THE CURSE OF RECURSION: TRAINING ON GENERATED DATA MAKES MODELS FORGET

Ilia Shumailov* University of Oxford
Zakhar Shumaylov* University of Cambridge
Yiren Zhao Imperial College London
Yarin Gal University of Oxford
Nicolas Papernot University of Toronto & Vector Institute
Ross Anderson University of Cambridge & University of Edinburgh

ABSTRACT

Stable Diffusion revolutionised image creation from descriptive text. `GPT-2`, `GPT-3(.5)` and `GPT-4` demonstrated astonishing performance across a variety of language tasks. `ChatGPT` introduced such language models to the general public. It is now clear that large language models (LLMs) are here to stay, and will bring about drastic change in the whole ecosystem of online text and images. In this paper we consider what the future might hold. What will happen to `GPT-`$\{n\}$ once LLMs contribute much of the language found online? We find that use of model-generated content in training causes irreversible defects in the resulting models, where tails of the original content distribution disappear. We refer to this effect as *model collapse*[1] and show that it can occur in Variational Autoencoders, Gaussian Mixture Models and LLMs. We build theoretical intuition behind the phenomenon and portray its ubiquity amongst all learned generative models. We demonstrate that it has to be taken seriously if we are to sustain the benefits of training from large-scale data scraped from the web. Indeed, the value of data collected about genuine human interactions with systems will be increasingly valuable in the presence of content generated by LLMs in data crawled from the Internet.

1 Introduction

A lot of human communication happens online. Billions of emails are exchanged daily, along with billions of social-media messages and millions of news articles. Almost all of this material was produced and curated only by humans in the early years of the worldwide web, yet since the turn of the century search engines have come to determine what people can find, and in the past decade smart text editors with spelling and grammar correction have helped tweak what we produce. Now, text can not only be groomed and analysed efficiently; it can also be generated – by large language models (LLMs). These models now (arguably) pass a weaker form of the Turing test in the sense that their output cannot be reliably distinguished from text written by humans [Solaiman et al., 2019].

The development of LLMs is quite involved and requires masses of training data. Anecdotally, some powerful recent models are trained using scrapes of much of the Internet, then further fine-tuned with reinforcement learning from human feedback (RLHF) [Griffith et al., 2013, OpenAI, 2023]. This further boosts the effective dataset size. Yet while current LLMs [Devlin et al., 2018, Liu et al., 2019, Brown et al., 2020, Zhang et al., 2022], including `GPT-4`, were trained on predominantly human-generated text, this may change in the future. If most future models' training data is also scraped from the web, then they will inevitably come to train on data produced by their predecessors. In this paper, we investigate what happens when text produced, *e.g.* by a version of `GPT`, forms most of the training dataset of following models. What happens to `GPT` versions `GPT-`$\{n\}$ as generation n increases?[2]

[1]The name is inspired by the Generative Adversarial Networks (GAN) literature on mode collapse, where GANs start producing a limited set of outputs that all trick the discriminator. *Model Collapse* is a process whereby models eventually converge to a state similar to that of a GAN Mode Collapse. The original version of this paper referred to this effect as 'model dementia', but we decided to change this following feedback that it trivialised the medical notion of 'dementia' and could cause offence.

[2]This is not limited to text models; one can also consider what happens when music created by human composers and played by human musicians trains models whose output trains other models.

We discover that learning from data produced by other models causes *model collapse* – a degenerative process whereby, over time, models forget the true underlying data distribution, even in the absence of a shift in the distribution over time. We give examples of *model collapse* for Gaussian Mixture Models (GMMs), Variational Autoencoders (VAE) and Large Language models (LLMs). We show that over time we start losing information about the true distribution, which first starts with tails disappearing, and over the generations learned behaviours start converging to a point estimate with very small variance. Furthermore, we show that this process is inevitable, even for cases with almost ideal conditions for long-term learning *i.e.* no function estimation error.

Finally, we discuss the broader implications of *model collapse*. We note that access to the original data distribution is crucial: in learning where the tails of the underlying distribution matter, one needs access to real human-produced data. In other words, the use of LLMs at scale to publish content on the Internet will pollute the collection of data to train them: data about human interactions with LLMs will be increasingly valuable.

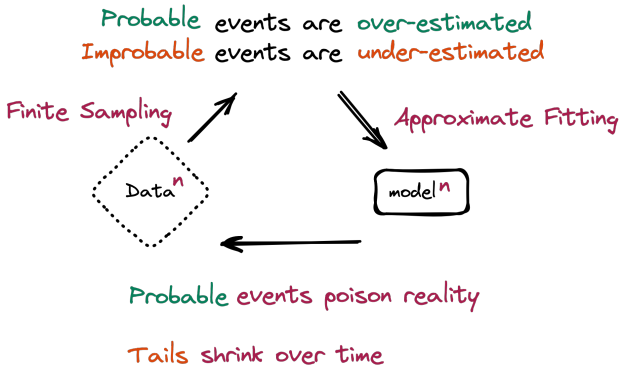


Figure 1: *Model Collapse* refers to a degenerative learning process where models start forgetting improbable events over time, as the model becomes poisoned with its own projection of reality.

This paper is structured as follows. First, in Sections 3 and 4 we describe the reasons why *model collapse* happens. To best describe the intuition, we present a simple example of a single-dimensional Gaussian where errors due to sampling inevitably cause *model collapse*, which are then extended to a multidimensional generative model under some assumptions. Under both models, similar lower bounds are derived on the risk, defined in terms of the Wasserstein distance from the true distribution. Next, we turn to GMMs and VAEs to show that additional functional approximation errors further exacerbate *model collapse*. Finally, we discuss the most commonly used setting of fine-tuned language models, where we report that only early signs of *model collapse* can be detected, if models are fine-tuned as opposed to trained from scratch.

In this paper we make the following contributions:

- We demonstrate the existence of a degenerative process in learning and name it *model collapse*;
- We demonstrate that *model collapse* exists in a variety of different model types and datasets;
- We show that, to avoid *model collapse*, access to genuine human-generated content is essential.

2 Related work

In this section we are going to cover two closest concepts to *model collapse* from existing literature: catastrophic forgetting and data poisoning. Neither is able to explain the phenomenon of *model collapse* fully, as the setting is fundamentally different, but they provide another perspective on the observed phenomenon.

2.1 Continual learning and catastrophic forgetting

Unlike traditional machine learning which seeks to learn from a static data distribution, *continual learning* attempts to learn from a dynamic one, where data are supplied in a sequential fashion [Van de Ven and Tolias, 2019]. This tends to be task-based, where data are provided with delineated task boundaries; *e.g.*, classifying dogs from cats and recognising handwritten digits. Our work is more similar to task-free continual learning [Aljundi et al., 2019] where data distributions gradually change without the notion of separate tasks. Our work examines a particular scenario in which the changed data distributions arise from the model itself, as a result of training in the previous iteration.

Example of text outputs of an OPT-125m model affected by *Model Collapse*– models degrade over generations, where each new generation is trained on data produced by the previous generation.

Input: some started before 1360 — was typically accomplished by a master mason and a small team of itinerant masons, supplemented by local parish labourers, according to Poyntz Wright. But other authors reject this model, suggesting instead that leading architects designed the parish church towers based on early examples of Perpendicular

Outputs:

Gen 0: Revival architecture such as St. John's Cathedral in London. The earliest surviving example of Perpendicular Revival architecture is found in the 18th @-@ century Church of Our Lady of Guernsey, which dates from the late 19th century. There are two types of perpendicular churches : those

Gen 1: architecture such as St. Peter's Basilica in Rome or St. Peter's Basilica in Buenos Aires. There is no evidence that any of these buildings were built during the reign of Pope Innocent III, but it is possible that they may have been built during the reign of his successor, Pope Innocent

Gen 7: architecture in England. In an interview with The New York Times, Wright said : " I don 't think there is anything wrong with me being able to do what I want to do. It just doesn 't work for me. " He added : " I don 't know if you can call it funny,

Gen 9: architecture. In addition to being home to some of the world's largest populations of black @-@ tailed jackrabbits, white @-@ tailed jackrabbits, blue @-@ tailed jackrabbits, red @-@ tailed jackrabbits, yellow @-

A typical challenge in continual learning is that the model forgets previous samples when learning new information; this is known as *catastrophic forgetting* [Kirkpatrick et al., 2017]. A typical way of preventing it is to use regularisations (Memory Aware Synpass [Aljundi et al., 2018]) or just rely on data (*e.g.* Learning without Forgetting [Li and Hoiem, 2017]). This has an indirect connection to our work, yet differs since the data in the process of *model collapse* are generated by different generations of models.

