EVALUATING DISTRIBUTIONAL DISTORTION IN NEURAL LANGUAGE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

A fundamental characteristic of natural language is the high rate at which speakers produce novel expressions. Because of this novelty, a heavy-tail of rare events accounts for a significant amount of the total probability mass of distributions in language (Baayen, 2001). Standard language modeling metrics such as perplexity quantify performance of language models (LM) in aggregate. As a result, we have relatively little understanding of whether neural LMs accurately estimate the probability of sequences in this heavy-tail of rare events. To address this gap, we develop a controlled evaluation scheme which uses generative models trained on natural data as artificial languages from which we can exactly compute sequence probabilities. Training LMs on generations from these artificial languages, we compare the sequence-level probability estimates given by LMs to the true probabilities in the target language. Our experiments reveal that LSTM and Transformer language models (i) systematically underestimate the probability of sequences drawn from the target language, and (ii) do so more severely for less-probable sequences. Investigating where this probability mass went, (iii) we find that LMs tend to overestimate the probability of ill-formed (perturbed) sequences. In addition, we find that this underestimation behaviour (iv) is weakened, but not eliminated by greater amounts of training data, and (v) is exacerbated for target distributions with lower entropy.

1 INTRODUCTION

Natural language is fundamentally creative—speakers and listeners frequently produce and comprehend sentences which have never been produced before (Fodor, 1975; Fodor & Pylyshyn, 1988; Chomsky, 1975, 1955). As a side-effect of this property, distributions in natural language are characterized by a *heavy-tail* of individually improbable events which collectively account for a significant amount of the total probability mass of the distribution (Khmaladze, 1988; Baayen, 2001). Precisely approximating this large number of rare events is one of the foundational challenges for models of natural language (Good, 1953; Jelinek, 1980; Katz, 1987; Kneser & Ney, 1995; Wood et al., 2011; Goldwater et al., 2011). *Autoregressive neural language models* (Bengio et al., 2003; Mikolov et al., 2013; Radford et al., 2019) attempt to do so by decomposing the probability of an event (a sequence) into a series of conditional distributions, each parameterized by a shared neural network.

Recently, a growing body work has sought to understand how these language models (LM) fit the distribution of a language beyond standard measures such as *perplexity*. Meister & Cotterell (2021), for example, investigated the statistical tendencies of the distribution defined by neural LMs, whereas Kulikov et al. (2021) explored whether they adequately capture the modes of the distribution they attempt to model. At the same time, increased focus has been given to performance on rare or novel events in the data distribution, both for models of natural language (Lent et al., 2021; Dudy & Bedrick, 2020; Oren et al., 2019) and neural models more generally (see, for example Sagawa et al., 2020; D'souza et al., 2021; Chen et al., 2021; Blevins & Zettlemoyer, 2020; Czarnowska et al., 2019; Horn & Perona, 2017; Ouyang et al., 2016; Bengio, 2015; Zhu et al., 2014). Neither of these branches of work, however, has explored instance-level LM performance on rare sequences in the distribution. As a result, we have relatively little understanding of how neural LMs approximate sequences in the heavy-tail characteristic of natural language.

In this work, we introduce a controlled methodology to explore how LMs estimate the probability of sequences in the heavy-tail of the distribution. Our instance-level evaluation scheme explicitly compares the target probability distribution of the language to the distribution defined by the LM. Since the true distribution of any natural language is in practice unknown, we use a Transformer LM trained on natural data as a generative model to define target *artificial languages* for which we can exactly compute sequence probabilities. Training LSTM and Transformer LMs on sequences sampled from these target artificial languages, we compare the sequence-level probability estimates given by neural LMs to the target probabilities in the language. By controlling the entropy of the generative model's conditional distributions, we create a set of artificial languages with varying distributional properties, and analyze how LM estimation behaviour is modulated by the properties of the target distribution.

Our experiments uncover the extent to which neural LMs provide a distorted fit of the language they are trained to model. We find that LSTM and Transformer LMs (i) systematically underestimate the probability of sequences drawn from the target language and (ii) do so more when such sequences are rare. Where did this underestimated probability mass go? We do not find that the underestimation is accompanied by overestimation in the head of distribution. Rather, we find that LMs tend to (iii) overestimate the probability of rare perturbed (ill-formed) sequences. Interpreted together, these findings indicate that on the one hand, neural LMs under-represent well-formed sequences in the tail of the language they attempt to model, and on the other hand, over-represent ill-formed sequences far away from high probability zones in sequence-space. In addition, we find that (iv) greater amounts of training data lessen underestimation but do not eliminate it and that (v) underestimation is exacerbated for target distributions with lower entropy.

2 Background

We begin by briefly characterizing why distributions with a large number of rare events (LNRE) emerge in natural language, and why these events pose challenges for LMs. Furthermore, we motivate the need for instance-level evaluation when dealing with a large number of rare events.

Productivity In the context of language production, a language user has the ability to produce, at any given point in their linguistic lifespan, an utterance which they have never produced before. This creativity is the result of the generative property of *productivity*, which states that on the basis of finite linguistic experience, a language user can produce and comprehend an unbounded number of grammatically acceptable utterances (Chomsky, 1975, 1955). Productive processes induce a distribution which places non-zero probability on unseen events at all practical sample sizes. Because of this property, many of the distributions in natural language—particularly the distribution over the sequences of a language—are characterized by a heavy-tail of rare events.

LNRE Zone To make explicit the connection between productivity and a heavy-tail of rare events, let $\mathcal{P}_N$ denote the probability of sampling a novel (previously unseen) event from some distribution after having sampled N events. Then productivity as described above states that $\mathcal{P}_N > 0$ for all sample sizes N that occur in practice. The range of sample sizes N for which it is the case that $\mathcal{P}_N > 0$ is known as the *LNRE zone* (Khmaladze, 1988; Baayen, 2001). The LNRE zone for natural language appears to be very large, and it seems likely that $\mathcal{P}_N$ will remain greater than 0 for samples of natural language many orders of magnitude larger than all the data currently available for training LMs.[1] In the LNRE zone, it is difficult to obtain accurate estimates of the probability of events using straightforward maximum likelihood estimation (MLE). Accounting for this enormous amount of novelty is thus a central challenge in language modeling.

