PRECISION-RECALL COVER: A METHOD FOR ASSESSING GENERATIVE MODELS

FASIL CHEEMA

A THESIS PROPOSAL
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO
AUGUST 2023

Abstract

Generative modelling has seen enormous practical advances over the past few years from LLMs like ChatGPT to image generation. However, evaluating the quality of a generative system is often still based on subjective human inspection. To overcome this, very recently, the research community has turned to exploring formal evaluation metrics and methods. In this work, we propose a novel evaluation method based on a two-way nearest neighbor test. We define a new measure of mutual coverage for two probability distributions. From this, we derive an empirical analogue and show analytically that it exhibits favorable theoretical properties while it is also straightforward to compute. We show that, while algorithmically simple, our derived method is also statistically sound. We complement our analysis with a systematic experimental evaluation and comparison to other recently proposed measures. Using a wide array of experiments, we demonstrate our algorithm's strengths over other existing methods and confirm our results from the theoretical analysis.

Acknowledgements

I would like to express my heartfelt gratitude to all those who have contributed to the completion of this thesis. Their support, guidance, and encouragement have been instrumental in making this journey a rewarding one. First and foremost, I am deeply thankful to my supervisor, Dr Ruth Urner, for her invaluable mentorship and insightful feedback throughout the research process. Dr Urner has been supportive and I feel that my growth as a researcher could not have been possible without her. Her expertise in the field of machine learning has been a guiding light, and I am truly grateful for the opportunity to learn under her guidance. I could not have asked for a better supervisor. I extend my appreciation to York University for providing a conducive environment for machine learning research. The university has been pivotal in shaping the trajectory of this work and fostering a positive, supportive environment for scientific research. My friends have been a constant source of motivation and positivity. Their social and emotional support has kept me going throughout this project. Your camaraderie has truly been a blessing and

critical to the success of this project. To my family, I owe a debt of gratitude that words can hardly capture. Your unwavering support, patience, and belief in my capabilities have been my foundation. Your encouragement has carried me from my first day of school to this point in my education. In closing, I want to express my heartfelt thanks to everyone who has played a role in this endeavor. Each contribution, no matter how small, has had a significant impact on the final outcome. This thesis stands as a testament to the power of collaboration, support, and the pursuit of knowledge. Thank you.

Table of Contents

List of Figures

1 Introduction and Motivation

Generative modelling is a field of machine learning that has garnered significant attention since the introduction of generative adversarial networks, diffusion models, and Large Language Models (LLMS) such as ChatGPT [4, 5, 9, 12, 13, 21, 26, 30]. Despite the considerable progress that has been achieved, the assessment of a generative system's quality frequently relies on subjective and time-consuming human evaluation, as opposed to utilizing formal quantitative metrics. In order to better aid in the development of better generative models, recent work has been done in quantitatively grading models' performance with the creation of novel evaluation metrics [2, 8, 19, 20, 23]. A noteworthy and innovative approach to addressing this issue involves the concept of gauging the coverage of one distribution in comparison to another. This comparison draws parallels to the statistical concepts of precision and recall [23]. The main idea is to evaluate the degree to which a generative model produces instances that are of high quality. This is akin to ensuring if the generative distribution is covered by the true distribution. Also, the degree to which the

generative model manages to reflect the full diversity of the true distribution that it is aiming to mimic. This can be seen as gauging if the true distribution is covered by the generative distribution.

These methods have a very appealing property: they use general algorithmic tools that do not take any specific parametric form. Due to this property, they are not restricted to specific generative distributions, nor do they require direct access to the probability distribution. Generative models such as kernel density estimators (KDEs) [22] or normalizing flows [17, 27] construct explicit probability distributions where we can readily employ a divergence measure or some other notion of statistical distance [8, 28]. However, other types of generative models, such as generative adversarial networks (GANs) [9] and variational auto encoders (VAEs) [16] do not construct explicit generative probability distributions but implicitly construct a probability distribution via generative samples (see fig 1.1). Not having a parametric definition of the distribution proves to make scoring such models in an objective manner more challenging. The recent line of research on precision-recall type of metrics has thus aimed at developing methods to grade generative models based solely on the samples created [19, 20, 23].

