Do Massively Pretrained Language Models Make Better Storytellers?

Abigail See, Aneesh Pappu,[*] **Rohun Saxena,**[*] **Akhila Yerukola,**[*] **Christopher D. Manning**
Stanford University
`{abisee,apappu,rohun,akhilay,manning}@cs.stanford.edu`

Abstract

Large neural language models trained on massive amounts of text have emerged as a formidable strategy for Natural Language Understanding tasks. However, the strength of these models as Natural Language *Generators* is less clear. Though anecdotal evidence suggests that these models generate better quality text, there has been no detailed study characterizing their generation abilities. In this work, we compare the performance of an extensively pretrained model, OpenAI GPT2-117 (Radford et al., 2019), to a state-of-the-art neural story generation model (Fan et al., 2018). By evaluating the generated text across a wide variety of automatic metrics, we characterize the ways in which pretrained models do, and do not, make better storytellers. We find that although GPT2-117 conditions more strongly on context, is more sensitive to ordering of events, and uses more unusual words, it is just as likely to produce repetitive and under-diverse text when using likelihood-maximizing decoding algorithms.

1 Introduction

In 2018, large-scale neural models such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019) and OpenAI GPT (Radford et al., 2018) emerged as a dominant approach in NLP. By pretraining on massive amounts of unlabeled text (often orders of magnitude larger than the the target task's labeled dataset), these models achieve state-of-the-art performance across a variety of Natural Language Understanding benchmarks. In particular, the OpenAI GPT2 language model (Radford et al., 2019) achieves state-of-the-art performance on several language modeling benchmarks, even in a zero-shot setting. While GPT2's performance as a language model is undeniable, its performance as a text generator is much less clear.

Though the model has generated certain impressive samples of text – such as a widely-circulated passage about Ovid's Unicorn (Radford et al., 2019) – there has been no detailed study to formalize these observations.

In this work, we perform an in-depth study of the properties of text generated by GPT2-117 (the smallest version of GPT2) in the context of story generation. By comparing to a state-of-the-art, specialized-architecture neural story generation model (Fan et al., 2018), we ask the following questions. In what ways does a large amount of open-domain pretraining data change the characteristics of generated text? In what ways does it make no difference? And is a task-specific architecture necessary?

For any probabilistic language model, the generated text is strongly affected by the choice of decoding algorithm – this is especially true for *open-ended* text generation tasks such as storytelling and chitchat dialogue (Kulikov et al., 2018; Holtzman et al., 2019). Nevertheless, most natural language generation papers evaluate only *one* decoding algorithm – this is often due to the time and expense required for human evaluation. For example, Fan et al. use top-k sampling (a decoding algorithm in which k governs the quality-diversity tradeoff), but only evaluate one value of k. However, evaluating one k gives an incomplete view of the generation system – several researchers have emphasized the importance of evaluating generation systems over the entire quality-diversity spectrum, rather than a single point on it (Caccia et al., 2018; Hashimoto et al., 2019).

In this study, we prioritize evaluating text across the *whole k* spectrum, and measuring *many* different automatic metrics, rather than a few human metrics. Though the lack of human evaluation limits our ability to measure overall quality (Liu et al., 2016; Novikova et al., 2017; Hashimoto

---

[*]equal contribution

et al., 2019), we are able to produce an objectively defined, richly detailed and reproducible evaluation of the generated text. To our knowledge, this work is the first comprehensive analysis of the characteristics of GPT2-generated text. Our study provides insight into the effect of large-scale pre-training on open-ended natural language generation, as well as the effect of k on text generated with top-k sampling. We hope our results will inform other researchers' choice of models, pretraining schemes, and decoding algorithms – decisions that can often feel like blind choices. To enable readers to browse the generated text, conduct their own evaluations, or run our evaluations on their own text, we publicly release our generated stories and evaluation code.[1]

2 Background

WritingPrompts dataset WritingPrompts (Fan et al., 2018) is a story generation dataset containing 303,358 human-written (*prompt*, *story*) pairs collected from the /r/WritingPrompts subreddit – a forum where Reddit users compose short stories inspired by other users' prompts. An example can be seen at the top of Table 2. The mean prompt length is 28.4 words and the mean story length is 734.5 words. The dataset is 887MB of text in total, contains 200 million story words, and is divided into 90% train, 5% validation and 5% test splits.

The Fusion Model The Fusion Model is a state-of-the-art neural story generation architecture trained on the WritingPrompts dataset (Fan et al., 2018). It is based on the Convolutional Seq2seq model of Gehring et al. (2017) and aims to improve two aspects of story generation: modeling long-range context and increasing relevance of the story to the prompt. To achieve the former, the model uses a multi-scale gated self-attention mechanism. For the latter, the model uses a fusion mechanism (Sriram et al., 2018) in which one seq2seq model is trained on the task, then frozen, and a second seq2seq model is trained on the task with access to the first model's hidden states. Compared to the Convolutional Seq2seq model and other baselines, the Fusion Model achieves improved perplexity, story-prompt relevance and human preference scores. The Fusion Model has a vocabulary of 104,960 words, a 3-layer encoder and 8-layer decoder in the first seq2seq model, and a 5-layer encoder and 5-layer decoder in the second model – in total, 255.4 million parameters.

GPT2-117 GPT2 (Radford et al., 2019) is a large Transformer language model trained on WebText, a diverse corpus of internet text (not publicly released) containing over 8 million documents equalling 40GB of text in total. The full-size GPT2 model, which has 1542 million parameters, obtains state-of-the-art results on a variety of language modeling and other Natural Language Understanding benchmarks. At the time of our experiments, Radford et al. had only released the smallest of the models, known as GPT2-117.[2] This model, which we use for our experiments, has 12 layers and 117 million parameters. Like the full-size GPT2 model, it has a vocabulary of 50,257 byte-pair-encoding (BPE) tokens. The BPE encoding allows the model to encode and generate any Unicode string, regardless of pre-processing, tokenization, or vocabulary size. The model has a context size of 1024, meaning it can process text up to 1024 BPE tokens in length.

Decoding algorithms Inspired by Neural Machine Translation, most early attempts at open-ended neural text generation (such as conversational response generation) used the *beam search* decoding algorithm (Shang et al., 2015; Serban et al., 2016). Like greedy decoding, beam search is a *likelihood-maximizing* decoding algorithm – given the input sequence x, the objective is to find an output sequence y which maximizes $P(y|x)$. However, researchers have shown that for open-ended generation tasks (including storytelling), beam search produces repetitive, generic and degenerate text (Holtzman et al., 2019).

More recently, *top-k sampling* has emerged as a primary decoding algorithm for open-ended text generation (Fan et al., 2018; Radford et al., 2019). In top-k sampling, on each step of the decoder the probability distribution over the vocabulary is truncated to the top k tokens, then re-normalized. The next token is sampled from the new distribution. Top-k sampling can be regarded as somewhere between a likelihood maximizing algorithm (when $k = 1$; greedy decoding) and an unbiased sampling algorithm (when $k =$ vocabulary size). Fan et al. use top-k sampling (with $k = 10$) to

---

[1]Code and generated stories available at `https://github.com/abisee/story-generation-eval`

[2]Since conducting our experiments, larger models have been publicly released. At the time of writing, the full-size GPT2 model has not been publicly released.

generate stories, and Radford et al. show impressive samples of generated text (primarily from the full-size GPT2 model) for $k = 40$.

3 Experimental Details

Preprocessing Fan et al. truncate Writing-Prompts stories to 1000 words before training and testing. Due to the limited context size of GPT2-117, we additionally exclude (*prompt*, *story*) examples that are longer than 1024 BPE tokens when concatenated. The resulting dataset, which we call WritingPrompts-1024, has 192,364 training, 11,115 validation, and 10,686 test examples.