Language modeling A model M of the language L attempts to define a distribution p_M over variable length sequences $\boldsymbol{x} = (x_1, \ldots, x_{|\boldsymbol{x}|})$ which closely resembles the true distribution of the language p_L. That is, $p_M(\boldsymbol{x}) \approx p_L(\boldsymbol{x})$ for all $\boldsymbol{x} \in \Sigma^*$, where Σ denotes the vocabulary of M, and Σ^* is the set of all strings of finite length, known as the Kleene closure of Σ. In the LNRE zone, this means that a LM must define a distribution over a support containing a very large set of sequences

[1] For an empirical validation of this claim on a sample of practical size from the OpenWebText corpus, see the Appendix.

which have never occurred in a training corpus (or equivalently, have all occurred with frequency 0), which take on a very wide array of differing probabilities. For example, while the sequences $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \Sigma^*$ have likely never occurred in any sample of English, most would agree that $\boldsymbol{x}_1$ is far more probable than $\boldsymbol{x}_2$:

$\boldsymbol{x}_1$: *The East pond in Parc Lafontaine was filled to the brim with diet Coke.*

$\boldsymbol{x}_2$: *Certain nak indicate liberationing among theorter codity voters vandalized.*

LM Evaluation For a perfect LM of English, we would expect the estimated probabilities of the sequences $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ to match their probabilities under the true distribution p_{English}. However, since p_{English} and it's underlying generative process are unknown, it is not possible to explicitly evaluate how closely instance-level probability estimates align. As a proxy, the mean perplexity of the model on a holdout set of sequences $\mathcal{D}$ is typically used:

$$PP(p_M, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \exp \left\{ -\frac{1}{|\boldsymbol{x}^{(i)}|} \sum_{t=1}^{|\boldsymbol{x}^{(i)}|} \log p_M(x_t^{(i)} \mid x_{<t}^{(i)}) \right\} \tag{1}$$

which measures whether the model, on average, assigns high likelihood to unseen instances. This measure does not, however, tell us whether instance-level estimates align with their true counterparts, nor is it necessarily indicative of performance on rare, idiosyncratic events in $\mathcal{D}$. In this way, the lack of access to the ground-truth distribution severely complicates LM evaluation on the heavy-tail of rare sequences in language. In the following section, we introduce a methodology to overcome these limitations.

3 Language Model Evaluation in the LNRE Zone

Component	**Notation**	**Description**
Generative model	L	A LM trained on natural instance-level data.
Artificial language	p_L	The distribution over sequences induced by a sampling scheme from L.
Language model	p_M	The distribution of a LM trained on sequences sampled from p_L.
Target probabilities	$p_L(\boldsymbol{x})$	The probability assigned by p_L to the sequence $\boldsymbol{x}$.
Model probabilities	$p_M(\boldsymbol{x})$	The probability assigned by p_M to the sequence $\boldsymbol{x}$.

Table 1: Components of our instance-level evaluation scheme. Training p_M on samples from p_L, we compare $p_M(\boldsymbol{x})$ to $p_L(\boldsymbol{x})$ for $x \in \Sigma^*$.

We propose evaluating language model performance on the heavy-tail of rare events by working with a known probability distribution over sequences. Specifically, we train a Transformer LM on sequences sampled from a corpus of natural language to define a generative model L. The distribution over sequences induced by a sampling scheme from L, denoted p_L, is then our *artificial language*. We expect a model M of this artificial language to assign probabilities $p_M(\boldsymbol{x})$ to sequences $\boldsymbol{x}$ which match the *target probabilities* $p_L(\boldsymbol{x})$ of $\boldsymbol{x}$ under p_L. Here we define p_L using an ancestral sampling scheme with softmax $T = 0.85$. One could, for example, define an artificial language $p_{L'}$ by sampling from the top-k tokens of the distribution defined by L at each time step, or using an ancestral sampling scheme with softmax $T = T'$. To characterize neural LM behaviour on rare events, we train Transformer and LSTM LMs on data sampled from p_L, and compare the instance-level probability estimates given by p_M to target probabilities under p_L. We summarize the components of this methodology in Table 1, and overview it in greater detail in the following section.

3.1 Artifical Languages as Target Distributions

To define a generative model L, we train a randomly-initialized GPT2-medium on 1.5M sentences sampled from the OpenWebText corpus (Gokaslan & Cohen, 2019).[2] We set the maximum sequence

[2]All Transformer implementations were obtained from Huggingface, and training was done on 2 RTX-8000 GPUs.

$\boldsymbol{x}$	$\log p_L(\boldsymbol{x})$
"we're very excited to have the opportunity to help them," he says.	-29.3811
so what's going to happen?	-17.4128
to me, the fisheries are in the midst of a global financial crisis.	-41.4835

Table 2: Sample of sequences drawn from our artificial language.

length to be 128 tokens. We additionally train a byte-pair-encoding (BPE) tokenizer on this data set with a standard GPT2 vocabulary size of 57,256 tokens. For simplicity, this tokenizer is used for all models.

Using this generative model L, we define the target distribution over sequences—the artificial language—as the distribution induced by an ancestral sampling scheme from L. Thus, we draw instances $\boldsymbol{x} = (x_1, \ldots, x_{|\boldsymbol{x}|})$ from our language p_L by recursively sampling from the conditional distribution over tokens at each time step: $x_t \sim p_L(\cdot \mid x_{<t})$ where $x_0 = \text{BOS}$ and $x_t = \text{EOS}$. All experiments up until Section 4.5 are conducted on the distribution induced by ancestrally sampling from L with softmax temperature $T = .85$. In Section 4.5, we explore the effects of different values of T when sampling from p_L. Table 2 shows three sequences sampled from this distribution.

3.2 Sequence Probability Estimation in the LNRE Zone

Given an artificial language p_L, the task of M—the language model—is to define a distribution p_M whose sequence-level probability estimates closely align with the sequence-level probability estimates given by p_L. We refer to any deviation from this desiderata as *model estimation error*. To quantify the model estimation error for a sequence $\boldsymbol{x}$, we take the difference between the sequence's log probability under M and its true log probability under L:

$$\text{error}(\boldsymbol{x}) = \log p_M(\boldsymbol{x}) - \log p_L(\boldsymbol{x}) \tag{2}$$

This quantity is the log probability ratio, which measures, in log-space, the number of times more or less likely the sequence $\boldsymbol{x}$ is under the language model M. Note that $\text{error}(\boldsymbol{x}) < 0$ indicates that M underestimates the probability of $\boldsymbol{x}$, whereas $\text{error}(\boldsymbol{x}) > 0$ indicates that M overestimates the probability of $\boldsymbol{x}$. In practice, we train M on a set of sequences $\mathcal{D}_{\text{train}}$ sampled from p_L, and compute model estimation error on a separate set of sequences $\mathcal{D}_{\text{test}}$ sampled from p_L. In all cases, we compute the probability of a sequence $\boldsymbol{x}$ as its chain rule decomposition: $p(\boldsymbol{x}) = \prod_{i=1}^{|\boldsymbol{x}|} p(x_i \mid x_{<i})$ where $x_0 = \text{BOS}$ and $x_{|\boldsymbol{x}|} = \text{EOS}$. When computing the ground-truth sequence probabilities, we take into account any softmax tempering.

3.3 Neural Language Models

We study the estimation performance of two neural LM architectures: the Transformer (Vaswani et al., 2017) and the LSTM (Melis et al., 2020). When training either architecture, we halve the learning rate if validation loss increases at the end of an epoch. For all model sizes, we use a batch size of 128 sequences. Models with the lowest cross-entropy loss on a withheld validation set are used in experiments unless otherwise mentioned.

GPT2-small & GPT2-medium We use the Huggingface (Wolf et al., 2020) implementations of GPT2-small and GPT2-medium (Radford et al., 2019) as representative Transformer LMs. All model training was done from scratch on two RTX-8000 GPUs with mixed floating point precision. We use Adam Optimization with $\epsilon = 1e^{-8}$ and learning rates $\alpha = 5e^{-5}$ and $\alpha = 4e^{-5}$ for GPT2-small and GPT2-medium, respectively. Since these models are in the same model class as our artificial target language p_L, we expect the task of recovering the ground-truth distribution to be relatively easy compared to the true problem faced in modeling natural language, where both the distribution and the underlying generative process are unknown.