2.2 Data poisoning

Poisoning attacks are crafted and inserted during training in order to degrade the model's performance when deployed [Biggio et al., 2012]. Malicious data can be inserted into training data to induce unintended behaviors that can be activated by special triggers [Gu et al., 2017]. The early literature on data poisoning focused mainly on supervised learning, where classifiers are trained with labeled samples. But with the emergence of contrastive learning [Radford et al., 2021] and LLMs [Brown et al., 2020], more recent models are trained with large-scale web crawls, making data poisoning attacks more feasible on these untrustworthy web sources. Recent studies have demonstrated that web-scale datasets can be poisoned by introducing malicious data into a small percentage of samples [Carlini and Terzis, 2021, Carlini et al., 2023].

3 What is *Model Collapse*?

Definition 3.1 (*Model Collapse*)**.** *Model Collapse* is a degenerative process affecting generations of learned generative models, where generated data end up polluting the training set of the next generation of models; being trained on polluted data, they then mis-perceive reality. We separate two special cases: **early *model collapse*** and **late *model collapse***. In early *model collapse* the model begins losing information about the tails of the distribution; in the late *model collapse* model entangles different modes of the original distributions and converges to a distribution that carries little resemblance to the original one, often with very small variance.

Note that this process is different from the process of *catastrophic forgetting* in that we are considering multiple models over time, in which our models do not forget previously learned data, but rather start misinterpreting what they believe to be real, by reinforcing their own beliefs.

This process occurs due to two specific sources of error compounding over generations and causing deviation from the original model. Of these, one source of error plays a primary role, and in the absence of it, the process would not occur beyond the first generation.

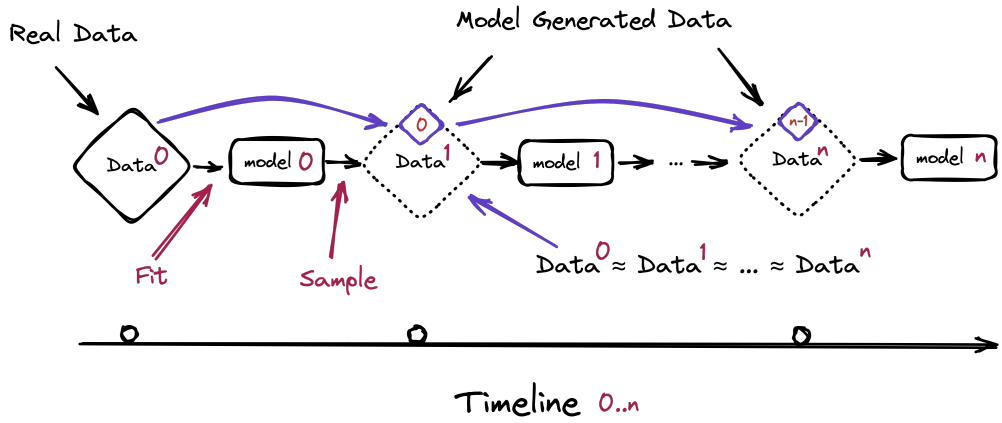


Figure 2: The high-level description of the feedback mechanism in the learning process. Here, data are assumed to be human-curated and start off clean; then model 0 is trained and data are sampled from it; at step n, data are added to the overall data from step $n-1$, and this ensemble is used to train model n. Data obtained with Monte Carlo sampling should ideally be statistically close to the original, provided *fitting* and *sampling* procedures are perfect. This process depicts what happens in real life with the Internet – model-generated data become pervasive.

3.1 Causes of *model collapse*

There are two main causes for *model collapse*, one primary and one secondary, which we describe now. Further mathematical intuition is provided in Section 4 to explain how these give rise to the errors observed, how different sources can compound and how we can quantify the average model divergence rate.

- *Statistical approximation error* – this is the primary type of error, which arises due to the number of samples being finite, and disappears as the number of samples tends to infinity. This occurs due to a non-zero probability that information can get lost at every step of re-sampling. Figure 12 shows an example of an approximation error. Here, a single-dimensional Gaussian is being approximated from a finite number of samples. Despite using a very large number of points, the errors remain significant; with 10^7 samples we estimate the mean to be $0.00024899 \pm 1.89382984e^{-4}$, when the true value is 0.

- *Functional approximation error* – this is a secondary type of error, which stems from our function approximators being insufficiently expressive (or sometimes too expressive outside of the original distribution support [Nguyen et al., 2015]). It is well known that neural networks are universal functional approximators in the limit, but in practice this is not always true. In particular, a neural network can introduce non-zero likelihood outside of the support of the original distribution. A simple example of this error is if we were to try fitting a mixture of two Gaussians with a single Gaussian. Even if we have perfect information about the data distribution, model errors will be inevitable. It is important to also note that in the absence of statistical error, functional approximation error only occurs at the first generation. Once the new distribution belongs to the image of functional approximator, it remains exactly the same over the generations.

Each of the above can cause *model collapse* to get worse or better. Better approximation power can even be a double-edged sword – better expressiveness may counteract statistical noise, resulting in a good approximation of the true distribution, but it can equally compound this noise. More often then not, we get a cascading effect where combined individual inaccuracy causes the overall error to grow. Overfitting the density model will cause the model to extrapolate incorrectly and might give high density to low-density regions not covered in the training set support; these will then be sampled with arbitrary frequency.

It is worth mentioning that modern computers also have a further computational error coming from the way floating point numbers are represented. This error is not evenly spread across different floating point ranges, making it hard to estimate the precise value of a given number. Such errors are smaller in magnitude and are fixable with more precise hardware, making them less influential on *model collapse*.

4 Theoretical intuition

In this section we aim to provide a theoretical intuition for the phenomenon of *model collapse*. We argue that the process of *model collapse* is universal among generative models that recursively train on data generated by previous generations. We construct toy mathematical models, which prove to be simple enough to provide analytical expressions for quantities of interest, but also portray the phenomenon of *model collapse*. We aim to quantify how different sources of error can affect the overall end approximation of the original distribution. As discussed in Section 3.1, there are two main sources we are interested in – *statistical* error and *functional* error. Since in the real world one rarely has infinite samples, quantifying the functional approximation error alone is of little interest for discussion of *model collapse*. Therefore, we will examine two simple cases: a discrete distribution in the absence of functional approximation error and a single dimensional Gaussian case, which portrays how functional approximation error can compound with statistical error.

The overall stochastic process we are going to be considering (which we call *Learning with Generational Data*) is the following. Assume that at generation i we have a dataset $\mathcal{D}_i$ comprising of i.i.d. random variables X_j^i, where $j \in \{1, \dots, M_i\}$ denotes the sample number at generation i and $M_i \geq 2$. We will denote the distribution of X^i as p_i. Here we assume that p_0 denotes the original distribution, from which the data comes from. Going from generation i to generation $i+1$, we aim to estimate the distribution of samples in $\mathcal{D}_i$, with an approximation $p_{\theta_{i+1}}$. This step is what we refer to as functional approximation $\mathcal{F}_\theta : p_i \to p_{\theta_{i+1}}$. We then resample the dataset $\mathcal{D}_{i+1}$ from the distribution $p_{i+1} = \alpha_i p_{\theta_{i+1}} + \beta_i p_i + \gamma_i p_0$, with non-negative parameters $\alpha_i, \beta_i, \gamma_i$ summing up to 1, *i.e.* they represent proportions of data used from different generations. This corresponds to a mixing of data coming from the original distribution (γ_i), data used by the previous generation (β_i) and data generated by the new model (α_i). We refer to this as the sampling step. For the mathematical models to come, we consider $\alpha_i = \gamma_i = 0$ *i.e.* data only from a single step is used, while numerical experiments are performed on more realistic choices of parameters.

4.1 Discrete distributions with exact approximation

In this subsection we consider a discrete probability distribution, which is represented by a histogram, *e.g.* as shown on Figure 3. In what follows we consider the stochastic process in absence of functional approximation error, *i.e.* $\mathcal{F}(p) = p$. In this case, *model collapse* arises only due to statistical errors from the sampling step. At first, the tails (low probability events) begin to disappear due to low probability of sampling them, and over time the distribution becomes a delta function. Denoting the sample size as M, if we consider state i with probability $q \leq \frac{1}{M}$, the expected number of samples with value i coming from those events will be less than 1, which means that in practice we will lose information about them. This is portrayed on Figure 3, where infrequent events get cut off. Considering more generally some state i with probability q, using standard conditional probability one can show that the probability of losing information (*i.e.* sampling no data at some generation) is equal to $1 - q$. But this in turn means that we must converge to a delta function positioned at some state, and the probability of ending up at a certain state is equal to the probability of sampling said state from the original distribution.

But how do we show directly that this process is going to turn our distribution into a delta function? By considering the process as going from $\mathbf{X}^i \to \mathcal{F}_\theta \to p_{i+1} \to \mathbf{X}^{i+1}$, we see that this forms a Markov Chain, as $\mathbf{X}^{i+1}$ only depends on $\mathbf{X}^i$. Furthermore, if all the X_j^i have the same value, then at the next generation the approximated distribution will be exactly a delta function, and therefore all of X_j^{i+1} will also have the same value. This implies that the Markov chain contains at least one absorbing state, and therefore with probability 1 it will converge to one of the absorbing states. This is a well-known fact, of which a proof is provided in Appendix A.1. For this chain, the only absorbing states are those corresponding to delta functions. As a result, as we follow the progress of *model collapse*, we are guaranteed to end up in a constant state, having lost all the information of the original distribution when the chain is absorbed.[3] Based on the discussion above we see how both early and late stage *model collapse* must arise in the case of discrete distributions with perfect functional approximation.