The Fusion Model We use the pretrained version of the Fusion Model, which is available in the Fairseq framework (Ott et al., 2019). For comparability with GPT2-117, we evaluate the Fusion Model on WritingPrompts-1024 (see Table 1), obtaining perplexities similar to those reported by Fan et al. on the full WritingPrompts dataset.

GPT2-117 In order for the model to condition on prompts and generate stylistically correct stories, we finetune GPT2-117 on WritingPrompts-1024.[3] We frame WritingPrompts as a language modeling task, representing the prompt and story as a single sequence separated by a delimiter token. We finetune the pretrained model until convergence using the default hyperparameters provided in the HuggingFace repository (though we reduce batch size to fit on a single GPU), and use the finetuned model for all further evaluations.

We compute the *word-level* perplexity of the finetuned GPT2-117 on the WritingPrompts-1024 dataset. That is, we normalize the total negative log probability of the target text by the number of *word*-level (i.e. Fusion Model) tokens, not the number of BPE tokens. This enables us to compare the perplexities of the two models, despite the tokenization difference (Radford et al., 2019). The finetuned GPT2-117 obtains a test set word-perplexity of 31.54[4] – six points lower than the Fusion Model.

Generation settings For both models, we generate stories using top-k sampling, obtaining 1000 stories (from 1000 different test set prompts) for

Model	Valid ppl	Test ppl
Fusion Model	37.05	37.54
GPT2-117	31.13	31.54

Table 1: Word-level perplexities on WritingPrompts-1024 for the Fusion Model and finetuned GPT2-117.

several values of k ranging from 1 to vocabulary size. We use softmax temperature 1. Like Fan et al., we generate exactly 150-word stories and block the Fusion Model from generating `<UNK>`.

To obtain human-written stories for comparison, we truncate WritingPrompts-1024 test set stories to 150 words (discarding those shorter than 150 words). To reduce variance, measurements for human stories are computed over this entire set (rather than just 1000 stories).

4 Story-prompt relatedness

Prior research has observed that seq2seq systems frequently produce text that is unrelated to the provided context – particularly under likelihood-maximizing decoding algorithms such as beam search. The issue has inspired multiple explanations (Jiang and de Rijke, 2018) and multiple solutions – such as alternative training objectives (Li et al., 2016), decoding objectives (Baheti et al., 2018; See et al., 2019), and architectural changes (Fan et al., 2018). In this section, we measure how strongly the models condition on the prompt.

Prompt ranking accuracy For both models, we compute *prompt ranking accuracy* (Fan et al., 2018), which measures the language model's sensitivity to the provided prompt. Following the methodology of Fan et al., we randomly select 1000 human-written stories from the test set, and measure the probability (according to the model) of each story conditioned on 10 different prompts – the true prompt, plus nine randomly selected prompts. The prompt ranking accuracy of a model is the percentage of cases in which the model assigns a higher probability to the story under its true prompt than under all of the other nine. We find that GPT2-117 scores **80.16%** on this task, while the Fusion Model scores **39.8%**.[5] Random chance scores 10%. This striking result indicates

---

[3]We use the PyTorch re-implementation of GPT2-117 available at `https://github.com/huggingface/pytorch-transformers`

[4]This is similar to other GPT2-117 WritingPrompts finetuning experiments (Mao et al., 2019; Ziegler et al., 2019).

[5]Fan et al. (2018) report a prompt ranking accuracy of 16.3% for the Fusion Model. We provided the authors with our prompt ranking accuracy code (which was built on top of the authors' code). The authors indicated that the discrepancy may be due to some code version changes between the time of their original experiments and their code release.
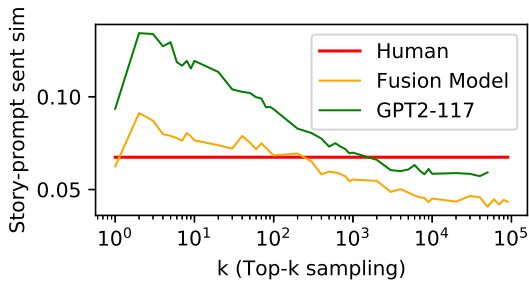
Figure 1: Compared to the Fusion Model, GPT2-117 produces stories that are more semantically similar to the prompt. Similarity decreases as k increases.

that GPT2-117 conditions on the prompt much more strongly than the Fusion Model. This is notable, especially because the fusion technique is intended to improve story-prompt relevance.

N-gram similarity For $n = 1, 2, 3$, we measure the percentage of generated n-grams that also appear in the prompt. For all n and k, we find that GPT2-117 has a higher overlap (i.e. copies more from the prompt) than the Fusion Model – see Figure 6 in the Appendix. Furthermore, for $k < 100$, the GPT2-117 overlap is generally much higher than human levels. Both these phenomena can be seen in Table 2, where, for $k = 10$, GPT2-117 copies words such as *queen* more often than both the Fusion Model and the human-written story.

Sentence embedding similarity To capture a higher-level notion of semantic similarity, we measure *story-prompt sentence similarity* – the cosine similarity of story-prompt sentence pairs, averaged by taking the mean over all pairs. Sentences are represented by the embedding method of Arora et al. (2017) – a weighted average of the GloVe embeddings (Pennington et al., 2014) of the words, with the first principal component removed. As shown in Figure 1, we find a similar pattern as for n-gram similarity: GPT2-117 generates sentences that are more similar to the prompt than the Fusion Model for all k, and both models' prompt similarity decreases as k increases.

Named entity usage Generally, most named entities mentioned in the prompt (such as *Queen* and *England* in Table 2), should also be mentioned in the story. Using the spaCy named entity recognizer,[6] we measure the *prompt entity usage rate*, which is the percentage of all prompt named enti-

ties that appear in the story.[7] As shown in Figure 7 in the Appendix, we find that GPT2-117 uses more of the prompt named entities than the Fusion Model (as well as more named entities overall), but both models use fewer named entities than humans when k is less than vocabulary size.

These patterns can be seen in Table 2: GPT2-117 uses the prompt entities *Queen* and *England* whereas the Fusion Model does not (for either k), and GPT2-117 uses specific time entities such as *Thursday* and *3:26 PM*. While the human story introduces highly-related entities such as *Charles Windsor* and *Prince of Wales* that were not in the prompt, neither model does this (for either k).

Conclusion In this section, we found that GPT2-117 conditions on the prompt much more strongly than the Fusion Model – a result which holds both in language modeling and generation settings. The latter result supports Radford et al.'s informal observation that GPT2 has a 'chameleon-like' ability to 'adapt to the style and content of the conditioning text'.[8] We speculate that GPT2-117's stronger conditioning ability may derive from its Transformer decoder architecture, whose powerful self-attention is used for story-prompt attention. Though the Fusion Model uses a similar self-attention mechanism in the decoder (i.e., story side), the prompt-story attention has a simpler formulation – for example, there are no separate key and value vectors (Gehring et al., 2017). Lastly, we note that very strong prompt-conditioning is not always a good thing – GPT2-117 often generates stories that copy too much or too literally from the prompt when k is small (this can be seen in Figure 6 in the Appendix).

5 Coherence

A good story generation model should produce coherent text with a logical ordering of events. Similarly, the underlying language model should be a good coherence scorer – assigning higher probability to coherent text than incoherent text. Barzilay and Lapata (2008) evaluate a coherence scorer by measuring its ability to rank shuffled human-written text as less coherent than the original unshuffled text. We use this method to evaluate our story generation models.