The problem is then to find a way to assess two probability distributions in a quantitative way using the samples only. While the originally proposed notion of precision and recall for generative models [23] was based on an intriguing formal

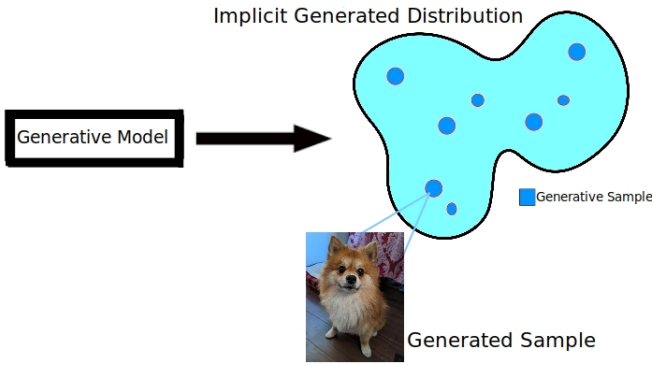

Figure 1.1: In this illustration we see that a generative model creates an implicit generated distribution. The user does not have access to this distribution but rather samples from this implicit distribution. Samples such as the image of this small dog are generally the output of such generative models such as GANs.

foundation, it lacks clear theoretical analysis as well as a lack of a formal algorithm to determine the arbitrary choices affecting the outcome [8, 23]. Follow up studies have proposed capturing the idea of measuring precision and recall through nearest neighbor type tests [19, 20]. However, these methods also lacked a concise theoretical framework to provide theoretical guarantees and properties of their measures.

In our work we introduce a nearest neighbor based method to evaluate precision and recall measures (in its empirical version), which we term precision recall cover. We also introduce a natural population level notion that captures the degree to which relevant areas of one distribution receive sufficient coverage by the

other distribution. "Relevant" and "sufficient" are quantified by two parameters β and α that allow for fine-tuning the coverage requirements. This population level notion leads way to an empirical version which is a k, k'-nearest neighbor test, where sufficient coverage is defined by a k'-nearest neighbor ball from a distribution containing at least k sample points from the other distribution. We provide some theoretical analysis of both the population level and empirical version of the precision recall cover measure and hence argue for its statistical soundness. Additionally we provide some simple and interpretable experiments, comparing our measure's performance as well as failure and success cases with the previous precision recall type measures. Finally, we provide a proof of concept of how our measure would assess the behavior of a generative adversarial network (GAN).

The thesis has been organized in a manner reflective of the chronology of the work done. In Chapter 2 we start with a literature review of existing work done in assessing generative models. We focus mainly on the most relevant precision-recall line of work, where we extensively delve into each method thoroughly examining their merits and limitations. Naturally, this leads us into the next chapter which showcases our work. In Chapter 3 we present preliminary elements, including assumptions that guide our theoretical analysis. We also outline our method and theoretical analysis behind our work. In Chapter 4 we construct a set of experiments to see different measures' performances on simple sets of distributions. We

also present experiments that check how fast uniform distributions for both generative and real distributions convergence to the common 'true' values of precision and recall for comparable measures. Finally, in this chapter we quickly show a litmus test by demonstrating our measure's score on actual state of the art (SOTA) GANs. We also construct several experiments that showcase desired behaviours of an ideal measure in real life data and compare several measures. This work has been published and presented at AISTATS 2023.

2 Literature Review

Due to the accelerating developments in generative modelling in recent years, the task of developing a way to quantitatively assess generative models has received immense attention. As such, there is now a significant amount of work dedicated to quantitatively assessing generative models [2, 19, 20, 23]. For the sake of brevity, we will focus our discussion here on evaluation measures most relevant to ours. Some of the initial methods to assess GANs were the Frechet Inception Distance (FID) [11] and the Inception Score (IS) [24]. FID still continues to be one of the most popular metrics for generative models. FID models both samples from the true and generated distributions as having been generated by multivariate normal distributions. FID then fits the samples to respective normal distributions and measures the distance of these Gaussians based on their predicted parameters. In practice, the FID score is computed on embeddings usually obtained from an Inception V3 network run on high-dimensional data (such as images), then the score is computed on the two sets of real and generated embeddings. The quality of the metric's score

is then dependent on how appropriate the assumption of the distributions as Gaussians was. For generative models aimed at producing a variety of outputs from a set of diverse and exotic distributions not constrained to multi-variate normal distributions, such modelling might be sub-optimal.

