Than Humans at Next-token Prediction

Anonymous authors
Paper under double-blind review

Abstract

Current language models are considered to have sub-human capabilities at natural language tasks like question-answering or writing code. However, language models are not trained to perform well at these tasks; they are trained to accurately predict the next token given previous tokens in tokenized text. It is not clear whether language models are better or worse than humans at next-token prediction. To try to answer this question, we performed two distinct experiments to directly compare humans and language models on this front: one measuring top-1 accuracy and the other measuring perplexity. In both experiments, we find humans to be consistently *worse* than even relatively small language models like GPT-2-small and GPT-3-Ada at next-token prediction.

1 Introduction

Recent language models (LMs) have demonstrated impressive capabilities in natural language tasks, like writing convincing human-like text, coding, or answering general knowledge questions. However, LMs are not considered to have yet surpassed human performance at these tasks. But performance at such tasks is not a fair way of comparing LMs and humans. LMs are not explicitly trained to perform well at natural language tasks. Their loss function is simply *next-token prediction*: accurately predicting the next token given previous tokens in tokenized text.

How good are modern LMs compared to humans at next-token prediction? While one can construct tasks in which humans make next-token predictions better than any language model, there have been no "apples-to-apples" comparisons on non-handcrafted datasets. To answer this question, we performed two experiments that directly compare humans to language models on next-token prediction, using the OpenWebText dataset (Gokaslan & Cohen, 2019).

Two natural ways of measuring the quality of next-token prediction are *top-1 accuracy* and *perplexity*. Top-1 accuracy is the fraction of times, over many predictions, that the predictor assigns the highest probability to the correct next token. This is relatively easy to measure, but does not capture information about the rest of the probability distribution that the predictor assigns over possible next tokens. A more all-encompassing but harder to measure variable is perplexity: defined by 2^L where the loss L is the cross-entropy of the predictor's distribution and the true distribution over possible next tokens.

Contrary to some previous claims, we found humans to be consistently worse at next-token prediction than even small models like GPT-3-Ada (with 350 million parameters), in terms of both top-1 accuracy and perplexity. That is, even small LMs are superhuman at next-token prediction.

We structure this paper as follows. We first review claims made to date about the comparison between human and language model next-token prediction in Section 2. In sections 3 and 4 we detail two small experiments we ran to measure human top-1 accuracy and perplexity respectively, along with their results. In section 5 we discuss the implications of these results and conclude.

2 Related work

One commonly cited source on the topic of human vs LM next-token prediction is a presentation by Omohundro (2020). This presentation contains the claim that humans have a perplexity around 12, compared to the 20.5 of GPT-3 (Brown et al., 2020).

This comparison is problematic for two reasons, one small and one fatal. The smaller problem is a mismatch of text corpora: the language model statistics are word-level perplexities computed on Penn Tree Bank (PTB) (Marcus et al., 1993), while the human word-level perplexity is estimated on the 1 Billion Words (1BW) benchmark (Chelba et al., 2014). This is a small problem in practice, as while GPT-3 was not evaluated on 1BW, GPT-2 performs slightly worse on 1BW than PTB. The bigger issue is the methodology used to estimate human perplexity. The value for human perplexity is quoted from Shen et al. (2017). Here, humans were asked to rate sentences on a 0-3 scale, where 0 means "clearly inhuman" and 3 means "clearly human". Then they computed a "human judgment score": the ratio of sentences rated 3 over those rated 0. They then fit a degree-3 polynomial regression to the LMs they had (of which the best was a small LSTM), which they extrapolated significantly out of distribution to acquire the "human" perplexity (see Figure 1 from Shen et al. (2017)). Due to this far extrapolation among other problems, we don't agree with the claim that humans have perplexity 12.

Another claim is from OpenAI regarding the LAMBADA dataset (Radford et al., 2019), where they give a perplexity of around 1-2 for humans (compared to 3 for 0-shot GPT-3). However, the authors don't cite a source. The authors likely made an educated guess based on how the LAMBADA dataset was constructed. In addition, LAMBADA is a much restricted dataset, which consists of guessing single words requiring broad context. So this comparison isn't very informative to the question of how good language models are at the task they're trained on – next-token prediction on typical internet text.