---

[6] https://spacy.io

[7] Given that we limit stories to 150 words, this percentage is lower than it would be if we generated longer stories.
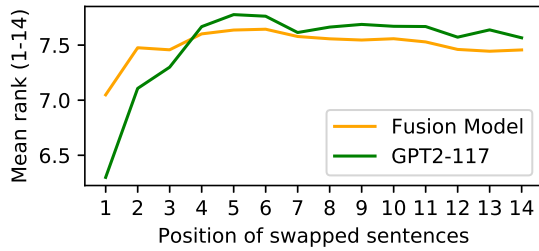
[8] https://openai.com/blog/better-language-models/

Figure 2: Sensitivity of the models to swapped sentences in different positions. A higher mean rank indicates higher sensitivity (i.e. the model assigns lower probability) relative to other positions. Both models are less sensitive to swapped sentences at the beginning of the text, compared to later. GPT2-117 shows this pattern more strongly, indicating greater use of context.

For each story in the test set, we select the first 15 sentences. We then produce 14 corrupted versions of the story by switching each pair of adjacent sentences. We use the language model to compute the probability of each of the 14 corrupted stories, as well as the original story. The model's error rate is the percentage of cases in which it rates any of the 14 corrupted candidates better than the original candidate. Random guessing yields 93.33% error. Both models perform well on this task – the Fusion Model has an error rate of **3.44%** and GPT2-117 an error rate of **2.17%**. This 36.92% error reduction indicates that GPT2-117 is more sensitive to ordering of events.

We also investigate how the position of the swap affects its plausibility (relative to other positions). Figure 2 shows, for each swap position, the mean rank assigned to that swap by the model (where rank 1 is the most probable of the 14 corrupted candidates, and rank 14 the least probable). GPT2-117 assigns a much lower rank to the first few swap positions (i.e., rates them more probable) than the later positions. The Fusion Model shows a similar but less pronounced pattern. This shows that both models are less sensitive to out-of-order sentences that occur at the beginning of the text, than those occurring later.[9] The stronger pattern for GPT2-117 may be due to its stronger context conditioning (as shown in Section 4) – thus becoming more sensitive as context increases. However, even for the first three swaps, GPT2-117 is more accurate than the Fusion Model at distinguishing the swapped text from the original.

---

[9] It's also possible that out-of-order sentences are inherently harder to detect at the beginning of text.
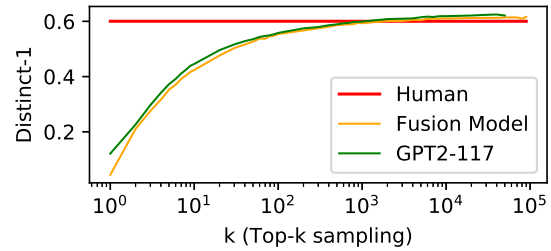

Figure 3: Repetition (low distinct-1) is primarily caused by choice of decoding algorithm (here low k), not insufficient training data. GPT2-117 is trained on $45\times$ more data than the Fusion Model, but is similarly repetitive for all k.

6   Repetition and rareness

Generic, under-diverse and repetitive text is a well-documented problem in neural text generation (Jiang and de Rijke, 2018). While there are many proposed solutions to the problem (Li and Jurafsky, 2016; Vijayakumar et al., 2018; Baheti et al., 2018; Zhang et al., 2018; See et al., 2019), it has been shown that a primary cause is likelihood-maximizing decoding algorithms such as greedy decoding and beam search (Holtzman et al., 2019). In this section we investigate the role of large-scale pretraining, and the role of k, in this problem.

N-gram repetition   The *distinct-n* metric of a piece of text is the number of unique n-grams divided by the total number of generated n-grams (Li et al., 2016). We measure distinct-n of the generated stories for $n = 1, 2, 3$. A high ratio indicates a high level of within-story lexical diversity, while a low ratio indicates a large amount of within-story repetition. As shown in Figure 3, both models' unigram diversity is far below that of human text when k is small. For example, at $k = 10$ (the setting used by Fan et al.), the Fusion Model obtains a distinct-1 of 42.4%; much less than the human level of 60.0%. This results in a high level of repetition, as shown in Table 2: for $k = 10$, both models repeat many phrases (such as *always*, *so scared*, and *finally*).

For bigrams and trigrams, the pattern is similar to unigrams (see Figure 9 in the Appendix). For both models, distinct-n increases as k increases, converging to a value close to the human level as k approaches vocabulary size. Though GPT2-117 has a slightly higher distinct-n than the Fusion Model for most values of k, the difference is negligible compared to the influence of k. We

make three conclusions from these patterns: (1) Our findings support Holtzman et al.'s observation that repetition is strongly related to choice of decoding algorithm, and that likelihood maximizing algorithms (such as top-k sampling with low k) are a primary cause of repetition. (2) The models have in fact learned the correct rate of repetition in human text – they are able to match this rate when they sample from their full (untruncated) distribution. (3) Repetition is unlikely to be solved by more pretraining data alone – even though GPT2-117 is trained on 45 times as much data as the Fusion Model, it produces text that is almost equally repetitive (for equal k).

Rare word usage We compute the *mean log unigram probability* of the words in the generated story[10] – a high value indicates using fewer rare words while a low value indicates more rare words. As shown in Figure 12 in the Appendix, word rareness is primarily governed by k – however, GPT2-117 has a lower mean log unigram probability (i.e., uses more rare words) than the Fusion Model for all equal values of $k \geq 2$. This can be seen for example in Table 2, where GPT2-117 generates rarer words such as *idle* and *copious* for $k = 1000$. GPT2-117 also generates fewer stopwords than the Fusion Model, for all equal k.

GPT2-117's slightly higher rare word usage (compared to the Fusion Model) might be explained by: (1) its BPE encoding, which allows it to generate new words, not just those in a fixed vocabulary; (2) pretraining on a large amount of diverse text, allowing it to learn to produce a greater variety of words; (3) stronger conditioning on the prompt as described in Section 4 – which may inject more rareness into the generated text.

Conclusion Choice of decoding algorithm is a primary factor in diversity and repetition problems, with likelihood-maximizing algorithms the main culprit. Although GPT2-117 generates more rare words and is very slightly less repetitive than the Fusion Model, the difference is small compared to the effect of k, indicating that training data alone is unlikely to solve these problems.

7 Syntactic style and complexity

A well-trained story generation model should match both the syntactic style and complexity of

its training data. Low complexity can be a sign of less sophisticated writing, while high complexity can be a sign of poor readability (Beers and Nagy, 2009; McNamara et al., 2010). In this section, we measure some features related to the syntactic style and complexity of the generated stories.

Sentence length Sentence length is a simple but effective feature to estimate readability and syntactic complexity of text (Kincaid et al., 1975; Roemmele et al., 2017). We find that both models generate sentences that are on average shorter than human sentences when k is small, but converge to approximately human length as k increases (see Figure 8 in the Appendix).

Part-of-speech usage It has been shown that the distribution of parts-of-speech (POS), and more generally the distribution of POS n-grams[11] is a useful feature to represent the style of a piece of text (Argamon et al., 1998; Ireland and Pennebaker, 2010; Roemmele et al., 2017).

Firstly, we compare the part-of-speech distributions of the model-generated text and the human text (see Figure 11 in the Appendix). Both models (especially GPT2-117) closely fit the human POS distribution as k approaches vocabulary size.[12] This implies that, as with lexical diversity, the models have no difficulty fitting the statistical distribution of human syntax. However, under likelihood-maximizing decoding algorithms such as low k, a completely different distribution emerges, in which text contains more verbs and pronouns than human text, and fewer nouns, adjectives and proper nouns.