The IS score also takes the Inception network to embed into an embedding space and then computes the KL-Divergence between the conditional probability distributions of labels and the marginal distributions of the labels over the generated samples. The authors of [2] note a lot of work still needs to be done as IS does not capture intra-class diversity well, and requires many samples to be reliable. Both these methods are still widely used, but the community acknowledges the need for improvement [2, 19, 20, 23].

As discussed prior, an emerging research direction has aimed at assessing generative models through methods analogous to the classical notions of precision and recall. We will now delve into the first three of this type of work [23, 19, 20].

2.1 Assessing Generative Models via Precision and Recall [23]

The initial step in this research direction introduced a metric aimed at measuring how much each sample covered the support of the other [23]. The paper establishes an elegant framework of measuring mutual coverage by representing the generated and real distributions as mixtures of joint and exclusive components. The mixture coefficients (the portion of each distribution that is in the joint and exclusive regions) then yields what the authors call a "precision recall curve". Although the fundamental idea behind the paper is elegant, there is no explicit algorithm tied to the framework. We will refer to the Precision Recall measure as PR for short.

In this thesis the major idea is to develop a framework for quantitatively assessing how accurate generative models are in comparison to a 'true' distribution. In this setup there is a true underlying data generating distribution called P, and a generative distribution Q. The authors start off by introducing a new notion of precision and recall extended to distributions. The authors intuitively describe the new framework of PR as precision being the 'part of P' that can generate Q and recall as being the 'part of Q' that can generate P. These are the mixture components referenced prior. Intuitively precision is the fidelity of the generated samples

as the better Q is the bigger part of P can be used to make Q. On the other hand recall is the diversity of Q since the better recall is, the bigger part of Q can be used to encompass P. The authors then introduce a formula that reflects this intuitive definition:

$$P = \bar{\beta}P_S + (1 - \bar{\beta})P_{\bar{S}} \quad \text{and} \quad Q = \bar{\alpha}Q_S + (1 - \bar{\alpha})Q_{\bar{S}} \tag{2.1}$$

where S is the common support of both P and Q. That is $\text{supp}(P) = \{\omega | P(\omega) > 0\}$ and S is $S = \text{supp}(P) \cap \text{supp}(Q)$ where $P(\omega)$ is the probability mass of event ω. Precision is typically denoted as α and recall as β. Here P_S is the distribution that has both joint support of distributions S and P, and $P_{\bar{S}}$ is the part of distribution P that cannot be generated by distribution Q. They then adjust the distributions P_S and Q_S by parameterizing them with a common distribution μ:

$$P_S = \beta'\mu + (1 - \beta')P_\mu \quad \text{and} \quad Q_S = \alpha'\mu + (1 - \alpha')Q_\mu \tag{2.2}$$

The distribution μ is a distribution on the common part of both P and Q. Whereas the distribution P_μ is the part of μ missed by Q so it is a loss of recall. Whereas the distribution Q_μ is the part of μ missed by P so it is a loss of precision.

The authors combine the 2 equations by plugging (2.2) into (2.1) and get:

$$P = \beta\mu + (1 - \beta)\nu_P \quad \text{and} \quad Q = \alpha\mu + (1 - \alpha)\nu_Q \tag{2.3}$$

note that ν_P contains both $P_{\bar{S}}$ and P_μ that is the part of P that misses Q and also the noise. Then they define a definition of **PRD**(Q,P) which is all sets of α and β that suffice equation (2.3) for some choice of μ, ν_P and ν_Q also including the pair (0,0).