The most closely analogous study to ours is that of Goldstein et al. (2022). This found that an ensemble of 50 humans have a top-1 accuracy of 28% vs 36% for GPT-2, which is similar to what we saw for humans on webtext. However, they used a different, smaller dataset (the transcript from a podcast), which is not particularly representative of randomly-sampled English internet text.

Owens et al. (1997) pitted 8 humans against an n-gram model for missing word prediction. Humans achieved a top-1 accuracy of 26%. This is related but not the same as top-1 accuracy for next-token prediction, since the prediction was on word-level rather than token level, and the predictions were conditioned on words appearing after the target word.

There are a number of more narrow datasets on which we know both human and LM performance, where some humans still outperform LMs. The MATH dataset (Hendrycks et al., 2021) is an example. But, to our knowledge, there are no apples-to-apples comparison between humans and modern LMs for webtext next-token prediction.

3 Measuring human top-1 accuracy

The main difficulty for comparing human and LM performance is that, unlike with language models, it is infeasible for humans to give their entire probability distribution for the next token in a sequence, since OpenWebText is tokenized with a vocabulary of 50,000 tokens.

One way to get around this is to simply measure top-1 accuracy by asking humans what token is most likely to come next. We can't derive human perplexity from this, but top-1 accuracy might still give us a reasonable measure of how well humans do at next-token prediction. According to this measure, humans are worse than all language models we tried, even humans who have practiced for more than an hour.

3.1 Method

The human participants were either staff or advisors of our lab, or members of the Bountied Rationality Facebook group. They were paid $30/hour. There were 60 participants overall.

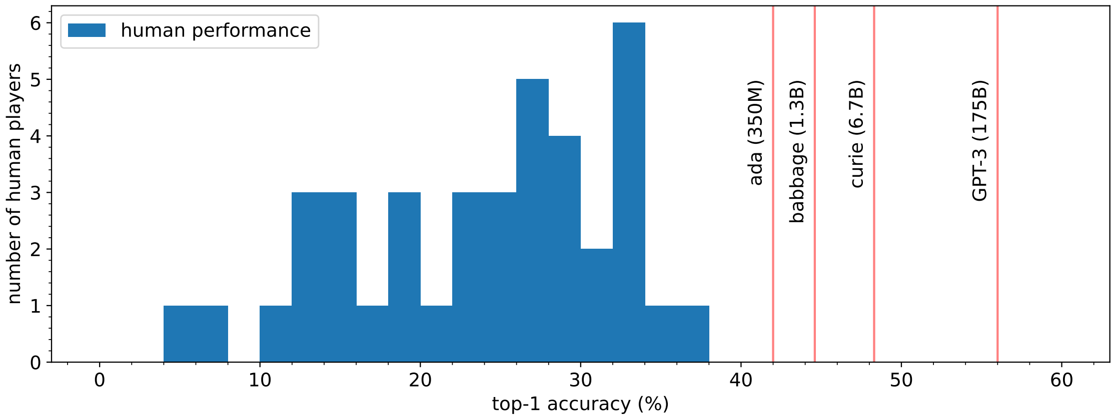

Figure 1: The distribution of human top-1 accuracy (how often a human guesses the correct next token given previous tokens) found in our study, with GPT-3 models of varying sizes for comparison. GPT-3 achieved better top-1 accuracy than all participants. Model size estimations are from Gao (2021).

The language models we evaluated were the Ada, Babbage, Curie, and Davinci models of GPT-3 (Brown et al., 2020).

We set up a top-1 token prediction game website. We recommend playing the game if you want to get a sense of what next-token prediction is like.

In this game, the player makes their way through a random segment from OpenWebText (Gokaslan & Cohen, 2019). At each token, they are asked to guess the single token they thought was most likely to come next. The correct next token is revealed to them, and they are once again asked to predict the most likely token to follow the revealed one. A total of 18530 guesses were made across participants.

Our website did not have any way for humans to guess "visually empty" tokens, such as newlines. We excluded cases where the correct guess was impossible from our analysis.