LSTM For the LSTM, we follow the implementation of the baseline LM described in (Hochreiter & Schmidhuber, 1997; Melis et al., 2020). We use 2 layers and adjust the hidden state and embedding dimension (2048 and 1024, respectively) to be such that the total number of parameters is approximately equal to GPT2-small (110M).

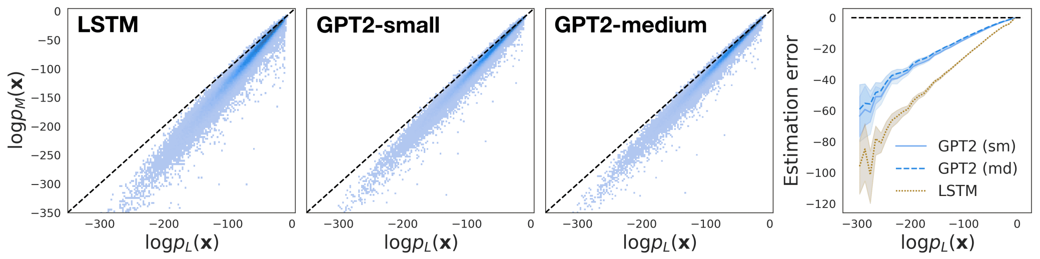


Figure 1: Test sequence probability estimates given by neural LMs. *Three left-most figures*: The joint histograms of sequence probability estimates. The dotted line denotes the cases in which the model's estimates perfectly align with the target probability; shading to the right of this line denotes underestimation. *Right-most figure*: Mean sequence estimation error by target sequence probability.

4 Results

4.1 Estimation Error with Fixed Data

We begin by exploring model estimation error on a fixed training set $\mathcal{D}_{\text{train}}$ of 1M sequences sampled from p_L. We first train LSTM and GPT models on $\mathcal{D}_{\text{train}}$, early-stopping as described above. Following training, we sample a test set $\mathcal{D}_{\text{test}}$ of 500,000 sequences from p_L, and score each sequence under both the model distribution p_M and the true language distribution p_L. From this, we obtain a set of probability estimates: $\mathcal{S}_{\text{test}} = \{\langle p_L(\boldsymbol{x}), p_M(\boldsymbol{x})\rangle \mid \boldsymbol{x} \in \mathcal{D}_{\text{test}}\}$. If the model M perfectly models the language L, then for each $\langle p_L(\boldsymbol{x}), p_M(\boldsymbol{x})\rangle \in \mathcal{S}_{\text{test}}$ we would expect $p_L(\boldsymbol{x}) = p_M(\boldsymbol{x})$. Figures 1(A) and 1(B) visualize this relationship with the x- and y-axes denoting the true and model estimated sequence probabilities respectively, and a dashed line representing equality. To compare probability estimates, we represent the set $\mathcal{S}_{\text{test}}$ in the form of a joint histogram over this coordinate space. Histogram bins are shaded based on the number of tuples which lie in the coordinate range they define. Importantly, any deviation of this histogram from the identity line indicates that the model distorts the shape of the distribution of the language.

Figures 1(A) and 1(B) provide evidence for distributional distortion in the form of underestimation. The majority of probability tuples in $\mathcal{S}_{\text{test}}$ lie to the right of the identity line, indicating that LSTM and GPT2 models consistently underestimate the probability of sequences sampled from p_L. Furthermore, the distance between the identity line and the probability tuples grows non-linearly as function of the true sequence probability, indicating that underestimation error is more severe for rarer sequences in the language. We validate both of these observations in the right-most plot of Figure 1, which shows mean estimation error decreasing non-linearly as a function of the target sequences probability.[3] In addition, comparing underestimation behaviour across model size, we find that while GPT2-medium performs slightly better than GPT2-small, these improvements are typically within the range defined by the bootstrapped 95% confidence intervals.

4.2 Estimation Error across Training Time

To understand the training dynamics underlying the previously reported underestimation, we compute model probability estimates on subsets of $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ at the end of each training iteration i. Once computed, we sort each set of probability tuples $\mathcal{S}^{(i)}_{\text{train}}$ and $\mathcal{S}^{(i)}_{\text{test}}$ by their target sequence probabilities $p_L(\boldsymbol{x})$, and split the probability tuples into 50 equally-sized bins. We plot estimation curves in Figure 2: Each curve represents a 50th of the sequences, with darker curves denoting estimation error for sequences with lower target probabilities (rarer sequences). At any given point, then, the distance between estimation curves represents the degree to which estimation error is dependent on the target probability of the sequence.

[3]To compute this curve, we split the target sequence probability $p_L(\boldsymbol{x})$ range into N equally sized bins (by probability range). We report the mean estimation error for each bin with > 10 sequences. We additionally compute 95% confidence intervals with $10,000$ bootstraps for each mean, resampling n equal to the number of sequences in the given bin.

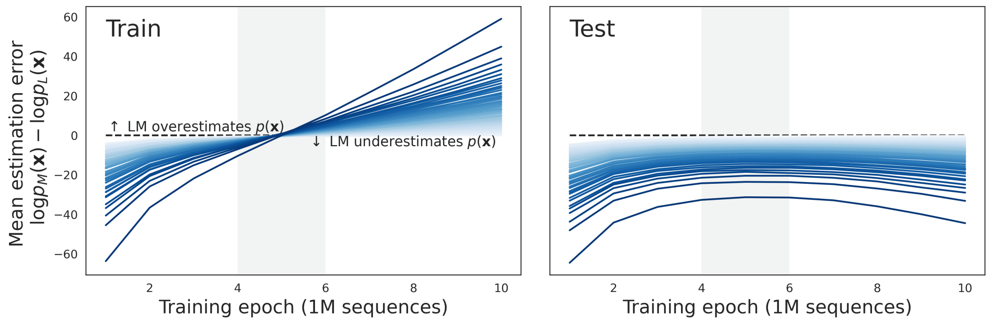


Figure 2: Mean model estimation error by training epoch (GPT2-medium). Each line denotes the mean estimation error for a 50th of the sequences; darker lines represent less probable sequences. The shaded area denotes the area in which validation cross-entropy reaches a minimum.

Figure 2 left visualizes underestimation error for sequences seen in training. Around the fifth epoch, estimation error for train sentences converges to zero, that is $(p_L(\boldsymbol{x}) \approx p_M(\boldsymbol{x}))$, indicating that GPT2-medium is able to almost perfectly recover the target probabilities of training sequences no matter their target probability. At the same time, this convergence happens almost simultaneously for all sequences, indicating that a complete reduction in error during training occurs throughout the entire range of target sequence probabilities.

Figure 2 right visualizes GPT2-medium model's performance on a separate set of test sequences. First, unlike for $\mathcal{D}_{\text{train}}$, estimation error for $\mathcal{D}_{\text{test}}$ does not converge to zero, meaning that even when the model has perfectly recovered the target probability of train sequences, the target probabilities for test sequences remain underestimated. Second, in the case of $\mathcal{D}_{\text{train}}$, the difference between estimation curves of different shades converges to zero, indicating that estimation performance becomes uniform across the distribution of train sequences. We do not see such behaviour in $\mathcal{D}_{\text{test}}$. Instead, the error curves remain at a relatively consistent distance from one another, indicating that the discrepancy in estimation error at different parts of the distribution is unchanging for sequences not seen during training.