Secondly, we measure the syntactic diversity of the text using the distinct-n metric for POS n-grams ($n = 1, 2, 3$) – see Figure 10 in the Appendix. As with lexical diversity (see Section 6), we find that syntactic diversity is similar for the two models, is very low when k is small, and matches human level as k approaches vocabulary size. It's likely that for low k, the syntactic under-diversity of the text is largely caused by lexical under-diversity (i.e. repetition). However, we note that as k increases, lexical diversity reaches human level sooner than syntactic diversity – for example, GPT2-117's lexical distinct-3 reaches human level at $k = 600$ (Figure 9c), but its POS distinct-

---

[10]The unigram probability distribution was calculated with respect to the WritingPrompts training set.

[11]For example, the sentence *I like cats* has the POS bigrams PRONOUN VERB and VERB NOUN.

[12]One exception is Proper Noun: both models fail to produce enough of these even as k approaches vocabulary size.
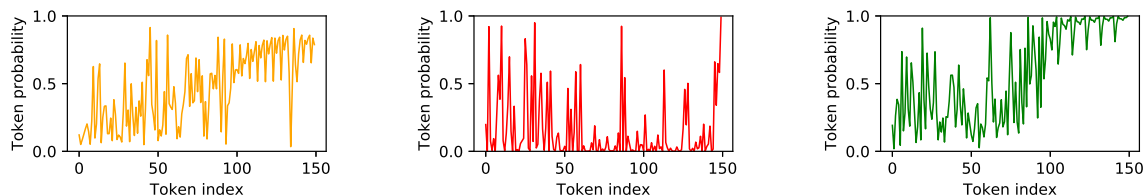
(a) **Fusion Model** ($k = 2$): *I had never seen a man so young before. I had never seen him before, but he had always seemed to be a man of a man. He was young, and he was young. He was a man of a man, and a man who was young, and a man who was* [...]

(b) **Human Text**: *"Looks like the rain's stopped." I peered out the window. Art was right; time to get to work. "Alright, let's move out." I could hear the scraping of the stone armor as the men slowly stood. Despite the training,* [...]

(c) **GPT2-117** ($k = 2$): *I've always been a man of the people. I've always been a strong man. I've always been a strong man. I was born in the city, I was raised in the country. I was raised in a family that wasn't very good. I'm not a good man.* [...]

Figure 4: Under top-k sampling with small k ($k = 2$), the two models (left and right) produce text that falls into increasingly confident repeating loops. By contrast, human text (center) maintains an irregular pattern of surprising (low probability) tokens. The human text probabilities are measured with respect to the Fusion Model, but similar patterns hold for GPT2-117. Inspired by Holtzman et al. 2019's figure showing probabilities under beam search.

3 reaches human level at $k = 6000$ (Figure 10c). This implies that, even when the text is no more repetitive than human text, it may still be syntactically repetitive (using the same part-of-speech patterns repeatedly).

Conclusion We find when k is small, syntactic complexity of generated text is low, consisting of shorter sentences and a narrower range of syntactic patterns. However, as k approaches vocabulary size, the syntactic style of generated text closely matches human syntactic patterns. As with n-gram diversity in Section 6, our results show that syntactic under-diversity is primarily caused by low k, not insufficient training data.

8 The element of surprise

Model confidence over time Several researchers have observed that *model over-confidence* (the model placing high probability on a small range of tokens) can cause poor quality generation (Jiang and de Rijke, 2018; Holtzman et al., 2019). In particular, they show that for likelihood-maximizing decoding algorithms such as beam search, model confidence can increase in a snowball-like effect, getting stuck in a loop of repetitive but increasingly self-confident text. We observe this problem in both our models when k is small. For example, in Figure 4, both models fall into self-reinforcing repetitive loops with rising confidence. The loop is difficult to break – the Fusion Model briefly escapes (shown as a sudden downwards spike), but quickly returns. By contrast, the human text does not show a strong rising trend in probability, and intermittently uses low probability words throughout.[13]

We formalize these anecdotal observations by measuring the average probability of each of the first 150 word-level tokens in the story (Figure 5). We find that even when teacher-forcing on human text, the token probabilities increase slightly as the story progresses. This is likely due to the usefulness of additional context, which increases the model's prediction accuracy. By comparison, we find that when generating with top-k sampling, the probabilities increase more rapidly, and the increase is even more rapid for smaller k. This confirms that likelihood-maximizing decoding algorithms (such as top-k sampling with small k) lead to more rapidly increasing model over-confidence. Furthermore, we find this pattern holds for both models, with probabilities increasing at a similar rate for equal k. This indicates that, like repetition, model over-confidence is unlikely to be solved by more training data, and is largely governed by choice of k.

Overall model confidence We also measure the models' overall confidence, as represented by the total log probability (according to the model) of the generated story. For both models, we find that story probability decreases as k increases – see Figure 13 in the Appendix. This makes sense, as higher k means sampling tokens with lower probability. As k approaches the vocabulary size, the Fusion Model's generated story

---

[13]Gehrmann et al. (2019) also identify presence of low probability words as an indicator of human-generated text.
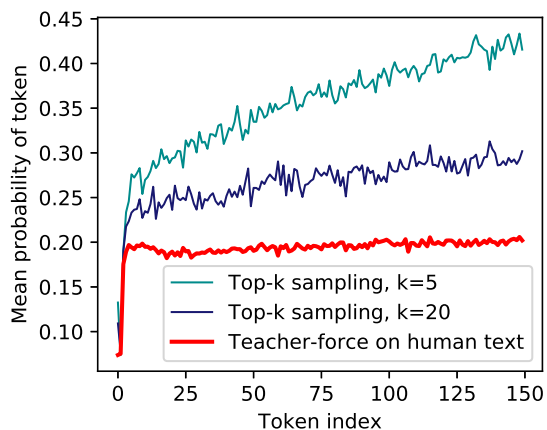
849

Figure 5: Mean probability for each of the first 150 word-level story tokens. When teacher-forcing the model on human text, probability increases slowly. When generating with top-k sampling, probability increases faster, especially for smaller k. This plot is for the Fusion Model; similar patterns hold for GPT2-117.

probability matches the probability it assigns to human-written WritingPrompts stories. Interestingly however, the same is not true for GPT2-117, which converges to a story probability that is *lower* than the probability it assigns the human stories. This means that under full (non-truncated) sampling, the Fusion Model produces text that is *equally surprising* (to itself) as the Writing-Prompts stories, whereas GPT2-117 produces text that is *more surprising* to itself. Explaining this observation is an open question – we speculate that GPT2-117's WebText pretraining may cause it to generate (under high k) text in a style or genre that is less predictable than WritingPrompts stories.

9 Concreteness

Brysbaert et al. (2014) define the *concreteness* of a word as 'the degree to which the concept denoted by a word refers to a perceptible entity'. Concrete words are generally easier to remember than abstract words, and psycholinguists have theorized they may be learned differently (i.e., concrete words by direct experience and abstract words by text and discourse). Brysbaert et al. provide human concreteness ratings for 40,000 common English lemmas rated on a scale from 1 to 5.[14] We use these ratings to measure the mean concreteness of the nouns and verbs in the story

---

[14] For example, the nouns *television*, *darkness*, and *idea* are rated 4.83, 3.85 and 1.61 respectively, and the verbs *talk*, *see*, and *hope* are rated 4.07, 3.21 and 1.25 respectively.

text – see Figure 14 in the Appendix.