3.2 Results

The mean accuracy of the participants' answers on this top-1 task was 29%. Of these players, 38 gave at least 50 answers, with an accuracy of 30%. This accuracy is low compared to the accuracy of LMs: when measured on the same dataset, GPT-3 achieved an accuracy of 56%. Even the smallest GPT-3 model Ada, with 350M parameters, achieved an accuracy above all players in our dataset. Figure 1 shows the distribution of top-1 accuracies across participants, along with the accuracy of the GPT-3 models.

We found that humans don't quickly get much higher performance with practice. There were 7 players who guessed over 500 tokens (taking roughly 5 hours), and achieved accuracies around the average of human performance (between 0.26 and 0.32).

4 Measuring human perplexity

Language models aren't trained to guess the single most likely next token, so the previous task isn't directly assessing the language models on the task they're trained on. They are trained to give a full distribution over next tokens, aiming to minimize their perplexity. It's easy to calculate the model's perplexity, because the models produce a probability for every next token. Unfortunately, humans aren't able to quickly provide a probability distribution over the 50,000 possible tokens. So human perplexity must be estimated indirectly.

4.1 Method

To gather the relevant data, we set up a second language modeling game website. 54 humans participated. They were again either staff of our lab or members of the Bountied Rationality Facebook group, paid $15 for answering a set of 120 rounds (taking roughly 30 minutes overall per participant). 19 participants answered all 40 questions of the first set, and 11 participants answered all 80 questions of the second set, answering a total of 1640 comparisons.

On each round, a prompt c from the validation dataset of OpenWebText (Gokaslan & Cohen, 2019) is shown (with a maximum length of 120 tokens). The participant is then asked to guess the likelihood of one candidate token x being correct, given that one of the two candidates x,y are correct. From this the ratio of probabilities of x and y can be determined, which we call $r(x, y|c)$.

From the responses we build a dataset of many values of $r(x, y|c)$ for many x, y pairs. This data can be used to estimate human perplexity. We make the following assumption: for a given prompt c, all humans have the same probability distribution over next tokens $h(y|c)$. In this case, the responses from participants can be written as $r(x, y|c) = \frac{h(x|c)}{h(y|c)}$.

In the following sections we detail how we translated the dataset into an estimate of human perplexity P_h.

We used importance sampling to generate the candidate tokens (see Section 4.1.1). This way of sampling can introduce a bias to the data, so we modified the game slightly to correct for this bias (Section 4.1.2). In Section 4.1.3 we describe how we estimate an uncertainty for the results.

4.1.1 Estimating perplexity from a few relative probabilities with importance sampling

Let T be the true distribution of tokens y after a context c. Human perplexity P_h is defined as

$$P_h = 2^{L_h}, \quad L_h = -\mathbb{E}_{(C,Y)\sim T}[\log h(Y|C)]. \tag{1}$$

L_h is referred to as the human loss function. We can't directly ask a human for the probability of the true token $h(y|c)$ without spoiling them on the answer, and it would be cumbersome to ask for their whole probability distribution. We can do better by asking for relative likelihoods. For a given context c and true token y, $h(y|c)$ can be rewritten as

$$h(y|c) = \left(\sum_{x\in\Omega} \frac{h(x|c)}{h(y|c)}\right)^{-1} = \left(\sum_{x\in\Omega} r(x, y|c)\right)^{-1}, \tag{2}$$

where Ω is the full vocabulary. That's better, but that would still cost around 50,000 questions (the number of tokens) for each value of $h(x|c)$. To lower this number we use importance sampling. The bulk of $\sum_x r(x, y|c)$ is where the most likely tokens are, so it's not worth asking for every one of them. We condition a reference language model G on our context c from which we can sample the most likely tokens. We can then estimate $h(y|c)$ with the following approximation:

$$h(y|c) = \left(\mathbb{E}_{X\sim G_c}\left[\frac{r(X, y|c)}{g(X|c)}\right]\right)^{-1} \approx \left(\frac{1}{n}\sum_{x\sim G(c)} \frac{r(x, y|c)}{g(x|c)}\right)^{-1}, \tag{3}$$

where n is the number of tokens generated by G, which can be much smaller than 50,000. $g(x|c)$ is the probability of x given c outputted by G. $\sum_{x\sim G(c)}$ is the sum over the set of tokens that the model G conditioned on c generates in practice when queried n times. So in each round of the language model game, one candidate token x is the true next token, while the other, y, is generated by $G(c)$.