4.3 Estimation Error by Amount of Training Data

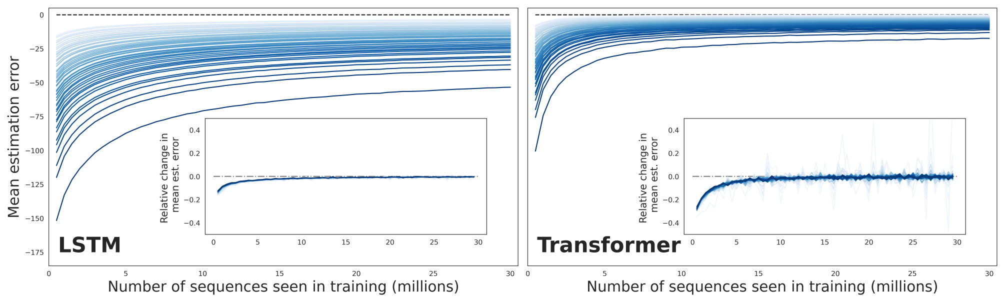
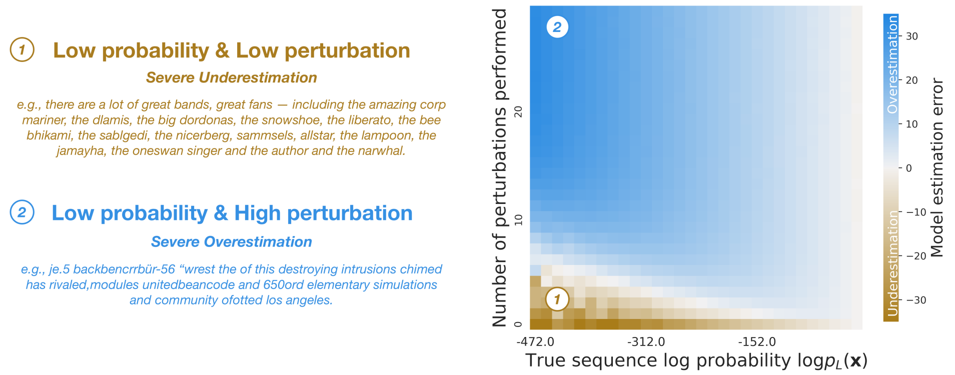


Figure 3: GPT2-medium and LSTM trained on 30M sequences sampled from p_L. *Main plots*: Mean estimation error on test sequences as a function of the number of sequences seen in training. *Inset plots*: Relative change in mean estimation error of test seqeunces as a function of the number of sequences seen in training. In both cases, each line denotes estimation behaviour for a 50th of the test sequences; darker lines represent less probable sequences.

Our previous experiment trained languages models on a set of 1M sequences. A plausible explanation for the model's underestimation behaviour on unseen test sequences is therefore that the language model has not seen enough data. Here we explore how estimation error varies as a function of the amount of training data. We train GPT2-medium and an LSTM model in the online "Ideal World" setting (Nakkiran et al., 2020) by sampling, at the beginning of each training iteration, a fresh set of 500,000 sequences from p_L, and training M on this sample. Doing so for 60 iterations, we obtain LMs which have been trained on 30 million sequences. We compute model estimation



1 **Low probability & Low perturbation**

Severe Underestimation

e.g., there are a lot of great bands, great fans — including the amazing corp mariner, the dlamis, the big dordonas, the snowshoe, the liberato, the bee bhikami, the sablgedi, the nicerberg, sammsels, allstar, the lampoon, the jamayha, the oneswan singer and the author and the narwhal.

2 **Low probability & High perturbation**

Severe Overestimation

e.g., je.5 backbencrrbür-56 "wrest the of this destroying intrusions chimed has rivaled,modules unitedbeancode and 650ord elementary simulations and community ofotted los angeles.

Figure 4: GPT2-medium estimation behaviour for 15M sequences in Σ^* across two dimensions: **sequence rarity** (x-axis) and **degree of perturbation** (y-axis). The heat map is shaded based on estimation error severity; blue areas indicate overestimation, whereas brown areas indicate underestimation. We also include example sequences from two zones in this sequence space.

error on $\mathcal{D}_{\text{test}}$ at the end of each iteration i. Figure 3 visualizes underestimation error throughout training for GPT2-medium and LSTM language models. We again split test sequences by their true probability, with darker lines denoting estimation trends for less probable target sequences.

The estimation curves in Figure 3 suggest that while increasing the amount of data in training initially leads to lower estimation error, this reduction eventually asymptotes. In figure 3, we visualize the relative change in mean estimation between epochs $i - 1$ and i. Relative change in estimation error eventually fluctuates around 0 (minimal change) for the majority of the distribution. Comparing architectures, we find that the Transformer is significantly more efficient at reducing mean estimation error throughout the distribution.

4.4 WHERE DID THE PROBABILITY MASS GO?

In the previous section, we saw that even when increasing the amount of training data, p_M consistently underestimates the probabilities of sequences sampled from the tail of p_L. At the same time, we did not find that p_M overestimated sequences in the head of p_L. Under the assumption that p_M is a proper probability distribution,[4] that is, $\sum_{\boldsymbol{x} \in \Sigma^*} p_M(\boldsymbol{x}) = 1$, these findings suggest that there exists sequences in Σ^* whose probability is overestimated by the model. In this section, we investigate where this probability mass went.

To do so, we compute model estimation error on perturbed sequences from p_L—sequences in Σ^* which are increasingly far away from the high-probability zones in p_L. We build a corpus of perturbed sequences by recursively applying 30 random perturbations to each sequence $\boldsymbol{x} \in \mathcal{D}_{\text{test}}$. Formally, the set of sequences at perturbation step i can be expressed as: $\mathcal{D}^{(i)}_{\text{perturbed}} = \{\text{PERTURB}(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{D}^{(i-1)}_{\text{perturbed}}\}$ where $\text{PERTURB}(\boldsymbol{x})$ is a function which returns a novel perturbed version of $\boldsymbol{x}$, and $\mathcal{D}^{(0)}_{\text{perturbed}} = \mathcal{D}_{\text{test}}$. Sequence perturbation operations are shown in Table 3. While it is possible that these operations produce other well-formed strings, we expect this to be a relatively rare outcome. We score each of these 15M sequences under both the target generative model p_L and the LM p_M. Note that we use as LM the GPT2-medium model from the previous section (trained on 30M sequences).

Swap two tokens in $\boldsymbol{x}$.
Delete a token from $\boldsymbol{x}$.
Insert a token from Σ at a position in $\boldsymbol{x}$.
Substitute a token in $\boldsymbol{x}$ with a token from Σ.

Table 3: Summary of perturbations. The function $\text{PERTURB}(\boldsymbol{x})$ randomly applies one of these perturbations to $\boldsymbol{x}$.