Now on to the major result of the paper; the algorithm. Now it is trivial if we can always find the pairs α and β that suffice equation (2.3). However, this may not be possible as we will need information on the distributions ν_P, ν_Q, and μ first in order to actually obtain the α and β pairs. Therefore the authors prove the following theorem and show that for our choice of several parameters we can get the PRD curves without requiring information on the distributions as stated before:

Theorem 2.[23] *Let P and Q be two probability distributions defined on a finite state space Ω. For $\lambda > 0$ define the functions*

$$\alpha(\lambda) = \sum_{\omega \in \Omega} \min(\lambda P(\omega), Q(\omega)) \quad \textit{and} \quad \beta(\lambda) = \sum_{\omega \in \Omega} \min(P(\omega), \frac{Q(\omega)}{\lambda}) \tag{2.4}$$

Then it holds that

$$\mathrm{PRD}(Q, P) = \{(\theta\alpha(\lambda)), \theta\beta(\lambda)) | \lambda \in (0, \infty), \theta \in [0, 1]\}.$$

So this theorem leads to their algorithm, it can be seen in figure 2.1 and in their paper as well [23]. Essentially, they start at the origin and map out lines of the

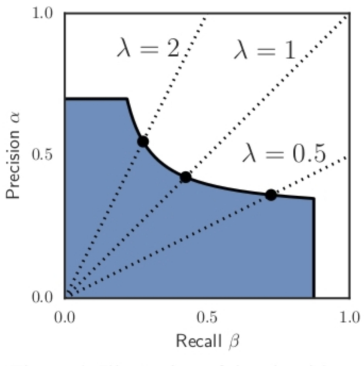

Figure 2.1: An illustration of the PR algorithm, with a construction of PR curves. Image is figure 4 from [23].

form $y = \lambda x$ where we start at the origin and end at the maximum achievable point (α, β). To find the maximum achievable points we simply use the functions defined in (2.4) via Theorem 2 of [23]. So the only question left is what choice of λ? This is picked out as the author as states as an 'angular resolution' of $m \in N$ [23]. This can be seen as $\tan(\frac{i}{m+1}\frac{\pi}{2})$ where $i = 1, 2...m$. The idea behind this is to trace out the PR curves by using trigonometry. We use tan to determine the steepness of the slope as it is the ratio of height over width, and they quantify it by choosing to iterate i over a given specification m to take the tan function from 0 all the way to 1 (counterclockwise). Therefore the algorithm is simply computing $\alpha(\lambda)$ and $\beta(\lambda)$ in (2.4) over our defined λ:

$$\widehat{\mathrm{PRD}}(Q, P) = \{(\alpha(\lambda), \beta(\lambda)) | \lambda \in \Lambda\} \text{where} \quad \Lambda = \{\tan(\frac{i}{m+1}\frac{\pi}{2}) | i = 1, 2, ..., m\}.$$

Finally the authors also note that for choice of $\lambda = 1$ they obtain the total variation distance. This is simply the supremum of the difference between any point in P and Q. Simply put the largest possible difference between the probabilities that the two probability distributions can assign to the same event.

The paper finally discusses algorithmic implementation of their paper. In the algorithm the state space Ω discussed in Theorem 2 of [23] is approximated using k-means clustering [10]. The authors state they use a choice of $k = 20$ for their implementation. The clusters are obtained by applying the k-means algorithm over the union of both samples from $\hat{P}$ and $\hat{Q}$. Note that for high dimensional data such as images, it is standard to reduce the data to a lower dimensional feature space. This is done via an inception-v3 network run over image data to extract features [23]. After obtaining the features and obtaining the k-means cluster they then compute the PR (α, β) pairs via equation (2.4) over the clusters that approximate the state space Ω. Note that an issue the authors discuss is using the k-means clustering algorithm to define the state space. This algorithm is sensitive to initialization, and can construct a different set of clusters depending on initial conditions. The authors then recommend running the algorithm multiple times in order to avoid issues due to the initialization [23].

This paper has a nice high level approach to the problem of assessing generative models. However, algorithmically the implementation faces some challenges. Running k-means multiple times to avoid randomization may not be the most efficient method to obtain the state space Ω. Future methods attempt to remedy this by using other algorithms to construct the state space Ω. Also, although the PR curves are quite informative and non-parametric (only relying on the data) they do suffer from not providing a concrete score on the model's performance. Additionally, there is no analysis of consistency for the k-means based heuristic with respect to the originally introduced measure.