To decrease the variance of this estimator for $h(y|c)$, we instead estimate

$$\frac{h(y|c)}{g(y|c)} = \left(\mathbb{E}_{X\sim G_c}\left[\frac{g(y|c)}{g(X|c)} r(X, y|c)\right]\right)^{-1} \approx \left(\frac{1}{n}\sum_{x\sim G(c)} \frac{g(y|c)}{g(x|c)} r(x, y|c)\right)^{-1}. \tag{4}$$

The variance is lower because, if h and g are close, most terms in the sum will be close to 1 (whereas $r(x,y|c)/g(x|c)$ can get very large or small on some samples).

With access to $h(y|c)/g(y|c)$, we can compute $L_h - L_G$ (where L_G is the loss of the reference language model G defined by Equation 1, with h replaced with g). Using N samples from the true target corpus, from Equations equation 1 and equation 4, we find

$$L_h - L_G = \mathbb{E}_{(C,Y)\sim T}\left[\log \frac{g(Y|C)}{h(Y|C)}\right] \approx \frac{1}{N}\sum_{(c,y)\sim T} \log\left(\frac{1}{n}\sum_{x\sim G_c}\frac{g(y|c)}{g(x|c)} r(x,y|c)\right). \tag{5}$$

We can then add the known value for L_G to find L_h, and from that, P_h. In practice, we use GPT-2-small - a 12-layer language model - as our generator language model G.

4.1.2 Controlling for sample bias in importance sampling

Due to the fact that one of the two candidate tokens is sampled from a language model G, the participants no longer have an incentive to submit their actual guess at $r(x,y|c)$, since they have extra information about how x and y are sampled.

Naively, we would find $r(x,y|c) = h(x|c)/h(y|c)$ by making the human guess if the prompt c is followed by x or y, and then the human should answer that c is followed by x with probability $h(x|c)/(h(x|c)+h(y|c))$. This would be true if one token was sampled from the true distribution and the other one was selected uniformly among all other tokens. However, this isn't true here because the other token is sampled from G_c.

A rational agent that perfectly knows T and G would answer that c is followed by x with probability

$$\frac{P(x\sim T(c) \wedge y \sim G(c))}{P(x\sim T(c)\wedge y\sim G(c)) + P(x\sim G(c)\wedge y\sim T(c))} = \frac{t(x|c)g(y|c)}{t(y|c)g(x|c)+t(x|c)g(y|c)}, \tag{6}$$

where $t(x|c)$ is the true distribution T. In the second line we used the independence of T & G. A human could use their knowledge of G as extra information that can help guess the right answer. For example, a human who believes $T(c)$ to be indistinguishable from $G(c)$ will answer 0.5 to every question, making it impossible to extract $h(x|c)/h(y|c)$.

The solution is to incentivize the human to give something other than their best guess. We ask the human for the probability p that c is followed by x, and we reward them with a weighted binary cross-entropy reward

$$R(p) = g(x|c)\log(p)1_{z=x} + g(y|c)\log(1-p)1_{z=y}. \tag{7}$$

where z is the correct answer. The expected value of this reward, according to a human believing that the generative model follows a distribution $\hat{G}$, is

$$\mathbb{E}[R(p)] = \hat{g}(x|c)\log(p)(h(x|c)\hat{g}(y|c)) + \hat{g}(y|c)\log(1-p)(h(y|c)\hat{g}(x|c)). \tag{8}$$

This is at its maximum when $d\mathbb{E}[R(p)]/dp = 0$, therefore, the optimal play satisfies

$$\begin{aligned}\frac{\hat{g}(x|c)\hat{g}(y|c)h(x|c)}{p^*} &= \frac{\hat{g}(y|c)\hat{g}(x|c)h(y|c)}{1-p^*}\\ \Rightarrow p^* &= \frac{h(x|c)}{h(x|c)+h(y|c)}.\end{aligned} \tag{9}$$

If we assume that humans play optimally (which is a questionable assumption), then the human's response, p^*, represents the human's true belief for $h(x|c)/(h(x|c)+h(y|c))$, no matter what their beliefs about the generative model are.