Figure 4 visualizes GPT2-medium's mean estimation error for these 15M sequences across two dimensions. On the x-axis, we plot the target probability of the sequence under p_L, and on the y-axis, the number of perturbations performed on the sequence. For example, the bottom-left corner (1) of Figure 4 visualizes estimation behaviour for rare sequences sampled directly from p_L, whereas

[4] See Welleck et al. (2020a) for discussion on consistency in the context of neural language model decoding.

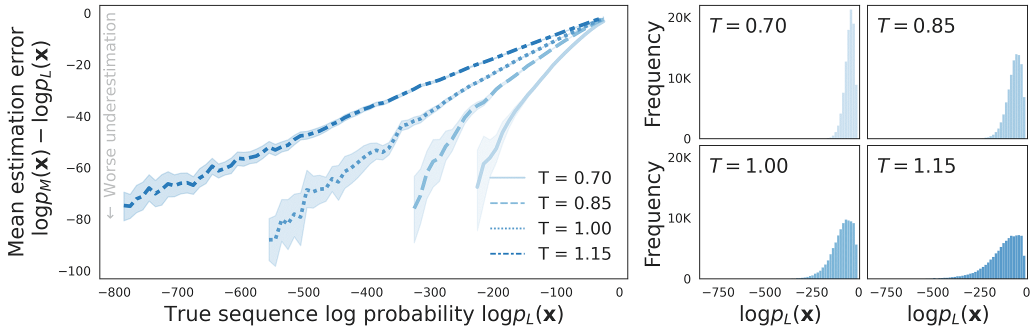


Figure 5: Model estimation error on test sequences as a function of target sequence probability for three different artificial languages. Each line visualizes estimation error for a GPT2-medium model trained to model a language with a specific softmax temperature parameter T.

the top-left corner (2) visualizes estimation error sequences which are equally rare, but which have been perturbed up to 30 times.

Figure 4 offers a nuanced characterization of underestimation behaviour. The brown area on the bottom of the figure re-states the underestimation findings of the previous section. When increasing the number of perturbations performed, however, we begin entering into a space of sequences which are at first well-estimated by p_M (the white areas) but then are quickly overestimated by p_M (the dark blue areas), confirming that there are indeed sequences in Σ^* which are overestimated by the language model. Furthermore, these findings suggest that the tail of rare events defined by the language model does not match the tail of the artificial language—the rare events typical in p_L are under-represented in p_M in favour of other sequences in Σ^*. See Section A.1 for experiments finding that random sequences from Σ^* are also overestimated by p_M.

4.5 Modulating the Shape of the Target Distribution

Up to this point, the target artificial language p_L was given as the distribution induced by an ancestral sampling scheme with softmax $T = 0.85$ from the generative model L. In the previous section, we saw that p_M placed excess probability mass on areas in Σ^* with low-probability under p_L. Here we modulate the shape of the sequence space defined by p_L to investigate how estimation error varies with respect to systematic interventions in the target distribution. To adjust the way that p_L allocates probability mass over Σ^*, we control the entropy of the conditional distributions at each generation step t by dividing the pre-softmax logits by a temperature value T.[5]

We visualize the effects of T on the shape of the distribution in the left of Figure 5. By increasing the value of T, we increase the entropy of the distributions over next tokens, which in turn, spreads probability mass across a larger number of sequences in Σ^*. We define four artificial languages with varying T and train GPT2-medium on an ancestral sample of 1M sequences from each of these artificial languages. Figure 5 visualizes model estimation error by true sequence probability for each model. We find that models trained on languages with increased entropy perform comparatively better than models trained on low entropy languages. Estimation error for models trained on languages with greater T is less severe, and this holds throughout nearly all target sequence probabilities. These results indicate that neural LMs provide a more accurate approximation of target distributions which spread mass more uniformly across Σ^*.

5 Related Work

This paper contributes to recent work investigating the properties of the distributions defined by LMs. Prior studies have focused on exploring (Takahashi & Tanaka-Ishii, 2019; 2017) and developing frameworks (Meister & Cotterell, 2021) to better understand whether the large-scale statistical

[5]Formally, for the ith component of the length K pre-softmax logits $\boldsymbol{x}$, this operation is given as: $\text{SOFTMAX}(\boldsymbol{x})_i = \frac{\exp(\frac{x_i}{T})}{\sum_{j=1}^{K} \exp(\frac{x_j}{T})}$

tendencies of natural language, such as Zipf's law (Zipf, 1949), are captured by LMs. We take a more fine-grained approach, proposing a methodology which draws off of instance-level evaluation schemes (Zhong et al., 2021) and the experimental control afforded by artificial corpora (White & Cotterell, 2021; Papadimitriou & Jurafsky, 2020). Indeed, closely related to our work is Kulikov et al. (2021)'s, in which artificial corpora produced by generative models were used to explore mode recovery in neural language modeling. Our analysis exploring the overestimation of ill-formed sequences extends previous findings on locally normalized conditional discriminative models assigning arbitrary probability mass to unlikely sequences (Andor et al., 2016; Goyal et al., 2019), neural LMs assigning high likelihood to sequences with repetitions (Welleck et al., 2020b), the consistency of decoding algorithms (Welleck et al., 2020a), and on machine translation models placing significant probability mass on the empty sequence (Stahlberg & Byrne, 2019).

We additionally contribute to the body work seeking to characterize and adapt neural model performance on rare or novel examples and classes (Horn & Perona, 2017; Bengio, 2015). In the context of language modeling, Lent et al. (2021) explored performance on under-resourced languages, whereas Oren et al. (2019) did so on under-represented domains in training corpora. Focusing on the word frequency distribution, Dudy & Bedrick (2020) found that LMs under-perform when less frequent examples are encountered at test time. In the classification setting, various approaches have been proposed to help alleviate class imbalance in the data distribution, such as data augmentation (Sagawa et al., 2020) or the transfer of knowledge from high-frequency classes to infrequent ones (Ouyang et al., 2016; Zhu et al., 2014; Chen et al., 2021). Prior to the current neural paradigm (Bengio et al., 2003), multiple approaches have been proposed to deal with the heavy-tail, such as smoothing and back-off approaches in statistical n-grams (Chen & Goodman, 1999) and two-stage Bayesian approaches (Goldwater et al., 2006). We hope our proposed evaluation scheme and empirical contribution can help continue to drive progress on this front.

6 Conclusion

Emerging as a result of a language user's ability to produce and comprehend novel expressions, the heavy-tail of rare events is one of the fundamental features of distributions in natural language. In this work, we introduce a controlled methodology to evaluate instance-level LM performance on this set of individually rare but collectively frequent events. We use generative models trained on natural language corpora to define a set of artificial languages for which we can exactly compute the probability of sequences. Training LSTM and Transformer LMs on sequences sampled from these artificial languages, our analysis compares the probability estimates given to sequences by the LMs to the target probabilities of sequences under the artificial language.

Our results indicate that neural LMs systematically under-represent sequences in the tail of the target distribution, even when increasing the amount of the training data. Investigating where this probability mass went, our perturbation experiments reveal that neural LMs do not tend to overestimate the head of the distribution, but rather overestimate the probability of sequences outside those typical in the target distribution. Comparing model performance on target distributions with varying properties, we find that neural LMs tend to provide more accurate approximations of distributions with greater entropy. Interpreted together, these results indicate that autoregressive neural language models have a tendency to spread probability mass too uniformly across the space of possible sequences.