4.2 Single dimensional Gaussian

Following the discussion about discrete distributions, we now move on to considering how both functional approximation error and sampling error can compound (or cancel out) the process of *model collapse*.

To demonstrate this, consider a single dimensional Gaussian $X^0 \sim \mathcal{N}(\mu, \sigma^2)$. If we have full faith in the data we observe, the functional approximation involves estimating sample mean and variance and fitting a single dimensional

[3]This argument also works in general due to floating point representations being discrete, making the Markov Chain over the parameters of the model discrete. Thus as long as the model parameterisation allows for delta functions, we *will* get to it, as due to sampling errors the only possible absorbing states are delta functions.

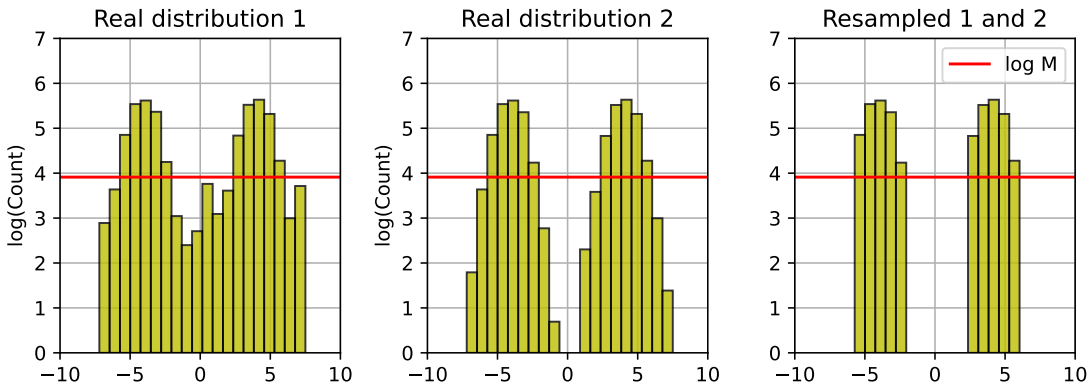


Figure 3: Shown in the middle is a histogram plot of samples from a Gaussian mixture with means $(-4, 4)$ and variances of 1. To the left of it is a similar distribution, but with 'fatter' tails, and on the right the same histograms are shown, but with low probability events being cut off due to finite resampling. Although distributions 1 and 2 are very different, when resampled (only assuming the expected behaviour), the tails get cut off, leading to the same observed distribution. In this case this is all states with probability less than $1/M$, or equivalently, bins with $\log \texttt{Count} \leq \log M$.

Gaussian. We can estimate them using the unbiased sample mean and variance estimators:

$$\mu_{i+1} = \frac{1}{M_i}\sum_j X_j^i; \quad \sigma_{i+1}^2 = \frac{1}{M_i - 1}\sum_j (X_j^i - \mu_{i+1})^2. \tag{1}$$

Note here, that if we were to use maximum likelihood estimation, we would instead arrive at a biased variance estimator. With these estimates, the functional approximation step simply corresponds to considering a normal distribution with these parameters, which we can sample from:

$$X_j^{i+1} | \mu_{i+1},\ \sigma_{i+1} \sim \mathcal{N}(\mu_{i+1}, \sigma_{i+1}^2). \tag{2}$$

This provides us with the conditional distribution of X_j^i, which allows us to calculate the full distribution of X_j^i. From Equation (3), we see that even after the first approximation, the distribution of X_j^i is no longer normal, it follows a variance-gamma distribution [Fischer et al., 2023]. However, instead of writing the probability density function at each generation, we can explicitly construct them in terms of independent random variables. In particular, it is well known [Cochran, 1934] that μ_1 and σ_1 are independent, with $\mu_1 \sim \mathcal{N}(\mu, \frac{\sigma^2}{M_0})$ and $(M_0 - 1)\sigma_1^2 \sim \sigma^2 \Gamma(\frac{M_0 - 1}{2}, \frac{1}{2})$. In what follows we will denote with Z random variables that are distributed with $\mathcal{N}(0, 1)$ and with S^i random variables that are distributed with $\frac{1}{M_{i-1}-1}\Gamma(\frac{M_{i-1}-1}{2}, \frac{1}{2})$.

$$\begin{aligned} X_j^0 &= \mu + \sigma Z_j^0; \quad X_j^1 = \mu + \frac{\sigma}{\sqrt{M_0}} Z^1 + \sigma\sqrt{S^1} Z_j^1; \quad \dots \\ X_j^n &= \mu + \frac{\sigma}{\sqrt{M_0}} Z^1 + \frac{\sigma}{\sqrt{M_1}}\sqrt{S^1} Z^2 + \dots + \frac{\sigma}{\sqrt{M_{n-1}}}\sqrt{S^1 \times \dots \times S^{n-1}} Z^n + \sigma\sqrt{S^1 \times \dots \times S^n} Z_j^n. \end{aligned} \tag{3}$$

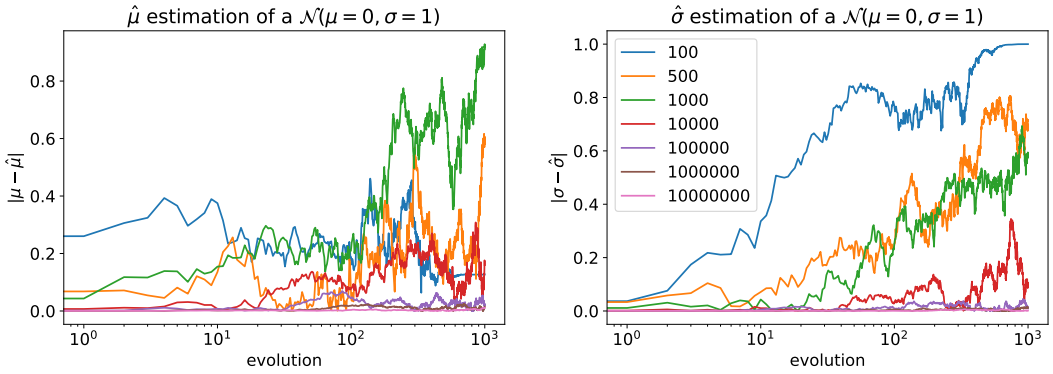
These are not joint distributions, as Z^n and S^n depend directly on Z_j^{n-1}, but when considering X_j^n on its own the formula above provides all the information about the full distribution.

The first thing we may try calculating is the variance. It is possible to find its exact value, but the mean and variance of the square root of gamma distribution are expressed in terms of gamma functions, making the result quite clunky. In what follows, we will expand everything to second order in each of $(1/M_i)$ as we assume each sample size to be large (in practice this becomes quite accurate after $M \sim 100$). We then find that

$$\frac{1}{\sigma^2}\operatorname{Var}(X_j^n) = \frac{1}{M_0} + \frac{1}{M_1} + \dots + \frac{1}{M_{n-1}} + 1 + \mathcal{O}(2).$$

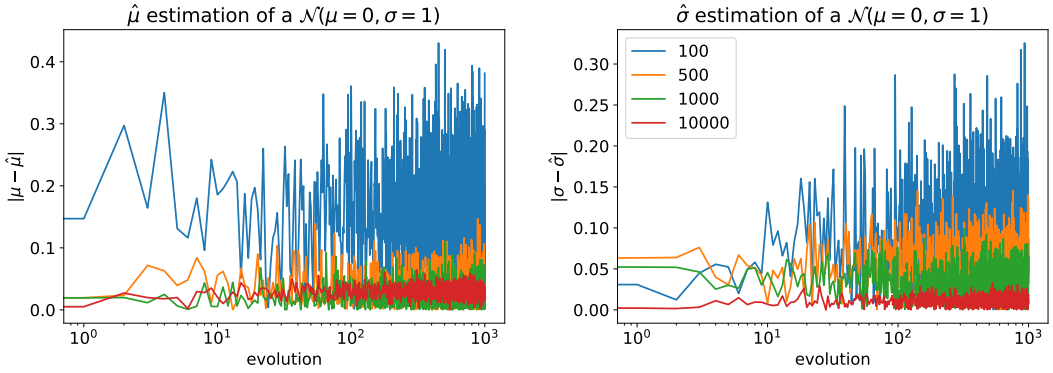
And if we were to assume that $M_i = M$ are constant, we would find that:

$$\operatorname{Var}(X_j^n) = \sigma^2\left(1 + \frac{n}{M}\right); \quad \mathbb{E}(X_j^n) = \mu.$$