In practice, we use a slightly different reward, $R(p) = 1000(g(x|c)(\log(p)-\log(0.5))1_{z=x} + g(y|c)(\log(1-p)-\log(0.5))1_{z=y}$, for which the optimal play is the same, but is more understandable for a human. Participants get a reward of 0 for saying $p = 0.5$, and the scaling makes their score more readable.

On rounds where $x = y$, the participant isn't asked to compare their relative likelihoods. Instead, the website automatically answers that both are as likely.

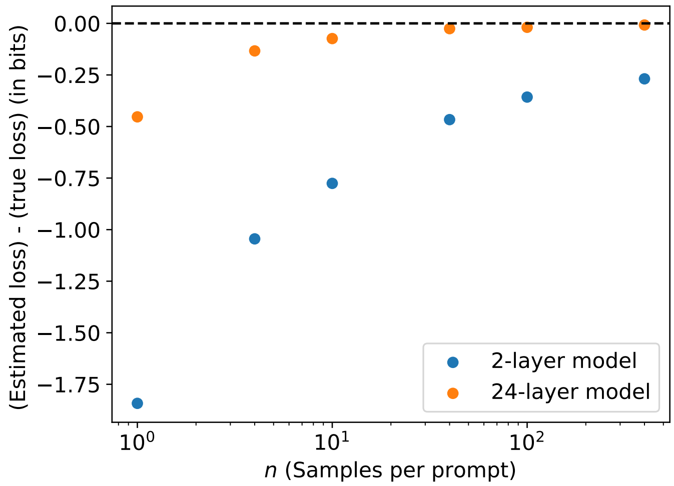

Figure 2: Difference between the loss estimated by Equation equation 5 and the ground truth loss for two LMs, for varying n (number of samples per prompt), for $N = 1000$ prompts.

4.1.3 Estimating uncertainty in our measurement of perplexity

If we assume humans play rationally in the game described above, there remain three sources of uncertainty in our measurement of human perplexity:

1. **Statistical uncertainty in the sum over $(c, y) \sim T$.** We collected data on only 120 different (c, y) pairs, which is small considering that the per-token loss has a large variance. This causes a non-negligible uncertainty over the measured perplexity, even if we had perfect estimates of each value of $h(y|c)/g(y|c)$ for each (c, y) pair. We compute the empirical standard deviation σ on L_h over (c, y) pairs. This gives us a lower bound $\exp(L_h - 2\sigma)$ and an upper bound $\exp(L_h + 2\sigma)$ on the perplexity $P_h = \exp(L_h)$.

2. **Statistical uncertainty in the sum over $x \sim G(c)$.** We now consider the uncertainty in each $h(y|c)/g(y|c)$ sample estimated by Equation equation 4. A natural way to estimate this is to use the empirical standard deviation over the sum in Equation equation 4 and propagating this uncertainty to L_h. However, as explained below, it is not necessary to include this source of uncertainty.

3. **Systematic uncertainty in the sum over $x \sim G_c$.** The sum in Equation equation 4 is heavy-tailed. This means that, since we only have a small amount of samples (≈ 10) for every prompt, we underestimate the perplexity of any predictor using this technique. The degree of underestimation cannot be easily quantified, as the weight of the heavy-tail examples depends on the distance between human predictions and the reference LM prediction, as can be seen in Figure 3. To test this effect, we estimated the loss of a 2-layer LM and a 24-layer LM, using the 12-layer LM as a generator. We find the underestimation to be at most 0.5 bits away from the ground truth when $n \approx 40$, for models that are very dissimilar (like a 2-layer model vs a 12-layer LM, for which the difference in true loss is 1.3 bits). The results are shown in Figure 2. This is a large difference, but this is still good enough for the purpose of comparing humans to language models because we find differences between humans and LMs much larger than ~ 0.5 bits. Hence we do not attempt to estimate this error, and accept it as a limitation of our study.