2.2 Improved Precision & Recall [19]

Precision-Recall garnered a lot of attention towards metrics for generative modelling. Follow up studies have aimed at providing a remedy by basing the precision recall evaluation on nearest-neighbor based computations. A metric called Improved Precision and Recall (IPR) [19] aimed to maintain the initial motivations of the PR Curves paper by seeking out regions where one distribution covered another and vice versa. A nearest neighbor based measure is naturally based on samples from both the true distribution P, and the generated distribution Q. However, this measure does not come with a population level analogue, and thereby not allowing for any analysis of statistical consistency.

The authors take the original paper on Precision-Recall [23], and note several issues with it and attempt to remedy them. The first most notable issue is that the authors point out, via experiments in a controlled setting with GANs, the PR curves incorrectly identify samples with very high precision as the ones with the worst precision, and similar issue for recall. The authors point out the key thesis and motivation for developing the Precision and Recall metric in the prior work: "...two separate goals emerge: individual samples drawn from the model should be faithful to the examples (they should be of "high quality"), and their variation should match that observed in the training set."[19] The authors show the following image figure 2.2 (which is figure 4 in [19]) demonstrates that the original PR curve framework does not match up with visual expectations. This is an issue because if a metric cannot detect such a blatant case of lack of quality/diversity in images then it has some serious issues. The authors take the direction of using two different metrics, one for precision (precision) and one for recall (diversity) however rectify some of the issues of the original. They note first that the usage of k-means clustering is problematic as it is prone to sensitivity with randomized initialization. To fix this they suggest using k-nearest neighbors to construct the 'divisions' and then assess precision and recall, as the fraction of generated samples within the real manifolds and the fraction of the real samples in the generated manifolds respectively.

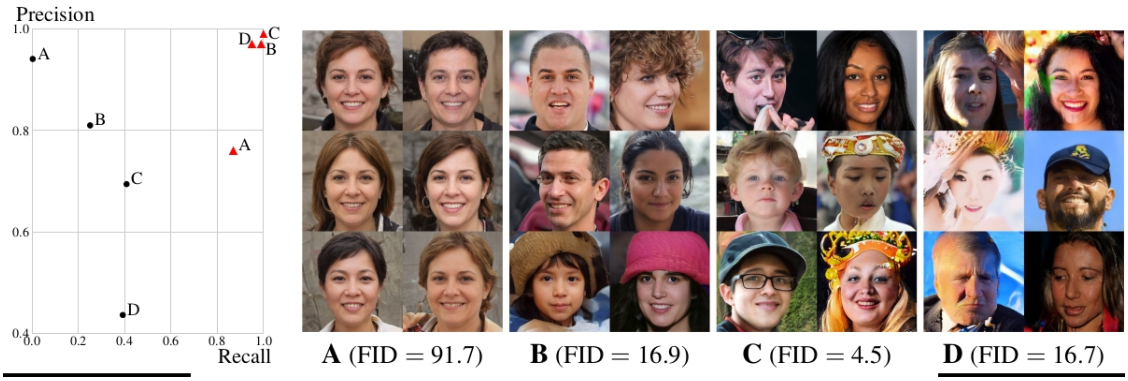

Figure 2.2: The lower the FID the higher quality a generated sample should be. As we can see the authors compare the original PR method's scores (red triangles) to their method (black dots). They synthetically control for the quality and diversity of the generated samples of a GAN using the 'truncation trick,' by truncating the normal vector in the generator. They show that their method does a better job in this setting aligning with expected behaviour. We can see as the quality increases and diversity decreases (furthest left) their score exhibits high precision but low recall. Also as diversity increases and quality decreases (furthest right) their score exhibits high recall but low precision. They also show FID scores do not always line up with what we subjectively expect as seen in the images on the right (the worst quality images) have the best FID scores. Note that the rightmost images have clear artifacts, clearly in contrast to the leftmost images. Image is figure 4 from [19].