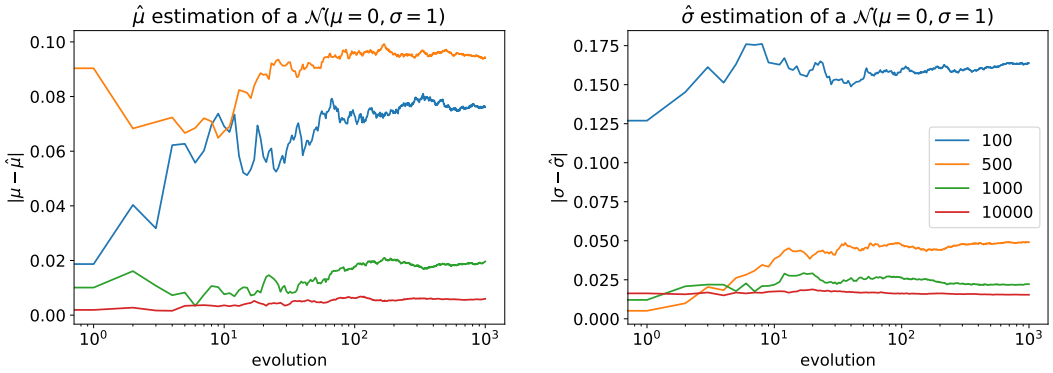
(a) Mean estimation (b) Standard Deviation

Figure 4: Recursive fitting-sampling of a 1D Gaussian with different numbers of samples drawn. We find that unless sampled a very large number of times, *i.e.* <100000, both standard deviation and mean get significantly affected. Here we report a single run; while re-running the experiment changes the initial performance, both μ and σ drift over time. The overall graph looks quite similar to that of a Gaussian random walk.



(a) Mean estimation (b) Standard Deviation

Figure 5: Recursive fitting-sampling of a 1D Gaussian with different numbers of samples drawn. In this plot data get accumulated in a pool, from which a fixed sample is drawn. In other words, a model n gets data sampled, its output is mixed with data sampled from models $1 \ldots n$, and then the mix gets sampled to fit the model $n + 1$. The uncertainty arising from all of the different modalities appearing in data causes the distribution parameters to jump around quite significantly.



(a) Mean estimation (b) Standard Deviation

Figure 6: Recursive fitting-sampling of a 1D Gaussian with different number of samples drawn. In this plot data are accumulated in a pool, all of which is used to fit a model. In other words, a model n gets data sampled, its output mixed with data sampled from models $1 \ldots n$, and then the result is used to fit the model $n + 1$. Over time the variance in estimates reduces due to linear growth of data.

This means that as $n \to \infty$, the variance diverges linearly. This is the same scaling as for a single dimensional Gaussian random walk. We can further see the similarities in numerical experiments shown on Figure 4 for a range of different sample sizes, confirming these theoretical intuitions.

Even though the variance of X_j^n diverges, it does not provide us with any information of what the corresponding estimates of μ_{n+1} and σ_{n+1}^2 are, or how far they are from the original μ and σ. In particular, we may want to consider what the distance would be between the true distribution and the approximated distribution at step $n+1$. To measure this we can consider the Wasserstein-2 distance between two normals:

$$R_{W_2}^{n+1} := W_2^2\left(\mathcal{N}(\mu, \sigma^2), \mathcal{N}(\mu_{n+1}, \sigma_{n+1}^2)\right) = \|\mu_{n+1} - \mu\|^2 + \|\sigma_{n+1} - \sigma\|^2$$

Now we can calculate the risk that occurs due to finite sampling, *i.e.* what the expected value of the distance is (expanding in $1/M_i$):

$$\mathbb{E}_{\mu_{n+1}, \sigma_{n+1}^2}\left[R_{W_2}^{n+1}\right] = \sigma^2\left(\frac{1}{M_0} + \frac{1}{M_1} + \cdots + \frac{3}{2M_n}\right) + \mathcal{O}(2), \tag{4}$$

$$\mathrm{Var}_{\mu_{n+1}, \sigma_{n+1}^2}\left[R_{W_2}^{n+1}\right] = \sigma^4\left(\frac{2}{M_0^2} + \frac{2}{M_1^2} + \cdots + \frac{3}{M_n^2} + \sum_{i \neq j}\frac{3}{M_i M_j}\right) + \mathcal{O}(3). \tag{5}$$

This result allows us to interpret exactly what occurs in this formulation of *model collapse*. To be precise, due to errors occurring from re-sampling the approximated distribution, each generation ends up corresponding to a new step in a random walk of model parameters. The risk that occurs in this model ends up diverging for a constant sample size at each generation. In order for the end distribution approximation to be accurate, and for the distance to be finite, the sampling rate M_i needs to increase superlinearly, *i.e.* one needs to collect increasingly more samples over time, perhaps quadratically. However, even in that case the expected distance after n steps remains non-zero and the only case in which it does in fact end up being 0 is when sampling is infinite at each step. Overall, this only shows us how far on average we go from the original distribution, but the process can only 'terminate' if the estimated variance at a certain generation becomes small enough, *i.e.* we effectively turn into a delta function.

Shown on Figures 5 and 6 are different runs of this process for different values of parameters of $\alpha_i, \beta_i, \gamma_i$ for different sample sizes, which was investigated numerically to see whether they can be enough to overcome *model collapse*, however even in those cases the changes are inevitable, although attenuated.

4.3 Noisy approximation model

With the simple example out of the way, we can now construct a lower bound on the distance of generation n distribution from the original and show that without superlinear sampling it similarly diverges in the limit of large n. A nice property of Wasserstein-2 distance is that Gaussians provide a universal lower bound for the Wasserstein distance [Gelbrich, 1990]. In particular, for κ and ν probability measures on a Euclidean N-dimensional space with μ_κ and μ_ν means, Σ_κ and Σ_ν covariance matrices, we have that

$$W_2^2(\kappa, \nu) \geq \|\mu_\kappa - \mu_\nu\|^2 + \mathrm{Tr}\left(\Sigma_\kappa + \Sigma_v - 2\left(\Sigma_\kappa^{1/2}\Sigma_v\Sigma_\kappa^{1/2}\right)^{1/2}\right) \geq \|\mu_\kappa - \mu_\nu\|^2$$

With this, instead of quantifying the distance exactly, we can instead lower bound it. The only limitation is that we are going to have to specify a functional approximation model. In order to achieve a W_2 bound, we will be required to specify how the mean changes between generations. In the scenario where we only have access to the sample mean, we would approximate the mean of the next generation distribution as Equation (1). However, as more information arrives, or the model begins using it better, we may end up diverging from the sample mean. We would still require that the model have good performance, *i.e.* on average the mean estimate is the same. We will also have to specify expected behaviour of the model over the the variance calculation, which once again will be chosen such that it averages out. Thus, we will adopt the following evolution over generations:

$$\mu_{i+1} = \frac{1}{M_i}\sum_j X_j^i + \varepsilon_{i+1} = \frac{\Sigma_i^{1/2}}{\sqrt{M_i}}T^{i+1} + \mu_i + \varepsilon_{i+1}; \quad \mathbb{E}_{X_j^i}(\Sigma_{i+1}) = \Sigma_i \tag{6}$$

where we define T^{i+1} to satisfy the equation above, *i.e.* $T^{i+1} = \frac{\Sigma_i^{-1/2}}{\sqrt{M_i}}\sum_j\left(X_j^i - \mu_i\right)$. With this normalisation T has mean 0 and covariance I_N and by the central limit theorem (CLT) we would have $T^{i+1}|\mu_i, \Sigma_i \xrightarrow{\mathcal{D}} \mathcal{N}(0, I_N)$, however the lower bound will not rely on this. To arrive at a lower bound for the risk, similar to that of Equation (4), we are going to have to make a few assumptions about the form of ε_{i+1}.

Assumptions:

1. On average we can capture the mean to be the same as at the iteration prior:
$$\mathbb{E}[\varepsilon_{i+1}|\mu_i, \Sigma_i] = 0 \tag{7}$$
2. Given all of X_j^i, epsilon must be constant, ***i.e.*** it is a function of only the data:
$$\varepsilon_{i+1} = \varepsilon_{i+1}\left(X_j^i\right) \tag{8}$$
In particular, it is dependent on μ_i and Σ_i only through the data.
3. The extra noise is orthogonal to the sample mean in the sense of random variables. This is effectively assuming that the noise does not contain any first moment information, ***i.e.*** we have:
$$\mathrm{Cov}(\varepsilon_{i+1}, T^{i+1}|\mu_i, \Sigma_i) = 0 \tag{9}$$
This may seem like a rather strong assumption, compared to the previous ones, however this property can be shown to hold true when imposing CLT on T^{i+1} in the limit of large M_i (note here that M_i can only be assumed to be **large**, and not infinite) and assuming that ε is strictly a function of moments higher than first. Furthermore, a property of this type is necessary to actually provide any information, since prior to it there would be no need to separate into an epsilon term and a sample mean term, since all could be absorbed into ε.