Finally, we would like to acknowledge that we do not know the degree of structural difference between our Transformer-generated ground-truth distributions and the distributions of actual natural languages. It is likely that the distribution defined by our ground truth models is less structured than the distribution of a natural language. Therefore, it is possible that some systematic difference between natural language distributions and our ground-truth distributions may affect our results to a certain degree. That being said, our experiments in Section 4.5 suggest that it may actually be easier for neural LMs to learn less structured distributions, and we expect the task of recovering a ground-truth distribution to be made easier when the target distribution and LM are in the same model class. Nevertheless, future work should seek to conduct similar experiments using ground-truth distributions with more explicit structure.

REFERENCES

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks, 2016.

R. Baayen. Word frequency distributions. 2001.

R. Harald Baayen. Productivity in language production. *Language and Cognitive Processes*, 9(3): 447–469, 1994. doi: 10.1080/01690969408402127. URL `https://doi.org/10.1080/01690969408402127`.

R Harald Baayen. 41. corpus linguistics in morphology: Morphological productivity. In *Corpus linguistics*, pp. 899–919. De Gruyter Mouton, 2009.

Samy Bengio. The battle against the long tail. In *Talk on Workshop on Big Data and Statistical Machine Learning*, volume 1, 2015.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The journal of machine learning research*, 3:1137–1155, 2003.

Terra Blevins and Luke Zettlemoyer. Moving down the long tail of word sense disambiguation with gloss-informed biencoders. *CoRR*, abs/2005.02590, 2020. URL `https://arxiv.org/abs/2005.02590`.

Howard Chen, Mengzhou Xia, and Danqi Chen. Non-parametric few-shot learning for word sense disambiguation. *arXiv preprint arXiv:2104.12677*, 2021.

Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394, 1999.

N. Chomsky. *The Logical Structure of Linguistic Theory*. Springer US, 1975, 1955. ISBN 9780306307607. URL `https://books.google.ca/books?id=1D66ktXOITAC`.

Paula Czarnowska, Sebastian Ruder, Edouard Grave, Ryan Cotterell, and Ann Copestake. Don't forget the long tail! a comprehensive analysis of morphological generalization in bilingual lexicon induction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 974–983, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1090. URL `https://aclanthology.org/D19-1090`.

Daniel D'souza, Zach Nussbaum, Chirag Agarwal, and Sara Hooker. A tale of two long tails. *arXiv preprint arXiv:2107.13098*, 2021.

Shiran Dudy and Steven Bedrick. Are some words worth more than others?, 2020.

Jerry A Fodor. *The language of thought*, volume 5. Harvard university press, 1975.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

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

Sharon Goldwater, Mark Johnson, and Thomas Griffiths. Interpolating between types and tokens by estimating power-law generators. In Y. Weiss, B. Schölkopf, and J. Platt (eds.), *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2006. URL `https://proceedings.neurips.cc/paper/2005/file/4b21cf96d4cf612f239a6c322b10c8fe-Paper.pdf`.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Producing power-law distributions and damping word frequencies with two-stage language models. *Journal of Machine Learning Research*, 12(68):2335–2382, 2011. URL `http://jmlr.org/papers/v12/goldwater11a.html`.

I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3/4):237–264, 1953. ISSN 00063444. URL `http://www.jstor.org/stable/2333344`.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. An empirical investigation of global and local normalization for recurrent neural sequence models using a continuous relaxation to beam search, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *CoRR*, abs/1709.01450, 2017. URL `http://arxiv.org/abs/1709.01450`.

F. Jelinek. Interpolated estimation of markov source parameters from sparse data. 1980.

S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987. doi: 10.1109/TASSP.1987.1165125.

E. Khmaladze. The statistical analysis of a large number of rare events. 1988.

R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pp. 181–184 vol.1, 1995. doi: 10.1109/ICASSP.1995.479394.

Ilia Kulikov, Sean Welleck, and Kyunghyun Cho. Mode recovery in neural autoregressive sequence modeling. *CoRR*, abs/2106.05459, 2021. URL `https://arxiv.org/abs/2106.05459`.

Heather Lent, Emanuele Bugliarello, Miryam de Lhoneux, Chen Qiu, and Anders Søgaard. On language models for creoles, 2021.

Clara Meister and Ryan Cotterell. Language model evaluation beyond perplexity, 2021.

Gábor Melis, Tomáš Kočiský, and Phil Blunsom. Mogrifier lstm. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SJe5P6EYvS`.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746–751, 2013.

Preetum Nakkiran, Behnam Neyshabur, and Hanie Sedghi. The deep bootstrap: Good online learners are good offline generalizers. *CoRR*, abs/2010.08127, 2020. URL `https://arxiv.org/abs/2010.08127`.

Yonatan Oren, Shiori Sagawa, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4227–4237, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1432. URL `https://aclanthology.org/D19-1432`.

Wanli Ouyang, Xiaogang Wang, Cong Zhang, and Xiaokang Yang. Factors in finetuning deep model for object detection with long-tail distribution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Isabel Papadimitriou and Dan Jurafsky. Pretraining on non-linguistic structure as a tool for analyzing learning bias in language models. *CoRR*, abs/2004.14601, 2020. URL `https://arxiv.org/abs/2004.14601`.

Alec Radford, Jeff Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations, 2020.

Felix Stahlberg and Bill Byrne. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3356–3362, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1331. URL `https://aclanthology.org/D19-1331`.

Shuntaro Takahashi and Kumiko Tanaka-Ishii. Do neural nets learn statistical laws behind natural language? *PloS one*, 12(12):e0189326, 2017.

Shuntaro Takahashi and Kumiko Tanaka-Ishii. Evaluating Computational Language Models with Scaling Properties of Natural Language. *Computational Linguistics*, 45(3):481–513, 09 2019. ISSN 0891-2017. doi: 10.1162/coli_a_00355. URL `https://doi.org/10.1162/coli_a_00355`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. Consistency of a recurrent language model with respect to incomplete decoding, 2020a.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2020b. URL `https://openreview.net/forum?id=SJeYe0NtvH`.

Jennifer C. White and Ryan Cotterell. Examining the inductive bias of neural language models with artificial languages. *CoRR*, abs/2106.01044, 2021. URL `https://arxiv.org/abs/2106.01044`.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL `https://aclanthology.org/2020.emnlp-demos.6`.

Frank Wood, Jan Gasthaus, Cédric Archambeau, Lancelot James, and Yee Whye Teh. The sequence memoizer. *Communications of the ACM*, 54(2):91–98, 2011.

Ruiqi Zhong, Dhruba Ghosh, Dan Klein, and Jacob Steinhardt. Are larger pretrained language models uniformly better? comparing performance at the instance level, 2021.

Xiangxin Zhu, Dragomir Anguelov, and Deva Ramanan. Capturing long-tail distributions of object subcategories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

George Kingsley Zipf. Human behavior and the principle of least effort. 1949.