We find that, for the same k, GPT2-117 tends to generate more concrete words than the Fusion Model, and that for both models, concreteness converges to approximately human levels as k increases. Interestingly, however, when k is small, the noun concreteness is much *higher* than human levels, whereas the verb concreteness is much *lower*. This indicates that for small k, both models produce stories that, compared to human-written stories, have too many physical objects (as opposed to abstract nouns), and too few physical actions (as opposed to abstract verbs). This reflects the trend demonstrated in Table 2: when k is small, the models tend to generate descriptive sentences with mostly *is* verbs (e.g. *I was always so excited*), and physical nouns (e.g. *mother, father, queen*). Only when k increases do we see more tangible actions (e.g. *The bar patrons snickered*) and abstract nouns (e.g. *pain, glances*). A detailed example, with all nouns and verbs annotated with concreteness, is in the Appendix (Table 3).

10 Conclusions

The effect of massive pretraining In this study, we find that GPT2-117 is a better story generation model than the Fusion Model in several specific ways: it conditions much more strongly on the provided context, is more sensitive to correct ordering of events, and generates text that is more contentful (using more rare words, concrete words, and named entities). In particular, the stronger conditioning result is striking, as the Fusion Model is a complex task-specific architecture designed to increase story-prompt relevance. This demonstrates that sometimes, a general-purpose model architecture can outperform a complex task-specific architecture when provided with enough pretraining data.

However, we find that in other aspects, GPT2-117 performs *no better* than the Fusion Model: when k is small, the models generate text that is equally lexically under-diverse, syntactically under-complex, and repetitive – with a tendency to fall into a snowball effect of increasing over-confidence. However, these problems correct themselves (i.e., the metrics match human levels) when the models generate from their untruncated distribution. Our results show that these oft-cited neural generation problems are *not* the fault of the models themselves (which are in fact statisti-

Prompt: You just discovered England's biggest secret: every time they say long live the queen it extends the queens life.
Human-Written Story: I am immortal, I have inside me blood of kings. My name in Charles Windsor and I am heir to a throne. Prince of Wales, they've called me that my entire life and now I lay on my death bed and my mother - still going strong. I'm not sure how it's possible but she seems ageless, almost timeless. My whole life I've waited for that [...]

Fusion Model ($k = 10$): *I have no idea when my father would come to visit this place, I was always so excited. I was always so scared. I had to go back to work. I was a little scared of him, but when I was younger, he had never come in for a visit. His mother told me, "I'm so scared", as he always did. She never talked to me about anything [...]*	**GPT2-117** ($k = 10$): ** " So what are we going to do? " * It was the first time she asked me this question that I ever had asked her. It has taken years but finally I am finally able to do it. It seems that this queen is the queen of England and she's finally the one to tell me that this queen is the reason why she's not being allowed to die. [...]*
Fusion Model ($k = 1000$): *"Where am I? What happened?" "Having been working on my job for over 6 hours now, I do not know how you worked!" "I have been working for the last three years. Surely I am an ancient god now." The bar patrons snickered. "Hello?" "Those last three years have been worse than a year ago." Pain. [...]*	**GPT2-117** ($k = 1000$): *It was an odd occasion for the Queen of England to meet with her. The plane sat idle at 3:26 PM on a Thursday night. Yesterday, the Queen had taken it upon herself to try and get a good look at the plane which had recently been found abandoned. A copious amount of curious glances from around the room until [...]*

Table 2: A prompt and human story from the dataset, plus the models' top-k generated stories, for two values of k.

cally well-trained to match human text for these metrics), nor caused by too little training data (as these problems are not improved by GPT2-117's extensive pretraining). Instead, they are primarily caused by likelihood-maximizing decoding algorithms – such as greedy decoding, beam search, and top-k sampling with low k.

The effect of k This study detailed the typical characteristics of long-form text generated by neural language models in open-ended settings, under both high entropy (large k) and low entropy (small k) decoding algorithms. The negative characteristics of low k output (genericness, repetition, over-simplicity) are by now familiar to researchers. However, we also uncovered some less obvious characteristics of low-k generated text: compared to human-written text, it tends to copy more from the provided context (particularly GPT2-117); it contains more verbs and pronouns but fewer nouns and adjectives; its nouns are more concrete but its verbs are less concrete; and it uses a smaller range of syntactic patterns (a phenomenon that can't be entirely attributed to n-gram repetition).

As k increases to vocabulary size, we find that the model-generated text closely fits the human text on most of the metrics we measured. However, it is clear by inspection that the high-k model-generated text lacks many crucial aspects such as commonsense reasoning, world knowledge and multi-sentence coherence – an example of this superficially fluent but nonsensical text can be seen in Table 4 in the Appendix. We believe that true progress in open-ended Natural Language Generation will come from attempting to address

these high k problems – i.e., strategies to imbue the language model with better reasoning, knowledge and planning abilities – rather than continuing to seek ways to mitigate the diversity and repetition problems of the low k setting.

Limitations of this study This study uses only the smallest version of GPT2. It is likely that the larger versions of GPT2 may exhibit stronger statistical differences for the metrics we examine. Such a study would illustrate the effect of larger model capacity, and more fully reveal the possible benefits of massive pretraining. We release our annotation code so that other researchers may repeat our study on more models and datasets.

This study did not include human evaluation, which is currently the only reliable way to assess overall text quality, as well as quantify the deficiencies of high k output described above (coherence, reasoning, and world knowledge). As such, this study quantifies the *diversity* side more than the *quality* side of the quality-diversity tradeoff. Consequently, this study demonstrates the importance of developing better methods to computationally quantify notions such as text coherence, logicality and commonsense correctness – an effort that may ultimately hold the key to generating text with those desirable attributes.

11 Acknowledgments

References

Shlomo Argamon, Moshe Koppel, and Galit Avneri. 1998. Routing documents according to style. In *First International workshop on innovative information systems*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*.

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. In *Empirical Methods in Natural Language Processing*.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Scott F Beers and William E Nagy. 2009. Syntactic complexity as a predictor of adolescent writing quality: Which measures? which genre? *Reading and Writing*, 22(2):185–200.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior Research Methods*, 46(3):904–911.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. In *NeurIPS Workshop on Critiquing and Correcting Trends in Machine Learning*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Association for Computational Linguistics*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Tatsunori Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Molly E Ireland and James W Pennebaker. 2010. Language style matching in writing: Synchrony in essays, correspondence, and poetry. *Journal of personality and social psychology*, 99(3):549.

Shaojie Jiang and Maarten de Rijke. 2018. Why are sequence-to-sequence models so dull? Understanding the low-diversity problem of chatbots. In *EMNLP Workshop on Search-Oriented Conversational AI*.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, Fog count and Flesch reading ease formula) for navy enlisted personnel.

Ilya Kulikov, Alexander H Miller, Kyunghyun Cho, and Jason Weston. 2018. Importance of a search strategy in neural dialogue modelling. *arXiv preprint arXiv:1811.00907*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jiwei Li and Dan Jurafsky. 2016. Mutual information and diverse decoding improve neural machine translation. *arXiv preprint arXiv:1601.00372*.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Empirical Methods in Natural Language Processing*.

Huanru Henry Mao, Bodhisattwa Prasad Majumder, Julian McAuley, and Garrison W. Cottrell. 2019. Improving neural story generation by targeted common sense grounding. In *Empirical Methods in Natural Language Processing*.

Danielle S McNamara, Scott A Crossley, and Philip M McCarthy. 2010. Linguistic features of writing quality. *Written communication*, 27(1):57–86.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Empirical Methods in Natural Language Processing*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. Fairseq: A fast, extensible toolkit for sequence modeling. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI tech report*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI tech report*.

Melissa Roemmele, Andrew S Gordon, and Reid Swanson. 2017. Evaluating story generation systems using automated linguistic analyses. In *KDD Workshop on Machine Learning for Creativity*.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? how controllable attributes affect human judgments. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI Conference on Artificial Intelligence*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Association for Computational Linguistics*.