In Appendix A.2, we further provide an alternative to Assumption 3, wherein by bounding the size of noise we are able to recover a similar bound, but only as an expansion in $1/M_i$.
With all the assumptions in place, we now have the following bound:

$$\mathbb{E}\left[R_{W_2}^{i+1}\right] \geq \mathbb{E}\left(\|\mu_{i+1} - \mu\|^2\right) \tag{10}$$
$$= \mathbb{E}\left(\|\mu_i - \mu\|^2\right) + \mathbb{E}\left(\|\varepsilon_{i+1}\|^2\right) + \frac{1}{M_i}\mathbb{E}\left((T^{i+1})^\top \Sigma_i (T^{i+1})\right) + \tag{11}$$
$$+ \frac{2}{\sqrt{M_i}}\mathbb{E}\left((\varepsilon_{i+1})^\top \Sigma_i^{1/2} T^{i+1} + (\mu_i - \mu)^\top \Sigma_i^{1/2} T^{i+1}\right) \tag{12}$$
$$= \mathbb{E}\left(\|\mu_i - \mu\|^2\right) + \frac{\mathrm{Tr}\,\Sigma}{M_i} + \mathbb{E}\left(\|\varepsilon_{i+1}\|^2\right) + \frac{2}{\sqrt{M_i}}\mathbb{E}\left((\varepsilon_{i+1})^\top \Sigma_i^{1/2} T^{i+1}\right) \tag{13}$$

Now the only quantity to evaluate is

$$\frac{2}{\sqrt{M_i}}\mathbb{E}\left((\varepsilon_{i+1})^\top \Sigma_i^{1/2} T^{i+1}\right) = \frac{2}{\sqrt{M_i}}\int d\Sigma_i\, p(\Sigma_i)\,\mathrm{Tr}\left[\Sigma_i^{1/2}\,\mathrm{Cov}(\varepsilon_{i+1}, T^{i+1}|\Sigma_i)\right] = 0, \tag{14}$$

by Assumption 3. Therefore, the overall bound would be similar to the Gaussian case, but with extra noise variance terms:

$$\mathbb{E}_{\mu_{n+1},\sigma_{n+1}^2}\left[R_{W_2}^{n+1}\right] \geq \mathrm{Tr}\,\Sigma\left(\frac{1}{M_0} + \frac{1}{M_1} + \cdots + \frac{1}{M_n}\right) + \sum_{i=1}^{n+1}\mathbb{E}\left(\|\varepsilon_i\|^2\right) \tag{15}$$

As a result, we have shown that the same superlinear scaling would be required to minimise the lower bound on *model collapse* even in the case of more generic models of approximation, in which the mean at step $i+1$ can be separated orthogonally into the sample mean and 'extra'.

Overall, the message of this section can be summarised as follows:

*When learning on generational data, due to finite sampling, we are only able to **approximate** the original distribution. While on average we should recover the original distribution, the variance arising from this is non-zero. As a result, over the generations, the average distance of n'th generation from the original grows and can become infinite in the limit since errors compound over time.*

5 Evaluation

5.1 Training from scratch with GMMs and VAEs

Gaussian Mixture Models. In this subsection we evaluate the performance of Gaussian Mixture Models (GMM) [Reynolds et al., 2009]. The underlying task here is that a given GMM tries to separate two artificially-generated Gaussians. Figure 7 shows the progression of the GMM fitting process over time. The left-most plot shows the original two Gaussians with the ground truth labels. The next plot shows the GMM fitted on the original data with no cross-generational data used *i.e.* $\alpha_i = \gamma_i = 0$, where the error is minimal. Yet, within 50 iterations of re-sampling we arrive to a point where the underlying distribution is mis-perceived. The performance worsens over time and by iteration 2000 we arrive at a point estimate of the distribution with very little variance. The L2 distance between the original GMM and its descendants is plotted in Figure 13.

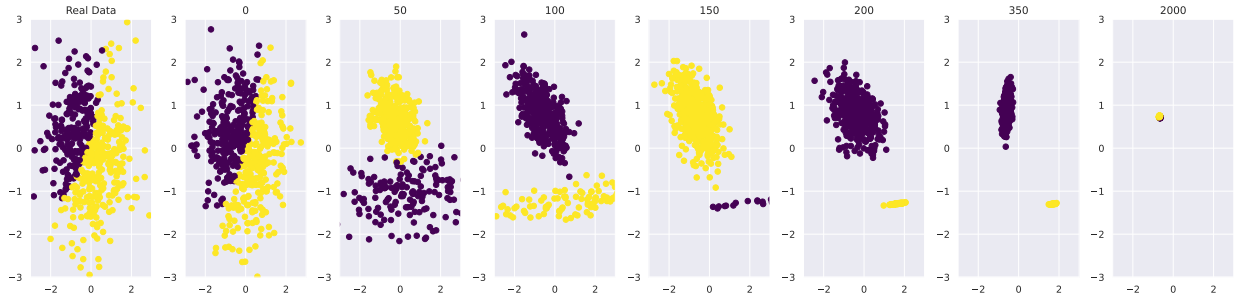
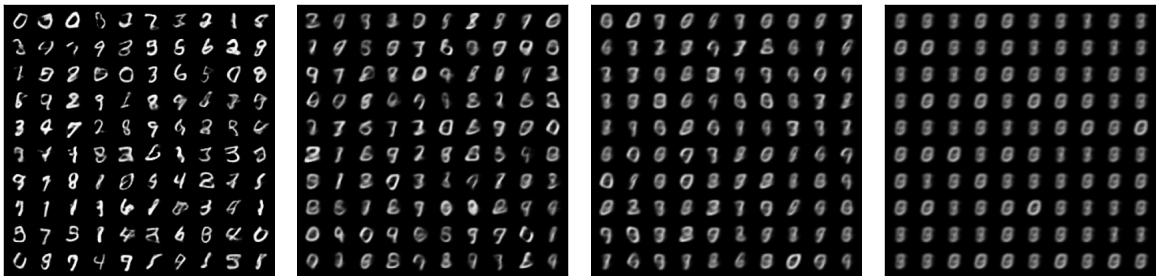


Figure 7: An examples of GMM fitting data at iterations $\{0, 50, 100, 150, 200, 350, 2000\}$. At first the model fits data very well as is shown on the left; yet even at generation 50 the perception of the underlying distribution completely changes. At generation 2000 it converges to a state with very little variance. GMM is sampled a thousand times.



(a) Original model (b) Generation 5 (c) Generation 10 (d) Generation 20

Figure 9: Random latent reconstructions from VAEs. No training data comes from the original distribution. Over the generations, different modes of the original distribution get entangled and generated data starts looking unimodal.

Variational Autoencoders. In this subsection we turn to Variational Autoencoders (VAE). As before, we train an autoencoder on an original data source, which we later sample. Here, we generate latents from a Gaussian distribution which are then used by the decoder to generate data for the subsequent generation. Figure 9 on the left shows an example of generated data using the setting described by Kingma and Welling.

Having performed the process a number of times we arrive at a representation that has very little resemblance of the original classes learned from data. On the right, one sees the generated images from generation 20, which appear to be a mix of all of the different digits. Interestingly, the original encoder perceives the generated data from its descendant with ever-growing confidence – the encoder places such data closer and closer to the mean. Figure 8 shows the density of the latent representation of the original model when presented with data generated by its descendants. As with single-dimensional Gaussians, tails disappear over time and all of the density shifts towards the mean.

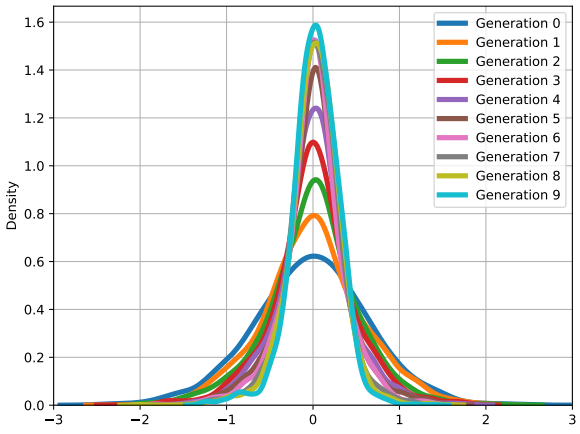
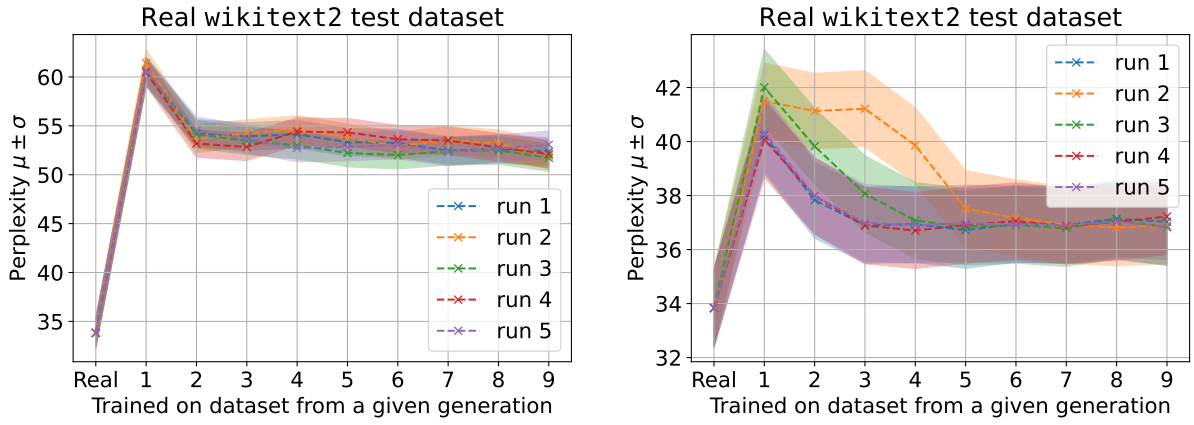