2.2.1 Algorithm in practice

The first step as with all evaluative measures for generative models if dealing with high dimensional data is to bring them to a lower dimensional feature space. Then after we have them in this space we now start construction of the manifolds. This is done in the form of hyperspheres centered around the samples (real and generated), and with radius equal to the k-nearest neighbor in this paper they state they use a choice of k=3. They say that k=3 is a robust choice that avoids saturating the values most of the time, and FID usually quotes 50,000 samples so they choose to use $|\Phi| = 50,000$ [19]. So after they construct the hyperspheres that represent the manifold, this is analogous to a probability density functions (except does not necessarily sum to 1), they can now compute values of precision and recall. Precision is defined as the number of generated samples within what they call the real manifolds (how much of generated distribution can be made by a part of the real dist):

$$precision(\Phi_r, \Phi_g) = \frac{1}{|\Phi_g|} \sum_{\phi_g \in \Phi_g} f(\phi_g, \Phi_r)$$

and then recall is defined as the portion of samples from the true distribution within the generated manifolds (how much of the real distribution used to construct the generated distribution):

$$recall(\Phi_r, \Phi_g) = \frac{1}{|\Phi_r|} \sum_{\phi_r \in \Phi_r} f(\phi_r, \Phi_g)$$

where f is defined as follows:

$$f(\phi, \mathbf{\Phi}) = \begin{cases} 1, \text{ if } ||\phi - \phi'||_2 \leq ||\phi' - \text{NN}_k(\phi', \mathbf{\Phi})||_2 \text{ for least one } \phi' \in \mathbf{\Phi} \\ \\ 0, \text{ otherwise,} \end{cases}$$

Another way to restate the above binary function, is that if there exists a sample from the set $\mathbf{\Phi}$ and our ϕ sample of choice is within the hypersphere (defined by the NN) we assign a value of 1 and 0 otherwise.

Recall that GANs are composed of a generator and discriminator network. The generator comes up with inputs from a randomized seeded predefined latent space (usually a multivariate normal distribution) [9]. Now the authors [19] use a relatively recent trick called the "truncation trick." This method uses a Gaussian distribution during training and a truncated Gaussian during inference, this is the truncation trick. In the words of the authors that created the 'trick': "We call this the Truncation Trick: truncating a z vector by resampling the values with magnitude above a chosen threshold leads to improvement in individual sample quality at the cost of reduction in overall sample variety."[4] This method provides a trade-off between quality and diversity of the generated samples; the more truncation the better image quality and less variety. Again another quote from the paper cementing this: "This technique allows fine-grained, post-hoc selection of the trade-off between sample quality and variety for a given G."[4] This provides a natural and

contained way to see if metrics used for assessing GANs' behavior will line up with what we expect to see.

Adjusting the truncation, we expect that as truncation ψ increases (varies from 1 to 0.3 where $\psi = 1$ has the least truncation) we have more recall (greater diversity) and less precision (lower quality) [19]. Their results show the decline in precision and increase in recall as truncation is increased (or going to a lower value of ψ such as the lowest which is 0.3). Also they map out Pareto frontiers for the multi-objective frontiers of improved precision and recall, and using these frontiers identify the best trained models of StyleGAN [19].

2.2.2 Summary

The authors highlight serious issues with prior work and make improvements to said issues of PR; mainly rectifying the issue of k-means random initialization. They also manage to artificially create GANs that are more or less inclined to create samples of higher precision or recall and therefore see if their metric is aligned with the GANs' samples. Although, the metric does do well with issues that faced prior work, it still faces some challenges. Most notably is using the k-nn in the way they

did to construct the manifold representing real and generated distributions. This type of setup is prone to over-covering regions of support for both distributions (see our later analysis) and some of the choices (such as the choice of k) seem arbitrarily based [20].

2.3 Reliable Fidelity and Diversity Metrics for Generative Models [20]

We now move on to one of the more recent works in metrics for assessing generative models [20]. The authors first start off by indicating severe issues that face several popular metrics that evaluate GANs and other generative models. Then they recognize the steps made by the recent paper [19], and note issues it faces and how to remedy them. Most notably the authors point out that when two distributions match, IPR fails to identify that the two distributions are the same. This case would be a very basic and serious flaw in the methodology. The work also seeks to remedy the issues the other methods face in regards to outliers. The authors showcase their improved methodology that builds off the intuition of IPR via experiments and some analytical results that show the metrics exhibit some nice behaviours.