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2018. Cold fusion: Training seq2seq models together with language models. In *Proc. Interspeech 2018*, pages 387–391.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search: Decoding diverse solutions from neural sequence models. In *AAAI Conference on Artificial Intelligence*.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, Jun Xu, and Xueqi Cheng. 2018. Learning to control the specificity in neural response generation. In *Association for Computational Linguistics*.

Zachary M Ziegler, Luke Melas-Kyriazi, Sebastian Gehrmann, and Alexander M Rush. 2019. Encoder-agnostic adaptation for conditional language generation. *arXiv preprint arXiv:1908.06938*.
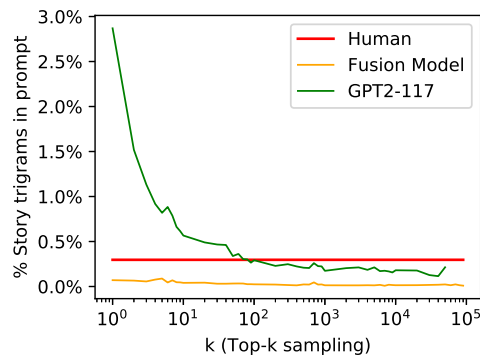
Appendix
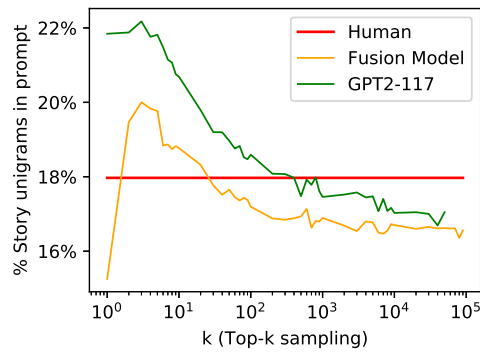
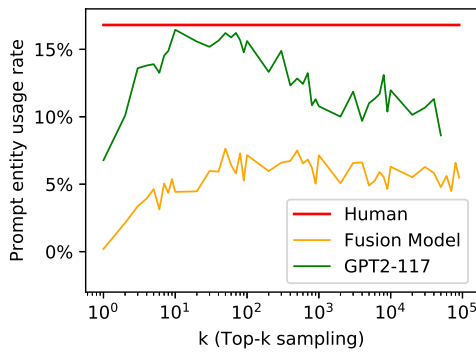(a) Percent of all story unigrams that are in the prompt.
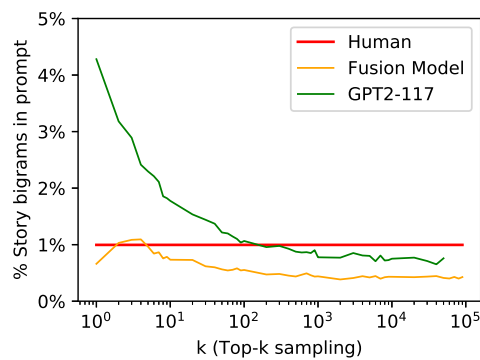
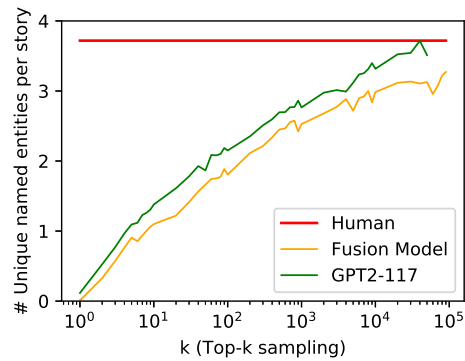(b) Percent of all story bigrams that are in the prompt.

(c) Percent of all story trigrams that are in the prompt.

Figure 6: n-gram similarity between prompt and story, for $n = 1, 2, 3$, for both models and all k. GPT2-117 copies many more n-grams from the prompt than the Fusion Model. See Section 4 for discussion.

(a) The proportion of all prompt named entities that are used in the story.

(b) The number of unique named entities that appear in the story.

Figure 7: Prompt entity usage rate (left) and mean number of unique named entities in the story (right), for both models and all k. GPT2-117 generally uses a larger proportion of the prompt named entities, and more named entities overall, than the Fusion Model. Both models generally use fewer named entities than human text when k is less than vocabulary size. See Section 4 for discussion.
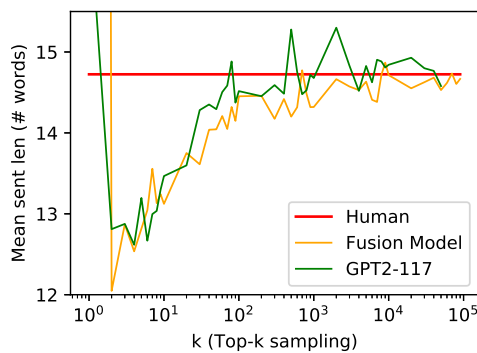
Figure 8: Mean sentence length for both models and all k. For both models, sentence length increases as k increases. The spike at $k = 1$ is due to long repeating sequences with no sentence-ending token. See Section 7 for discussion.
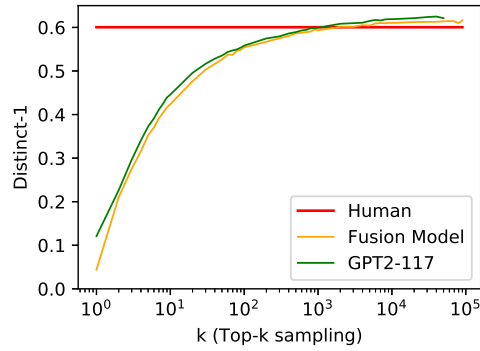
(a) Distinct-1 (ratio of unique unigrams in the story to total number of generated unigrams in the story).
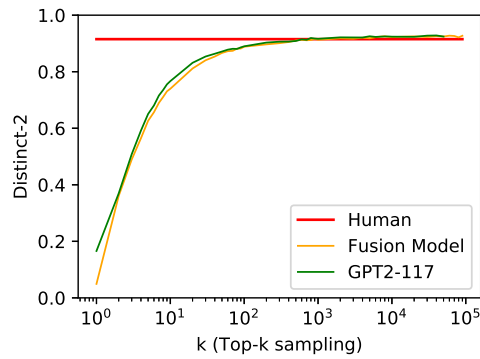
(b) Distinct-2 (ratio of unique bigrams in the story to total number of generated bigrams in the story).
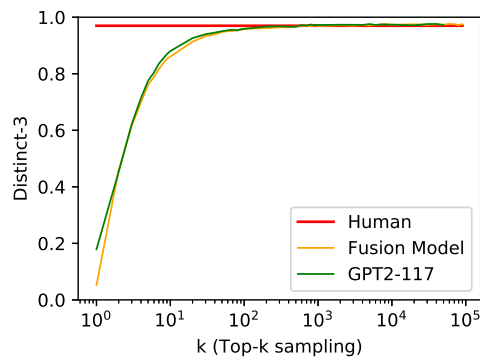
(c) Distinct-3 (ratio of unique trigrams in the story to total number of generated trigrams in the story).
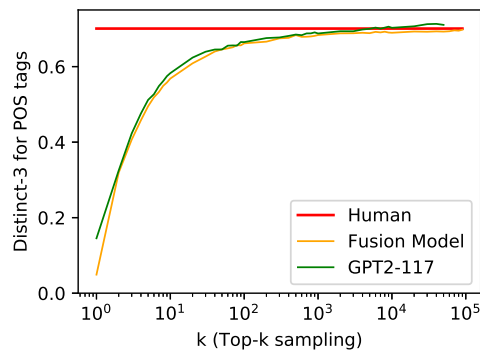
Figure 9: Distinct-n for $n = 1, 2, 3$, for both models and all k. The ratios, which represent lexical diversity, increase as k increases, with GPT2-117 reaching human levels at $k = 2000$ for unigrams, $k = 800$ for bigrams and $k = 600$ for trigrams. Lexical diversity is slightly higher for GPT2-117 than for the Fusion Model for equal k, but the primary determining factor is k. See Section 6 for discussion.
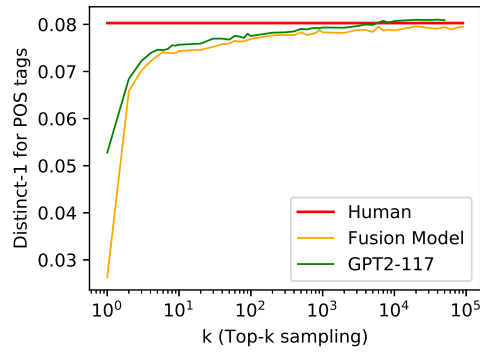
(a) POS tag distinct-1 (ratio of unique POS unigrams in the story to total number of generated POS unigrams in the story).
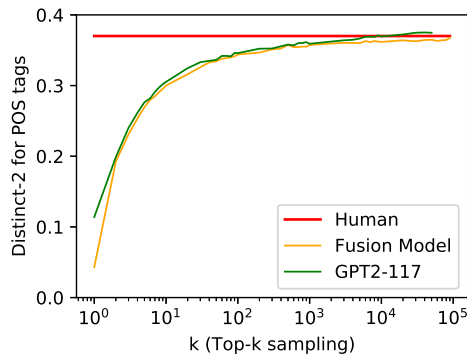
(b) POS tag distinct-2 (ratio of unique POS bigrams in the story to total number of generated POS bigrams in the story).