Figure 8: Changing distribution of latents over the learning process with generated data as perceived by the original encoder. Just as with the Gaussian case described above, the tails get washed away and the model arrives at the mean representation of the underlying data.

5.2 Language Models

By now it is clear that *Model Collapse* is universal across different families of ML models. Yet if small models such as GMMs and VAEs are normally trained from scratch, LLMs are different. They are so expensive to retrain from scratch that they are typically initialised with pre-trained



(a) No data preserved, 5 epochs

(b) 10% data preserved, 10 epochs

Figure 10: Performance of OPT-125m models of different generations evaluated using the original `wikitext2` test dataset. Perplexity is shown on the y-axis and for each independent run the graph of the mean and its standard deviation is shown with error bars. x-axis refers to the generation of the model – 'Real' refers to the 'model 0' trained on the original `wikitext2` dataset; model 1 was trained on the data produced by model 0; model 2 was trained on data produced by model 1 etc. with all generated datasets equal in size. We find that models trained on generated data are able to learn some of the original task, but with errors, as seen from the increase in perplexity.

models such as `BERT` [Devlin et al., 2018], `RoBERTa` [Liu et al., 2019], or `GPT2` [Brown et al., 2020], which are trained on large text corpora. They are then fine-tuned to various downstream tasks [Bommasani et al., 2022].

In this subsection we explore what happens with language models when they are sequentially fine-tuned with data generated by other models[4]. We evaluate the most common setting of training a language model – a fine-tuning setting where each of the training cycles starts from a pre-trained model with recent data. Data here comes from another fine-tuned pre-trained model. Since training is restricted to produce models that are close to the original pre-trained model and datapoints generated by the models will generally produce very small gradients, the expectation here may be that the model should only change moderately after fine-tuning. We fine-tune the `OPT-125m` causal language model made available by Meta through `Huggingface` [Zhang et al., 2022].

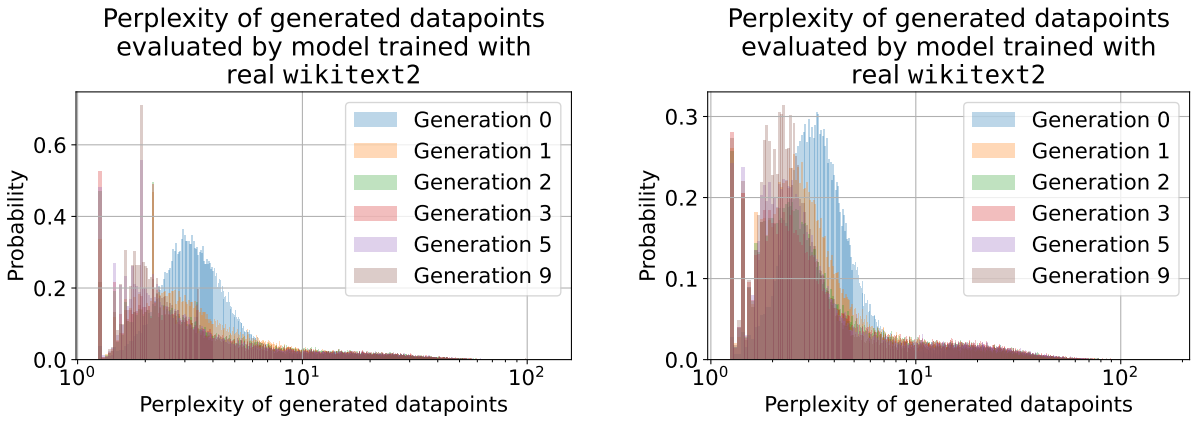
We fine-tune the model on the `wikitext2` dataset. For data generation from the trained models we use a 5-way beam-search. We block training sequences to be 64 tokens long; then for each token sequence in the training set, we ask the model to predict the next 64 tokens. We go through all of the original training dataset and produce an artificial dataset of the same size. Since we go though all of the original dataset and predict all of the blocks, if the model had 0.0 error it would produce the original `wikitext2` dataset. Training for each of the generations starts with generation from the original training data. Each experiment is ran 5 times and the results are shown as 5 separate runs. The original model fine-tuned with real `wikitext2` data gets 34 mean perplexity, from the zero-shot baseline of 115, *i.e.* it successfully learns the task. Finally, to be as realistic as possible, we use the best performing model on the original task, evaluated using the original `wikitext2` validation set, as the base model for the subsequent generations, meaning in practice observed *Model Collapse* can be even more pronounced.

Here we consider two different settings:

5 epochs, no original training data – Here, the model is trained for 5 epochs on the original dataset and no original data. The overall original task performance is presented in Figure 10.(a). We find that training with generated data allows one to adapt to the underlying task, losing some performance – from 20 to 28 perplexity points.

10 epochs, 10% of original training data preserved – Here the model is trained for 10 epochs on the original dataset and every new generation of training, a random 10% of the original data points are sampled. The overall original

[4]One can easily replicate an experiment described in Section 5.1 with a language model to demonstrate *model collapse*. Given that training a single moderately large model produces twice the American lifetime worth of CO_2 [Strubell et al., 2019], we opted to not run such an experiment and instead focus on a more realistic setting for a proof-of-concept. Note that just the language experiments described in the paper took weeks to run.



(a) No data preserved

(b) 10% data preserved

Figure 11: Histograms of perplexities of each individual data training sequence produced by different generations as is evaluated by the very first model trained with the real data. Over the generations models tend to produce samples that the original model trained with real data is more likely to produce. At the same time, a much longer tail appears for later generations – later generations start producing samples that would never be produced by the original model *i.e.* they start misperceiving reality based on errors introduced by their ancestors. Same plots are shown in 3D in Figure 15.

task performance is presented in Figure 10.(b). We find that preservation of the original data allows for better model fine-tuning and leads to only minor degradation of performance.

Both training regimes lead to degraded performance in our models, yet we do find that learning with generated data is possible and models can successfully learn (some of) the underlying task. We now turn to consider the underlying perception of probable events for each generation of our models.

Figure 11 shows histograms of individual datapoint perplexities generated by the models of different generations as is evaluated by the first model developed with real `wikitext2` training data. Here over the generations models tend to produce more sequences that the original model would produce with the higher likelihood. The observed effect is similar to that described for VAEs and GMMs in Section 5.1, where over the generations models started to produce samples that would be produced with higher probabilities by the original model. At the same time, we discover that generated data has much longer tails, suggesting that some of the data would never be produced by the original model – these are the errors that accumulate because of the *learning with generational data*.

We find that data generated by language models in our experiments end up containing a large number of repeating phrases. The repeating problem has been observed in nearly all text generation models [Keskar et al., 2019, Shumailov et al., 2021] and to rule this out as the cause of *Model Collapse*, we further provide numerical experiments when models are explicitly encouraged to produce non-repeating sequences with repeating penalty of 2.0. We find that this causes the models to produce lower score continuations to avoid using repeats, which as a result causes the consequent models to perform even worse. Figure 14 show model perplexities shift across the generations towards more probable token sequences. In particular, enforcing this for the LLM experiments causes the perplexity to double, compared to the original. Models remain as susceptible to *Model Collapse*, if not more.

The described process demonstrates that fine-tuning of language models does not curb the effects of *Model Collapse* and models that are being fine-tuned are also vulnerable. We find that over the generations models tend to produce more probable sequences from the original data and start introducing their own improbable sequences *i.e.* errors.

6 Discussion and Conclusion

We now discuss the implications of *Model Collapse* on the underlying learning dynamics of LLMs. Long-term poisoning attacks on language models are not new. For example, we saw the creation of *click, content, and troll* farms – a form of human 'language models', whose job is to misguide social networks and search algorithms. The negative effect these poisoning attacks had on search results led to changes in search algorithms: *e.g.*, Google downgraded

farmed articles[5], putting more emphasis on content produced by trustworthy sources *e.g.* education domains, while DuckDuckGo removed them altogether[6].