Uncertainties (i) & (ii) are hard to combine. However, because (ii) is small compared to (i), we chose to use only include (i) in our final uncertainties (shown as error bars in Figure 3).

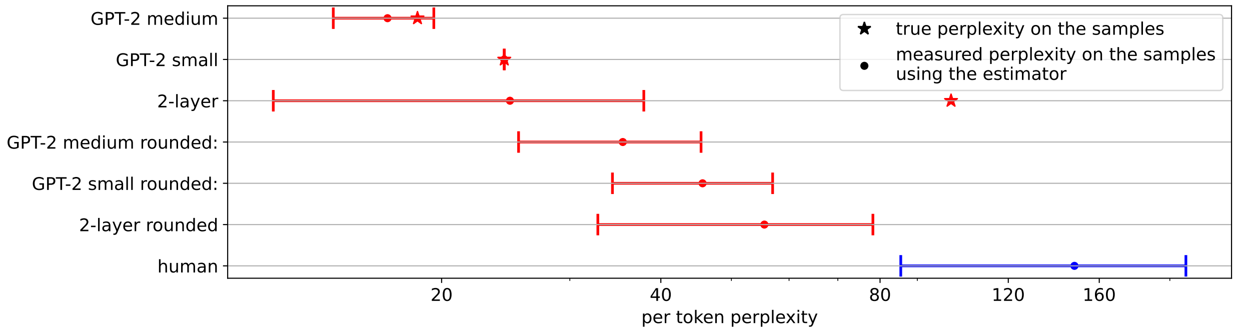

Figure 3: Our estimated perplexities for a number of language models and humans. The human perplexity was obtained from our study described in Section 4.1. GPT-2 small has no estimated value since this was used as the reference generator model. Error bars are determined according to uncertainty source (i) described in Section 4.1.3.

4.1.4 Estimating perplexity for language models

To ensure a true apples-to-apples comparison, we estimated the perplexity of the language models in the same way as described for humans above, on the same dataset. We used GPT-2 models (GPT-2 medium and small). We also used a 2-layer model we built ourselves as an example of a minimal model.

As our human participants could only enter one of the 11 ratios in our interface (99%, 90%, 80%, 70%, 60%, 50%, 40%, 30%, 20%, 10%, or 1%), we also report the "rounded" performance of our LMs - that is, the performance of our LMs if they choose the checkbox that is the closest to their probability ratio. (Note that we can't access the true perplexity of rounded models as only ratios are rounded and not the probability of the correct token.)

4.2 Results

Our estimates of perplexity for humans and LMs, along with the uncertainty estimate, are shown in Figure 3. Since we used the same method for estimating perplexity for humans and LMs, this is a direct comparison between human and LM performance. As explained in Section 4.1.3, all losses obtained by this method are underestimations of the true loss. GPT-2-small is used as the generator, which is why its measured perplexity using the estimator is perfect.

This method could also overestimate human perplexity: there could be other setups in which it would be easier for players to give calibrated probabilities. In fact, some players found the scoring system hard to understand, and if it led them to not express their true probability ratios, we might have underestimated human performance. In general, this method is very sensitive to failures at giving calibrated probability estimates: the high perplexity obtained here is probably partially due to humans being bad at giving calibrated probabilities, rather than humans just being bad at language modeling. In addition, as humans are restricted to one of 11 ratios, our setup could also underestimate performance by artificially reducing the resolution of our human participants.

Thus, while we don't have a good way to precisely measure human perplexity, these results give reasonable evidence that it is high. In particular, humans are worse than a two-layer model at giving calibrated estimates of token vs token probabilities.

5 Discussion

The results here suggest that humans are worse than even small language models the size of GPT-2-small (117M parameters) at next-token prediction, even on the top-1 prediction task. This is true even when the humans have practiced for 1-2 hours. Some humans may beat GPT-2-small with enough practice, but not substantially larger models.