(c) POS tag distinct-3 (ratio of unique POS trigrams in the story to total number of generated POS trigrams in the story).

Figure 10: POS tag distinct-n metric for $n = 1, 2, 3$, for both models and all k. The ratios, which represent syntactic diversity, increase as k increases, with GPT2-117 reaching human levels at $k = 6000$ for unigrams, $k = 9000$ for bigrams, and $k = 6000$ for trigrams. Syntactic diversity is slightly higher for GPT2-117 than for the Fusion Model for equal k, but the primary determining factor is k. See Section 7 for discussion.

857
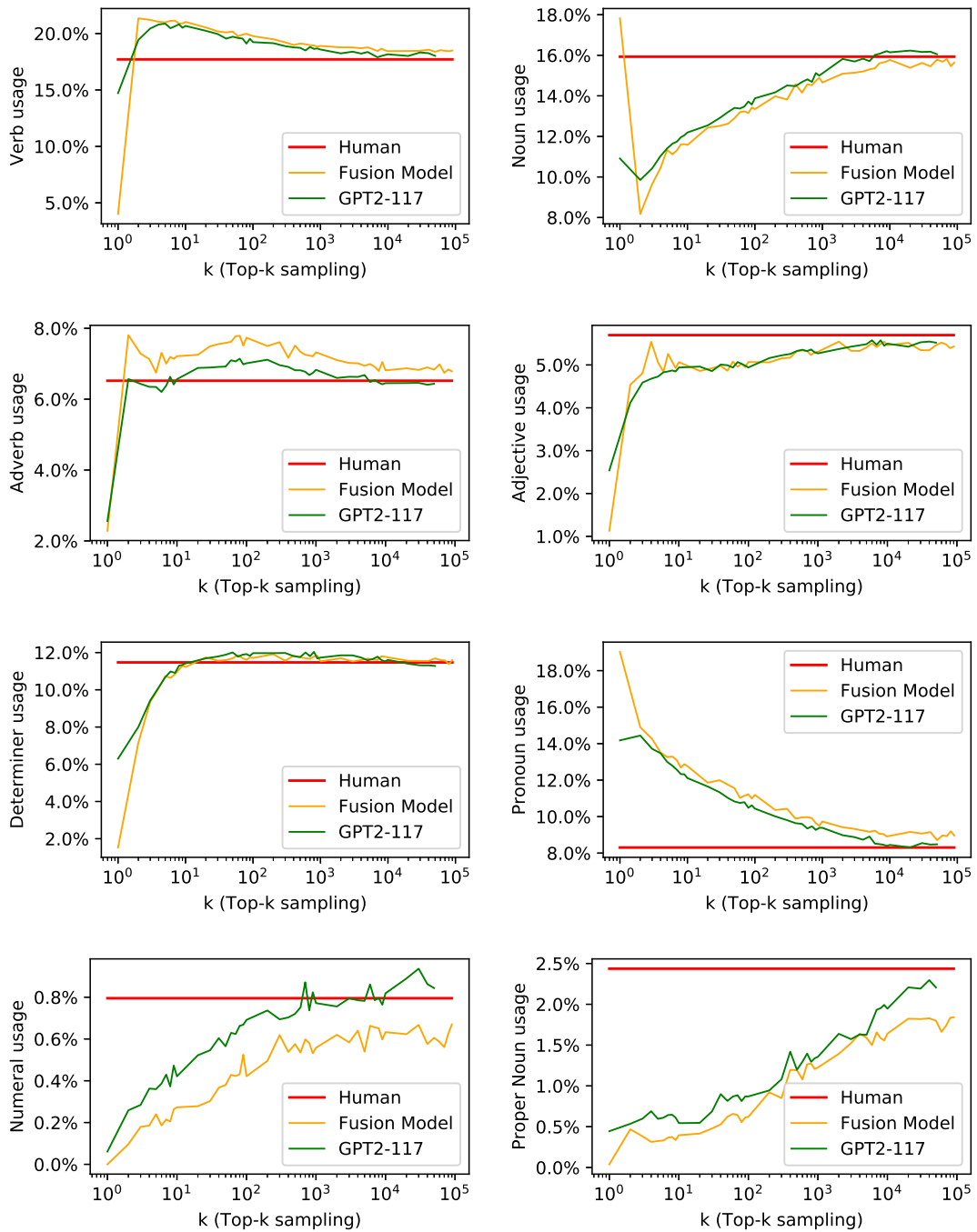
Figure 11: Usage of different POS tags in the generated stories. GPT2-117 tends to fit the human distribution more closely than the Fusion Model as k approaches vocabulary size, in particular producing more specific POS categories such as Numeral and Proper Noun. When k is small, generated text is characterized by more verbs and pronouns, and fewer nouns, adjectives, numerals and proper nouns, than human text. See Section 7 for discussion.
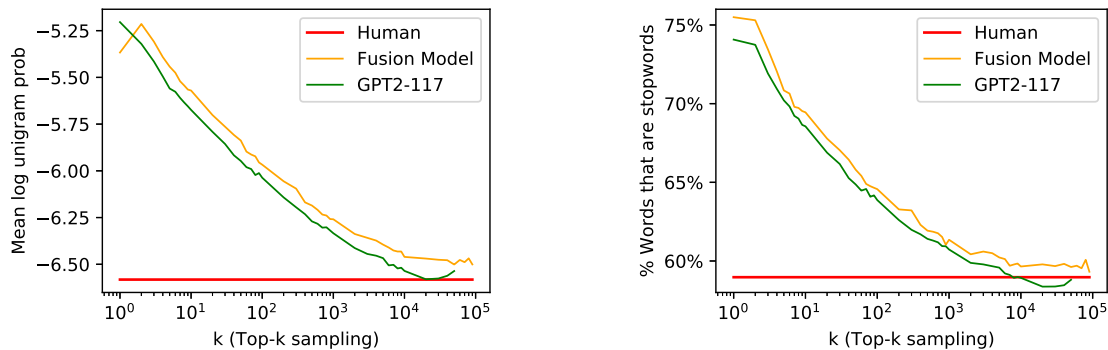
(a) The mean log unigram probability of generated words. Higher values indicate using fewer rare words while lower values indicate using more rare words.