What is different with the arrival of LLMs is the scale at which such poisoning can happen once it is automated. Preserving the ability of LLMs to model low-probability events is essential to the fairness of their predictions: such events are often relevant to marginalised groups. Low-probability events are also vital to understand complex systems [Taleb, 2007].

Our evaluation suggests a "first mover advantage" when it comes to training models such as LLMs. In our work we demonstrate that training on samples from another generative model can induce a distribution shift, which over time causes *Model Collapse*. This in turn causes the model to mis-perceive the underlying learning task. To make sure that learning is sustained over a long time period, one needs to make sure that access to the original data source is preserved and that additional data not generated by LLMs remain available over time. The need to distinguish data generated by LLMs from other data raises questions around the provenance of content that is crawled from the Internet: it is unclear how content generated by LLMs can be tracked at scale. One option is community-wide coordination to ensure that different parties involved in LLM creation and deployment share the information needed to resolve questions of provenance. Otherwise, it may become increasingly difficult to train newer versions of LLMs without access to data that was crawled from the Internet prior to the mass adoption of the technology, or direct access to data generated by humans at scale.

Acknowledgements

We want to thank Anvith Thudi, David Glukhov, Peter Zaika, and Darija Barak for useful discussions and feedback.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11254–11263, 2019.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. *arXiv preprint arXiv:2106.09667*, 2021.

[5] `https://googleblog.blogspot.com/2011/02/finding-more-high-quality-sites-in.html`

[6] `https://www.technologyreview.com/2010/07/26/26327/the-search-engine-backlash-against-content-mills/`

Nicholas Carlini, Matthew Jagielski, Christopher A Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. *arXiv preprint arXiv:2302.10149*, 2023.

W. G. Cochran. The distribution of quadratic forms in a normal system, with applications to the analysis of covariance. *Mathematical Proceedings of the Cambridge Philosophical Society*, 30(2):178–191, 1934. doi: 10.1017/S0305004100016595.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Adrian Fischer, Robert E. Gaunt, and Andrey Sarantsev. The variance-gamma distribution: A review, 2023.

Matthias Gelbrich. On a formula for the l2 wasserstein metric between measures on euclidean and hilbert spaces. *Mathematische Nachrichten*, 147(1):185–203, 1990.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL `https://proceedings.neurips.cc/paper_files/paper/2013/file/e034fb6b66aacc1d48f445ddfb08da98-Paper.pdf`.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images, 2015.

OpenAI. Gpt-4 technical report, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 212–231. IEEE, 2021.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. Release strategies and the social impacts of language models, 2019.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

Nassim Nicholas Taleb. Black swans and the domains of statistics. *The American Statistician*, 61(3):198–200, 2007.

Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

A Appendix

A.1 Absorbing Markov Chain

The subsection explains a well-known fact about absorbing Markov chains, that they converge to an absorbing state with probability one. Assume that $\mathbf{X}^m$ form a Markov chain. In order to reason about this chain we need to consider the transition probabilities. In general, these correspond to our functional approximation scheme. Due to the stochastic nature of the Markov chain, we expect to have the variance go up and down. But as the variance decreases, the newly sampled data, due to its finiteness, will be more concentrated, leading in the limit to a set of *i.e.* a delta functions. This argument assumes that the approximation scheme is good and can converge to delta functions. If not, the errors in approximation may prevent the propagation of errors in stochasticity.

As discussed in the previous section, we can model the process of repeated 'sampling' and 'fitting' as a Markov chain. In this subsection, we explain how such a process can converge to a stationary state *i.e.* the absorbing state of a Markov Chain. In this derivation we follow Allan Yashinski [7]. Suppose we have an absorbing Markov Chain with r transient states $t_1, \ldots, t_r$ and s absorbing states $a_1, \ldots, a_s$. The whole Markov chain has $r + s$ states, ordered as follows: $t_1, \ldots, t_r, a_1, \ldots, a_s$. The transition matrix is then defined as

$$T = \begin{bmatrix} Q & 0_{r\times s} \\ R & I_s \end{bmatrix}, \tag{16}$$

where

- Q is an $r \times r$ matrix holds the probabilities of moving from a transient state to another transient state
- R is an $s \times r$ matrix which holds the probabilities of moving from a transient state to an absorbing state.
- $0_{r\times s}$ is the $r \times s$ matrix of all 0's. There 0's represent the probabilities of moving from an absorbing state to a transient state (which is impossible by definition).
- I_s holds the probabilities of transitioning between the absorbing states. As transition is impossible, this is just the $s \times s$ identity matrix.

We are interested in $\lim_{k\to\infty} T^k(\mathbf{X}_0)$. For a given k, the matrix becomes

$$T^k = \begin{bmatrix} Q^k & 0_{r\times s} \\ R + RQ + \cdots + RQ^{k-1} & I_s \end{bmatrix} = \begin{bmatrix} Q^k & 0_{r\times s} \\ R\sum_{i=0}^{k-1} Q^i & I_s \end{bmatrix}. \tag{17}$$

Finally, for an absorbing Markov chain with $T = \begin{bmatrix} Q & 0_{r\times s} \\ R & I_s \end{bmatrix}$,

we have $\lim_{k\to\infty} T^k = \begin{bmatrix} 0_{r\times r} & 0_{r\times s} \\ R(I_r - Q)^{-1} & I_s \end{bmatrix}$.

Since in the limit the transition probabilities to transient states are zero, we end up converging to absorbing states and staying there. In the case of discrete distributions, where we can perfectly approximate a zero-variance dataset (*i.e.* a delta function), the absorbing states are delta functions centered at any non-zero probability point from the original distribution. In practice, we would like to know the expected number of steps before being absorbed, which may be large. But without knowing our fitting procedure it is impossible to calculate the matrix Q and therefore the average length of time before collapse.

A.2 Alternative assumption for noisy approximations

This subsection will cover an alternative assumption, which may be more realistic in **some** settings, in contrast to assumption 3 from Section 4.3, and this subsection mostly acts as an extension, rather than an alternative. In particular, instead of imposing orthogonality, we can instead impose a certain size requirement on the noise term. This in turn allows us to arrive to a similar result.

To be more precise, we will consider the same setting as in Section 4.3, but we will now replace Assumption 3 with Assumption 3*:

[7] `www.math.umd.edu/~immortal/MATH401/book/ch_absorbing_markov_chains.pdf`

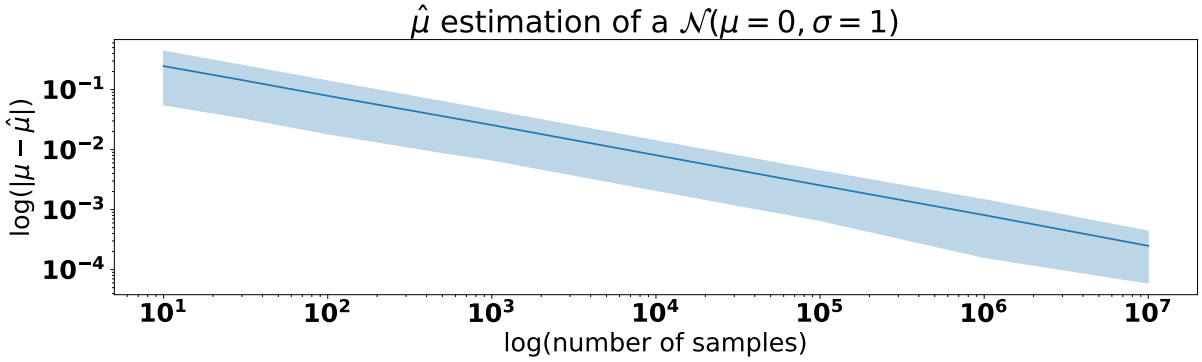


Figure 12: Approximation of a single-dimensional Gaussian $\mathcal{N}(0, 1)$ as a function of number of points. The mean estimator and its standard deviation are calculated from running the procedure 10000 times.