These results may be surprising because humans are better at writing coherent text than GPT-2 and therefore one might expect humans to be better at next-token prediction. But actually, these tasks are very different–if you train an autoregressive model to imitate human text, the model has to dedicate capacity to all the different features that might be informative for guessing the next token (including features that don't affect human judgments of coherence). Language models try to guess all possible reasonable ways a text could continue, while humans usually one have to find one reasonable completion.

What should we take away from this?

- Even current large language models are superhuman at language modeling. This is important for language model interpretability work, because it means one should expect a model to have more knowledge of the text than you have. It has been argued in Hurbinger (2019) that models may become more interpretable as they get to human level, and then become less interpretable again as they become superhuman. The fact that existing LMs are already superhuman (at the task they're trained on) is worth bearing in mind here.
- Next-token prediction is not just about understanding the world; it's also substantially about guessing sentence structure and word choice. This isn't actually a useful ability for models to have for most applications. Next-token prediction is probably much less efficient than other tasks at training a competent or useful language model per bit of training data. But data for next-token prediction is so cheap that it has proven to be the best pretraining task anyway.

6 Conclusion

In this paper, we compared the capabilities of humans versus language models (LMs) in next-token prediction tasks, using top-1 accuracy and perplexity as our metrics. We find that that even small language models outperform humans at next-token prediction.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, T. Brants, Phillip Todd Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. In *INTERSPEECH*, 2014.

Leo Gao. On the sizes of openai api models, 2021. URL `https://blog.eleuther.ai/gpt3-model-sizes/`.

A. Gokaslan and V. Cohen. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Ariel Goldstein, Zaid Zada, Eliav Buchnik, Mariano Schain, Amy Price, Bobbi Aubrey, Samuel Nastase, Amir Feder, Dotan Emanuel, Alon Cohen, Aren Jansen, Harshvardhan Gazula, Gina Choe, Aditi Rao, Catherine Kim, Colton Casto, Lora Fanda, Werner Doyle, Daniel Friedman, and Uri Hasson. Shared computational principles for language processing in humans and deep language models. *Nature Neuroscience*, 25:369–380, 03 2022. doi: 10.1038/s41593-022-01026-4.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *ArXiv*, abs/2103.03874, 2021.

E Hurbinger. Chris olah's views on agi safety, 2019. URL `https://www.alignmentforum.org/posts/X2i9dQQK3gETCyqh2/chris-olah-s-views-on-agi-safety`.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, jun 1993. ISSN 0891-2017.

Steve Omohundro. Openai's gpt-3 language model, 2020. URL `https://www.slideshare.net/numenta/openais-gpt-3-language-model-guest-steve-omohundro`.

M. Owens, P. O'Boyle, J. McMahon, J. Ming, and F. J. Smith. A comparison of human and statistical language model performance using missing-word tests. *Language and Speech*, 40(4):377–389, 1997. doi: 10.1177/002383099704000404. URL `https://doi.org/10.1177/002383099704000404`.

Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications, 2019. URL `https://openai.com/blog/better-language-models/`.

Xiaoyu Shen, Youssef Oualil, Clayton Greenberg, Mittul Singh, and Dietrich Klakow. Estimation of gap between current language models and human performance. In *INTERSPEECH*, 2017.

A Data collection interface

Figure 4 and 5 show the interface participants used.

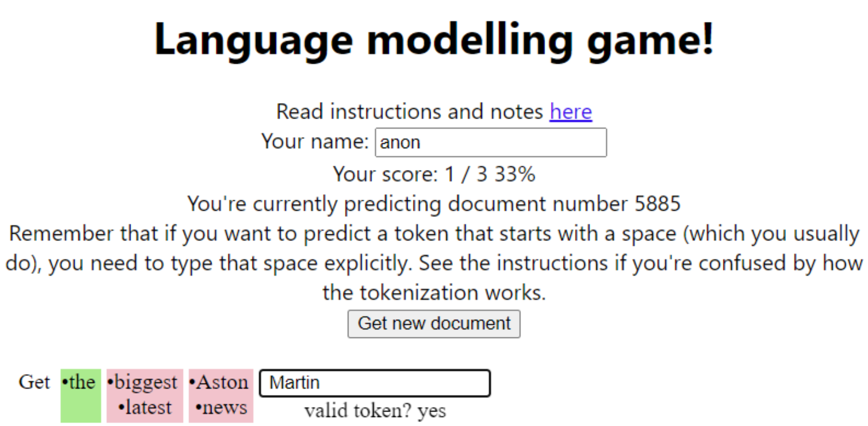

Figure 4: Interface for the experiment describe in Section 3. The interface is available at `https://rr-lm-game.herokuapp.com`