(b) The percent of generated words that are stopwords, for both models, across different k. We use the NLTK English stopword list.

Figure 12: Rare word usage metrics for both models and all k. GPT2-117 produces slightly more rare words (left) and slightly fewer stopwords (right) than the Fusion Model, for equal values of k. These rareness metrics do not reach human levels until k is close to vocabulary size. See Section 6 for discussion.
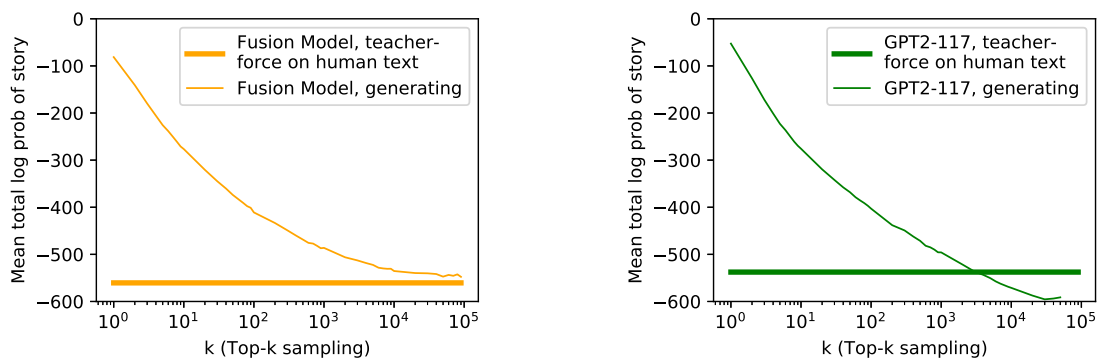
Figure 13: The mean total log probability of the story (150 words), as measured by the models on their own generated output and on human-written stories. Interestingly, the Fusion Model (left) converges to the same probability it assigns to human-written stories as k approaches vocabulary size, whereas GPT2-117 (right) converges to a lower probability. See Section 8 for discussion.

Prompt: A man finally discovers his superpower... well into his 80's.

Fusion Model ($k = 10$):

I be a child . bear into a world of darkness , and in the world , there be only a few people . My mother and I be always alone . She have never be like this . But , she have no idea what we be together to be . We be always alone . We be always together . And , of course , she have never exist . But , I never think about it , because she be always alone . In the world , it was n't like she have a power . I be always alone , and there be a time when the light would turn on . There be a time when I could see the light , and I could see it in her eyes , and I could see the light , and I could see it

Mean noun concreteness: 3.892
Mean verb concreteness: 2.173

GPT2-117 ($k = 10$):

A man in an old white coat sit in a corner . He wear a black jacket with a blue tie . His hair be grey . His eyes be black, but his voice be deep and clear . He look to where the man be sit , and he see a smile on his lips . It be a smile he know he would see from his own eyes . But he be too late . He be on the sidewalk by the river when the man come . He be wear a black coat with a purple tie . He have a black tie and a white shirt . But he be still wear a white suit . And it seem he would look back at him . A smile on his face . A look his friend do n't recognize . He have no

Mean noun concreteness: 4.720
Mean verb concreteness: 2.488

Fusion Model ($k = 1000$):

For a brief moment, the dark green of his eyes flash the last line s of reality into existence, finally fade slowly into existence so that he could not feel it. Only the millions of pixels float in his peripheral vision almost disappear. His radio respond to the pinging very hard silence of the outside world. Seven people have visit his main internal network for what seem like a lifetime but this time, the only switch to an external supply system that he could simply take advantage of. Unable to convey feelings about the last words he would have to endure, but it have respond to the innumerable message s and countless sleepless hour s. Most of them be always available on its surface, just to make sure. In his quest for to spend eternity on death, he send

Mean noun concreteness: 3.201
Mean verb concreteness: 2.435

GPT2-117 ($k = 1000$):

(First time poster, hope its ok) The young boy, watch tv, spot the television onscreen, before glance around to see the screen start the countdown on the tv, point to the screen in " It 's both the same. " "... let 's... let 's try this and... we will team up so that... we can... have the same power....like... so we can use this superpower over and over again. " A brief silence. Only a familiar conversation, interrupt his mad dash movement, follow with his high pitch slurred and wither voice : " I ca n't stand anyone talk like that son*s*. " More casual conversation that interrupt his childish step be rush to the scissor s.

Mean noun concreteness: 3.793
Mean verb concreteness: 3.162

Table 3: Generated stories from both models, under $k = 10$ and $k = 1000$. Nouns are highlighted in green and verbs in yellow. The highlighting intensity reflects the word's concreteness rating. For equal k, GPT2-117 generally generates more concrete words than the Fusion Model. For both models, low k is characterized by high noun concreteness (e.g. physical objects such as *jacket*) and low verb concreteness (e.g. non-physical actions such as *be*). Conversely, high k is characterized by low noun concreteness (e.g. abstract concepts such as *reality*) and high verb concreteness (e.g. physical actions such as *talk*). See Section 9 for discussion.
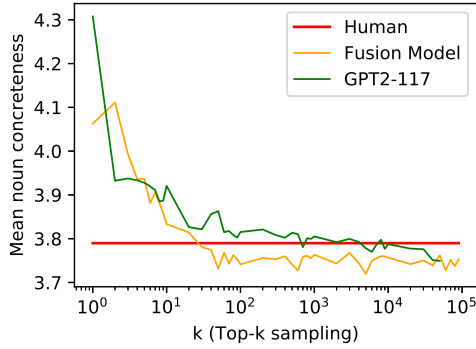
(a) Mean concreteness rating (1-5) of nouns in the story.
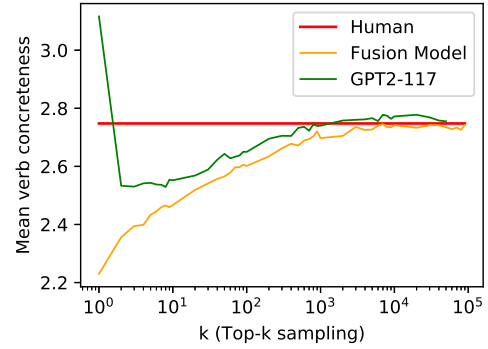
(b) Mean concreteness rating (1-5) of verbs in the story.

Figure 14: Mean concreteness rating of the nouns and verbs in the story, for both models and all k. GPT2-117 generally produces nouns and verbs that are more concrete than the Fusion Model for the same k. For both models, as k increases, noun concreteness reduces and verb concreteness increases. See Section 9 for discussion.

Prompt: *In an alternative reality where sleep is non-existent among living beings, our protagonist (spontaneously or after an event) falls asleep in which he/she experiences for the first time in human history what a dream is.*

GPT2-117 ($k = 1000$)**:** *I sat in my bed as my girlfriend sat behind me, buzzing into her e-reader, letting the day's stories write themselves on her's monitor. Like a blur, all the usual high-asyllabic drivel and senseless ramblings that normally attracted the attention of a horrid sleeping creature huddled about me like a faucet. She did not know how I placed this car with her. But I tried, first tried to ignore that I had hired the services of a dog to help and then quietly used it in a desperate bid to drive the car through the lawn. Each and every day, I watched her drool down an old dusty hardwood mattress her beady eyes trying desperately to think of this rotting dream. [...]*

Table 4: An example of syntactically natural but nonsensical text, generated with high k. Though the text is fluent and readable, it is surreal and bizarre. See Section 10 for discussion.