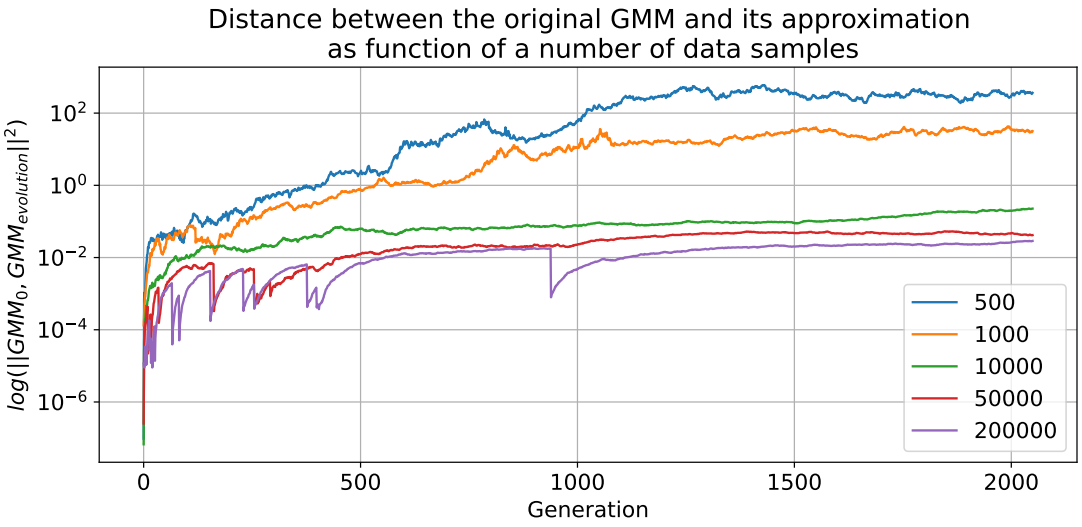


Figure 13: Progressive fitting of a GMM with different number of samples. On the y-axis is shown the logarithm of $L2$ distance between the two GMM distributions. Over the generations the distance begins to grow and can become quite large. The jumps in the distance for large sample sizes occur due to the fixed number of iterations and precision for the expectation maximization algorithm.

Assumptions:

3*. The extra noise is going to be assumed to be bounded and of the order larger than the sample mean deviation. To be precise we will have a constant K (not dependent on generation i), such that for all i:

$$\|\varepsilon_{i+1}\| \leq \frac{K}{M_i} \tag{18}$$

Now with the alternative assumption in place, we can follow the exact same calculations to arrive at

$$\mathbb{E}\left[R_{W_2}^{i+1}\right] \geq \mathbb{E}\left(\|\mu_i - \mu\|^2\right) + \frac{\operatorname{Tr}\Sigma}{M_i} + \mathbb{E}\left(\|\varepsilon_{i+1}\|^2\right) + \frac{2}{\sqrt{M_i}}\mathbb{E}\left(\left(\varepsilon_{i+1}\right)^\top \Sigma_i^{1/2} T^{i+1}\right) \tag{19}$$

Similar to before, we need to evaluate (which we instead bound this time):

$$\frac{2}{\sqrt{M_i}}\mathbb{E}\left(\left(\varepsilon_{i+1}\right)^\top \Sigma_i^{1/2} T^{i+1}\right) = \frac{2}{\sqrt{M_i}} \int d\Sigma_i \; p(\Sigma_i) \operatorname{Tr}\left[\Sigma_i^{1/2} \operatorname{Cov}(\varepsilon_{i+1}, T^{i+1}|\Sigma_i)\right] \neq 0 \tag{20}$$

$$\geq -\frac{2\sqrt{N}}{\sqrt{M_i}} \int d\Sigma_i \; p(\Sigma_i) \sqrt{\operatorname{Tr}\left[\Sigma_i \Sigma_{\epsilon_{i+1}}\right]} \tag{21}$$

$$\geq -\frac{2\sqrt{N}}{\sqrt{M_i}} \sqrt{\mathbb{E}\left(\varepsilon_{i+1}^\top \Sigma_i \varepsilon_{i+1}\right)}, \tag{22}$$

$$\geq -\frac{2\sqrt{N}}{\sqrt{M_i}} \sqrt{\frac{K^2 \operatorname{Tr}\Sigma}{M_i^2}} = \frac{-2K\sqrt{N}}{M_i\sqrt{M_i}} \sqrt{\operatorname{Tr}\Sigma}, \tag{23}$$

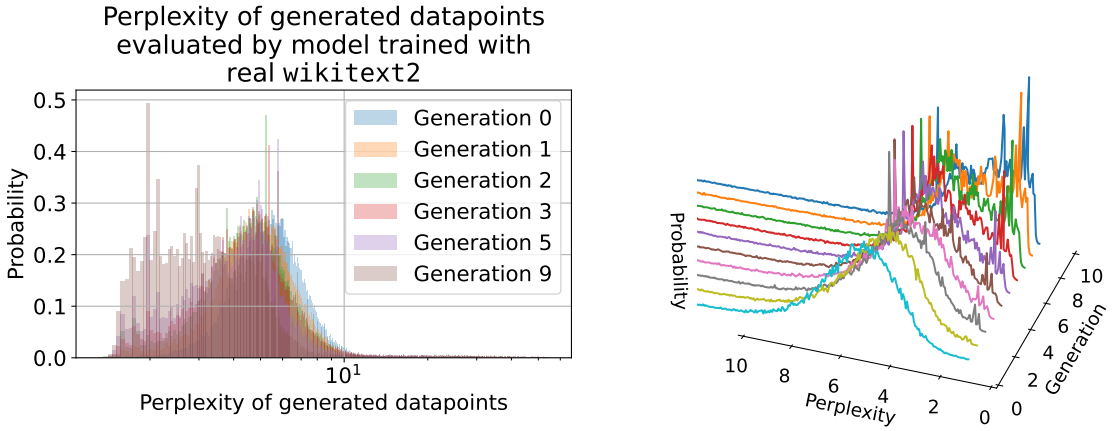
where we used the Cauchy-Schwarz and Jensen inequalities. Note that this is far from optimal inequality, since instead of using the expected value of the largest eigenvalue, we instead bounded it by $\operatorname{Tr}\Sigma$. In particular, the per step bound is then:

$$\mathbb{E}\left[R_{W_2}^{i+1}\right] \geq \mathbb{E}\left(\|\mu_i - \mu\|^2\right) + \frac{\operatorname{Tr}\Sigma}{M_i} + \mathbb{E}\left(\|\varepsilon_{i+1}\|^2\right) - \frac{2K\sqrt{N}}{M_i\sqrt{M_i}} \sqrt{\operatorname{Tr}\Sigma}. \tag{24}$$

Without knowledge of the specific values of K, N or $\operatorname{Tr}\Sigma$, the best we can do is consider what this means for the bound as M_i becomes large. In particular, contribution from the last two terms will be of order at most $3/2$. As a result we recover a bound similar to all of the ones observed so far:

$$\mathbb{E}_{\mu_{n+1},\sigma_{n+1}^2}\left[R_{W_2}\right] \geq \operatorname{Tr}\Sigma \left(\frac{1}{M_0} + \frac{1}{M_1} + \cdots + \frac{1}{M_n}\right) + \mathcal{O}(3/2) \tag{25}$$

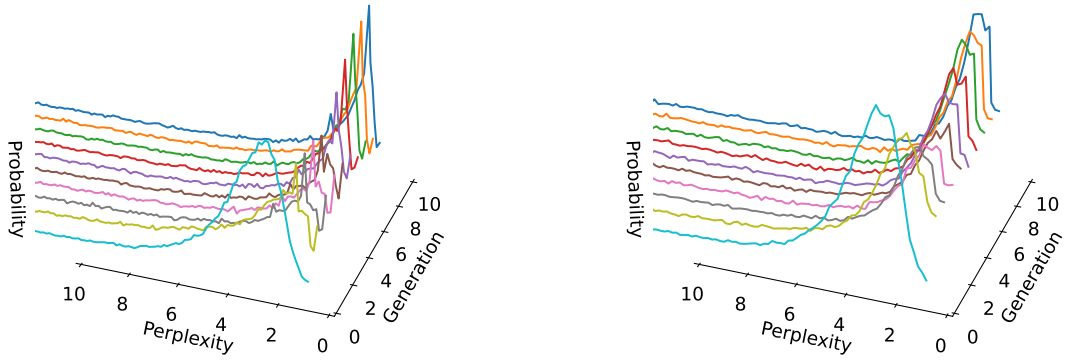
In particular, we find in the same way, that superlinear scaling would be required to minimise the lower bound on *model collapse* even in the case of more generic models of approximation, in which the mean at step $i+1$ can be separated into the sample mean and an extra bounded term of order at most $1/M_i$.



(a) Overlaid histograms

(b) 3D view

Figure 14: Histogram of perplexities of each individual data training sequence produced by different generations as is evaluated by the very first model trained with the real data. Over the generations models tend to produce samples that the original model (trained with real data) is more likely to produce. At the same time, a much longer tail appears for later generations – later generations start producing samples that would never be produced by the original model *i.e.* they start misperceiving reality based on errors introduced by their ancestors. Models here are explicitly forced to not repeat sequences with a penalty of 2.0.



(a) Figure 11.a in 3D. No data preserved.

(b) Figure 11.b in 3D. 10% original data preserved.

Figure 15: Histogram of perplexities of each individual data training sequence produced by different generations as is evaluated by the very first model trained with the real data. Over the generations models tend to produce samples that the original model (trained with real data) is more likely to produce. At the same time, a much longer tail appears for later generations – later generations start producing samples that would never be produced by the original model *i.e.* they start misperceiving reality based on errors introduced by their ancestors.