A Appendix

A.1 Alternative Perturbation Experiments

In Section 4.4, we study where the language model's underestimated probability mass went by computing model estimation error on perturbed sequences. We obtain a set of perturbed sequences by (i) sampling a sequence from p_L and then (ii) recursively perturbing this sequence according to the perturbations provided in Table 3. This method provides us with sequences which are increasingly far away from high-probability zones under p_L. However, they do so with an initial sequence sampled directly from p_L, and as a result, produce strings which are edit-adjacent to the high-probability zones (under p_L) in Σ^*.

To ensure that our results hold in other low-probability subspaces, we conduct an analogous experiment on sequences randomly sampled from Σ^*. We sample a sequence from Σ^* by first sampling a sequence length l from a Poisson distribution with $\lambda = 10$. Given this length, we sample l tokens from Σ^* and concatenate all tokens to form a sequence. We then score the sequence under both p_L and p_M, and compute model estimation error. We use as artificial language p_L GPT2-medium with $T = 0.85$ and we use as language model p_M a GPT2-medium model trained on 30M sequences ancestrally sampled from p_L.

Figure 6 visualizes mean estimation error as a function of the ground-truth probability of the sequence. Similarly to all other perturbation experiments, we do indeed find that p_M overestimates these sequences, regardless of their true sequence probability.

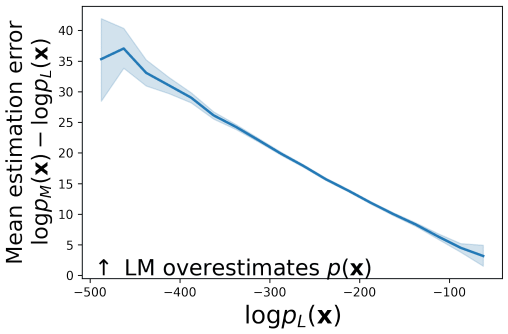


Figure 6: Mean model estimation error by true sequence probability for sequences randomly sampled from Σ^*.

A.2 Experiments with Pre-Trained Ground-Truth Model

In our previous experiments, our ground-truth model was given as a Transformer language model trainined on 1.5M sequences from the OpenWebText corpus. Here we explore underestimation behaviour when the ground-truth distribution is given by a pretrained GPT2-medium model fine-tuned on 1.5M sequences from the OpenWebText corpus. We sample from p_L with softmax $T = 1.00$.

Analogously to the experiment in Section 4.1, we train a randomly-initialized GPT2-medium model on 1M sequences sampled from the fine-tuned model. In the center of Figure 7, we visualize mean model estimation error for $50,000$ test sequences as a function of true sequence probability. Similarly to our experiments in Section 4.4, we find that the LM underestimates the probability of the majority of sequences, and does so more severely for less probable sequences. Note that this LM obtains a test perplexity of 67.97.

Finally, to ensure that our findings regarding the overestimation of ill-formed sequences hold, we compute model estimation error on random sequences sampled from Σ^* (see A.1 for details on how these sequences are constructed). Figure 7 center visualizes mean model estimation error as a function of target sequence probability. We find that p_M overestimates the majority of ill-formed sequences, indicating that these findings hold when the ground-truth distribution is defined using a pretrained model.

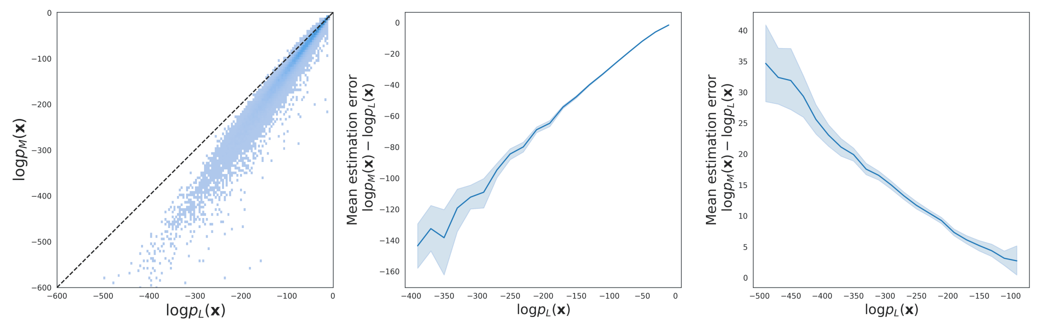


Figure 7: *Left*: Joint histogram of sequence probability estimates for test sequences. *Center*: Mean model estimation error by true sequence probability for test sequences. *Right*: Mean model estimation error by true sequence probability for sequences randomly sampled from Σ^*.

A.3 Experiments with $T = 1.00$

In Sections 4.1 to 4.4, we conducted all experiments on an artificial language defined by ancestral sampling scheme with $T = 0.85$. In Section 4.5, we saw that the underestimation findings held regardless of T. To provide further evidence that these results hold for other values of T, we conduct similar experiments as in Section 4.3 with an artificial language p_L defined by an ancestral sampling scheme with $T = 1.00$. Specifically, we train GPT2-small and an LSTM on a total of 16M sequences sampled from p_L, and we compute model estimation error on a set of withheld test sequences at each training iteration.

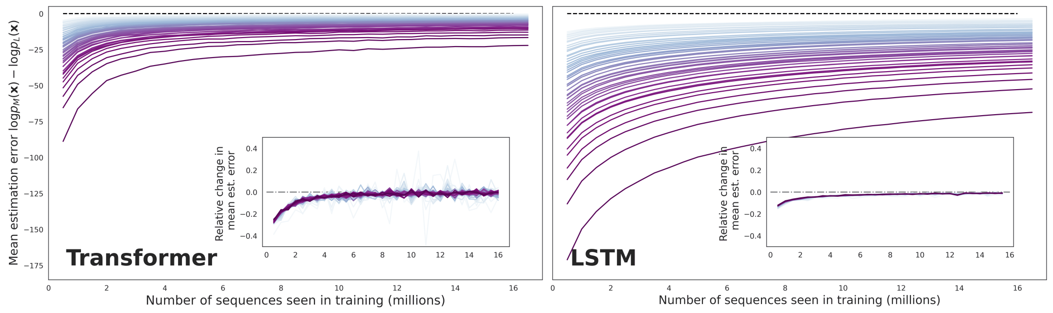


Figure 8: GPT2-small and LSTM trained on 16M sequences sampled from p_L with $T = 1.00$. *Main plots*: Mean estimation error on test sequences as a function of the number of sequences seen in training. *Inset plots*: Relative change in mean estimation error of test seqeuences as a function of the number of sequences seen in training. In both cases, each line denotes estimation behaviour for a 50th of the test sequences; darker lines represent less probable sequences.

A.4 Model Perplexity Values

In this section we report relevant perplexity values for all models used. For each model, we report mean perplexity across $125,000$ sentences drawn from (i) the validation set generated by the artificial language they attempt to model (Tables 4 and 7) and (ii) the OpenWebText corpus (Table 5).