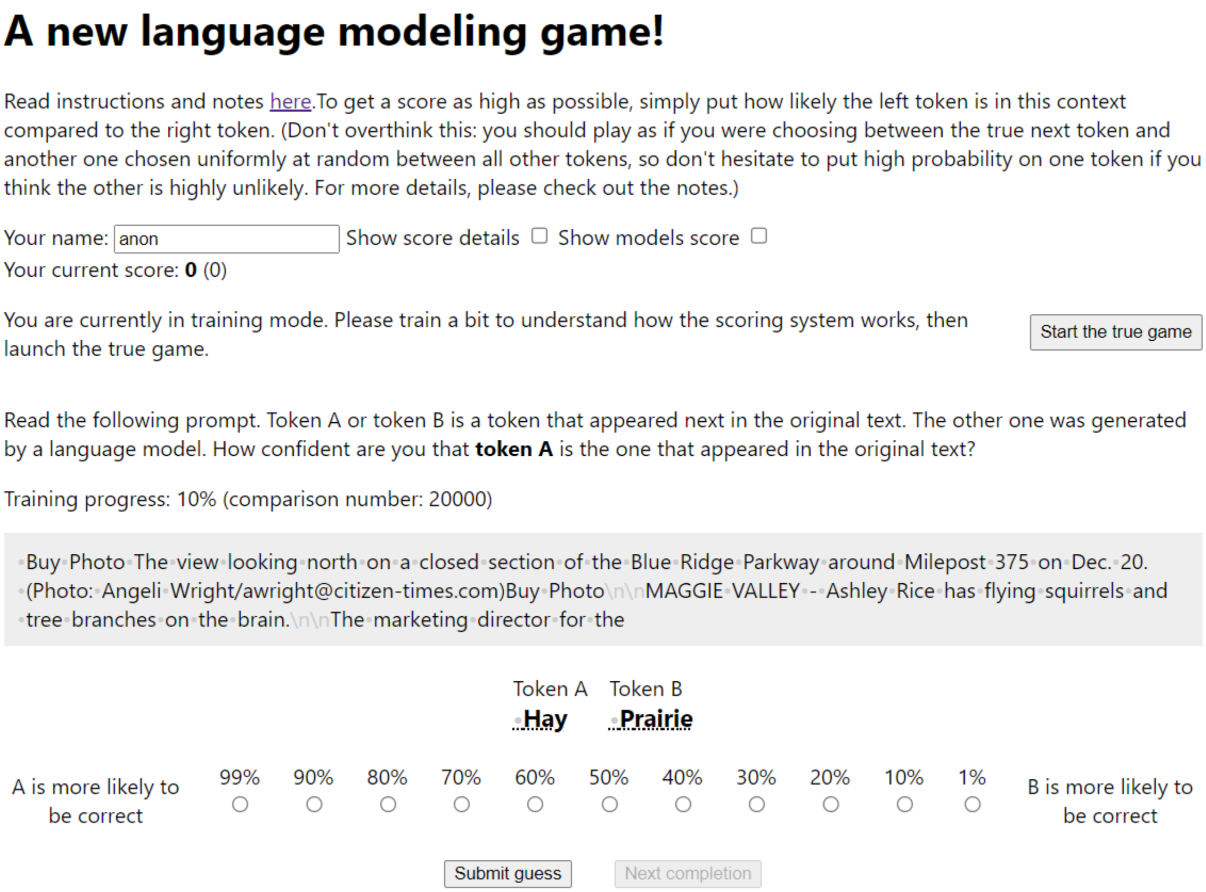

Figure 5: Interface for the experiment describe in Section 4. The interface is available at `https://rr-lm-game.herokuapp.com/whichonescored`.