Model	mean PP
LSTM	36.53
GPT2-small	26.66
GPT2-medium	25.42
LSTM (increased data)	35.01
GPT2-medium (increased data)	21.60

Table 4: PP on validation set generated by our artificial language with softmax $T = 0.85$. These models are used for experiments in Sections 4.1, 4.2, 4.3 and 4.4

Model	mean PP
LSTM	158.31
GPT2-small	129.84
GPT2-medium	132.56
LSTM (increased data)	149.02
GPT2-medium (increased data)	94.03
GPT2-medium (ground-truth model)	73.79

Table 5: PP on real sentences sampled from OpenWebText. These models are used for experiments in Sections 4.1, 4.2, 4.3 and 4.4

Softmax T	mean PP
0.70	11.61
0.85	25.42
1.00	75.30
1.15	290.69

Table 6: GPT2-medium PP on validation set generated by the artificial language the LM attempts to model. These models are used for the experiment in Section 4.5

Softmax T	mean PP
0.70	223.60
0.85	132.56
1.00	106.21
1.15	106.29

Table 7: GPT2-medium PP on real sentences sampled from OpenWebText. These models are used for the experiment in Section 4.5

A.5 Empirically Measuring the LNRE Zone

In section 2, we formally defined the LNRE zone as the range of values of N for which there is non-zero probability of sampling a novel event at the $N+1$th draw. Here we introduce a frequentist estimator for this probability. This in turn allows us to empirically verify if a sample exists in the LNRE zone.

Estimating $\mathcal{P}_N$ Suppose we have a set of N events $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ drawn from some generative process ϕ. Given $\mathcal{D}$, we aim to obtain an empirical estimate for $\hat{\mathcal{P}_N}$: the amount of probability allocated to unseen events as a function of N. We can do so using the Good-Turing estimate for the probability of an event given its frequency (Good, 1953).

Specifically, let $f(\boldsymbol{x}, \mathcal{D})$ be a function which returns the frequency of the event $\boldsymbol{x}$ in $\mathcal{D}$. Let N_m denote the number of types (unique events) in $\mathcal{D}$ for which $f(\boldsymbol{x}, \mathcal{D}) = m$. Good-Turing says that for large N, the probability of the event $\boldsymbol{x}$ given that it has occurred with frequency m in our sample $\mathcal{D}$ of size N is equal to

$$\hat{P}(\boldsymbol{x} \mid f(\boldsymbol{x}, \mathcal{D}) = m) = \frac{(m+1)}{N} \frac{N_{m+1}}{N_m} \tag{3}$$

To obtain an estimate for $\mathcal{P}_N$, we set $m = 0$:

$$\hat{\mathcal{P}_N} = N_0 \hat{P}(\boldsymbol{x} \mid f(\boldsymbol{x}, \mathcal{D}) = 0) = \frac{N_1}{N} \tag{4}$$

which states that the total amount of probability mass allocated to unseen events is equal to the proportion of events which occurred only once (*hapax legomena*) in $\mathcal{D}$. This quantity is known as the *potential productivity* of a linguistic process (Baayen, 2009; 2001; 1994).

We apply this method to a subset of OpenWebText, a popular language modeling corpus. In Figure 9, we plot the the empirical estimate of $\mathcal{P}_N$ as function of the sample size N for sentences sampled from a subset of OpenWebText. For comparison, we also include the potential productivity of two fair dice (uniform multinomial distributions), with increasingly large support. Whereas $\mathcal{P}$ quickly decreases to 0 in the case of fair dice, in the OpenWebText corpus, there is very high probability of observing a novel.

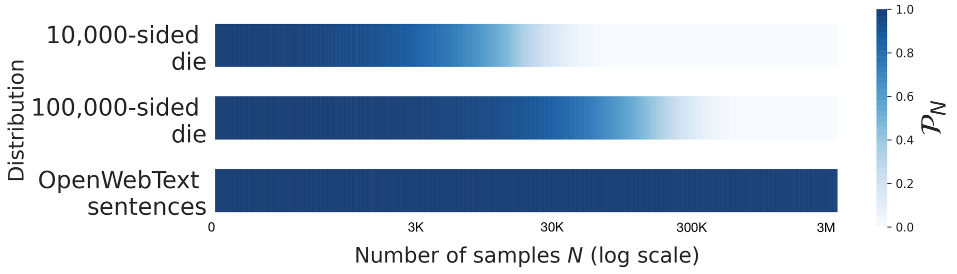


Figure 9: Productivity $\mathcal{P}$ as a function of sample size N. Distributions in natural language are characterized by a potentially unbounded amount of novelty.

A.6 Language Samples

$\boldsymbol{x}$	$\log p_L(\boldsymbol{x})$
he was a complete east end player .	-26.399
a former harvard university graduate told cnn that in recent weeks , u.s. intelligence officials have begun to gather evidence that trump 's campaign colluded with russia to influence the election .	-61.846
in the current study , we examined whether participants in the study performed more or less " active " in weight loss -lrb- p = 0.05 -rrb- .	-51.6566
you , the one that is the republican candidate , who is taking over the senate and government as a democrat and who is a bipartisan democrat , and you 've got to be able to get that done .	-92.703
since the 1970s , the city has been in the midst of a landmark urban pride .	-44.9523

Table 8: Sequences ancestrally sampled from the artificial language generated by GPT2-medium with softmax $T = 0.70$.

$\boldsymbol{x}$	$\log p_L(\boldsymbol{x})$
" i have no doubt that 's a big part of any kind of campaign machine , " he told al jazeera .	-54.5951
and it 's not a very good idea to work with claims in comics and film novels .	-59.7626
according to the new york times , susan macmahon , 54 , and her husband , elizabeth bailey , were in the vehicle with their son , aged between 12 and 19 .	-86.5217
anticipating experience	-21.7095
this is the reason i 'd like to take advantage of these ideas in an attempt to transform my life .	-53.1399

Table 9: Sequences ancestrally sampled from the artificial language generated by GPT2-medium with softmax $T = 0.85$.

$\boldsymbol{x}$	$\log p_L(\boldsymbol{x})$
beautiful bacon cookies	-19.8038
" this accident , despite most of its life , is dependent on one of the most precious bodies on this planet , which is comparable to that on mostrughenai mountains .	-142.4362
it 's not fair to say that we 're not in a position to permit same - sex marriages , and all the same issues that are presented on regular basis . "	-88.9437
from video -lrb- without editing -rrb- you can see original images produced at the official website , competitors , and event prices .	-100.3791
consumer electronics have become a perfect fit for bm 's ultra - low - cost turn .	-73.0217

Table 10: Sequences ancestrally sampled from the artificial language generated by GPT2-medium with softmax $T = 1.00$.

$\boldsymbol{x}$	$\log p_L(\boldsymbol{x})$
zy brace gym fingerprint – bom jerozo dell ’	-94.0307
shortly thereafter , will be a buoyant as the dynamo bust series starts one .	-90.5994
disjitor doom efeco became a gothicrevolution manager ofby city library for marvel studios .	-143.6075
earlier this week chicago ’s writer andrew o’t text ’ american catholics to interested readers the holiday kingsburyobee -lrb- expiration -rrb- at red for now atmotionweism race and bachelor whitney university , register as 1852 jim banner of airbus beer and of course smiled at us in formula 1 where he tells friends by name , ” finnish catholics : the encryptings on robot - type mothers .	-493.9313
middleware in germany	-27.3411

Table 11: Sequences ancestrally sampled from the artificial language generated by GPT2-medium with softmax $T = 1.15$.