2.3.1 High-level overview of metrics

The authors discuss what many papers on metrics in generative modelling are currently doing. Most papers seek to develop a way to judge the quality of a sample generated. For generative modelling many domains employ high dimensional data that cannot be meaningfully compared in the original space they reside within [20]. To address this, almost all the major papers that seek to come up with ways to evaluate GANs (and other generative models, but GANs in particular) use some method to reduce the dimensionality, usually via a pretrained network trained on ImageNet [20]. The authors break down the pipeline that has emerged in this niche of research and explain the importance of each component. We start off with the first component, the embeddings: this is usually in the form of an inception network or some model trained to bring the high dimensional data into some lower dimensional latent space. Then, next we have the comparison step between samples from the generated and real distributions. That is, we have to essentially conduct statistical analysis to compare if samples from the generated and real distributions are the same, and if not, how far apart they are (usually in the form of a distance measure between distributions). This involves first building these surrogate distributions in the latent space and then comparing them with a meaningful distance metric [20]. Some of these metrics have involved singular

quantities to summarize the difference between two distributions, while another increasingly promising trend is to have a dual-quantity metric that assesses both the quality of the samples (how good are the samples) and the diversity of the samples (how much variety is showcased within the samples) [19, 20].

2.3.2 Density & Coverage

The authors note a need to improve the improved precision and recall (IPR) method. IPR is explained in detail in a prior section however, to reiterate, it essentially constructs two manifolds (akin to pdfs) from the samples that are made from hyperspheres that vary based off of the k-NN of each sample. The precision is the portion of generated samples lying within the real manifolds, and vice versa for recall. The major issue of IPR is the constructed manifold assumes a uniform distribution within the hyperspheres and is prone to overestimating precision for regions around outliers (see figure from [20] for a visual example). To remedy this Density and Coverage seeks to reward high density regions, and punish sparse outliers. They do this by essentially adjusting the original definitions of IPR:

$$\text{density} \ := \frac{1}{kM} \sum_{j=1}^{M} \sum_{i=1}^{N} \mathbf{1}_{Y_j \in B(X_i, NND_k(X_i))}$$

and for coverage:

$$\text{coverage} \ := \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{\exists j \text{ s.t. } Y_j \in B(X_i, NND_k(X_i))}$$

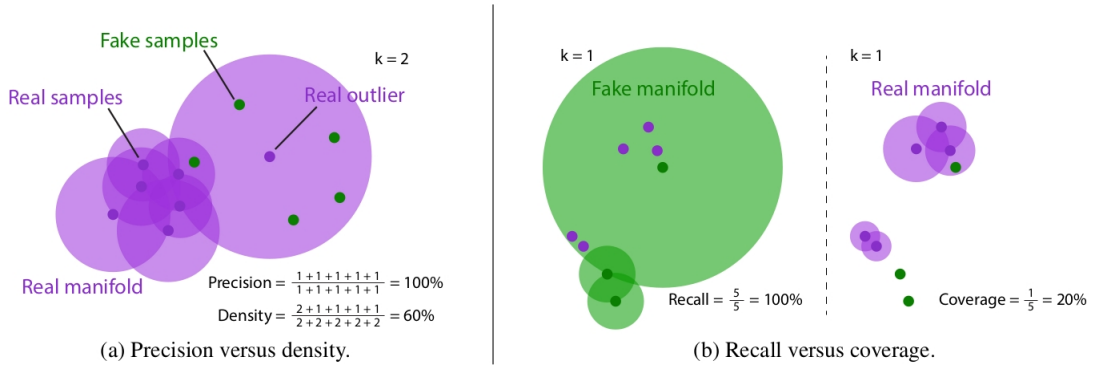

Figure 2.3: A visual aid: figure 1 from [20] showing cases where density and coverage don't make the trivial mistakes improved precision and recall. The authors refer to the generated distribution as the 'fake' distribution.

So precision (in IPR) measures how many generated samples are within any real point's k-nn ball's boundaries (real manifold). Density corrects this by counting how many generated samples are within how many real samples k-nn ball's boundaries. This puts weight on points that lie within the boundaries of multiple real samples k-nn balls, thus mitigating the issue prevalent in IPR. The authors point out that this is similar to a kernel density estimator to construct the pdf with a uniform kernel, and kNN being the window width, which is why there is a k-term in the denominator [20].

The coverage is also different than the recall term. Now instead of counting the number of real samples that lie within the boundaries of a generated sample k-nn

ball. The coverage counts the number of real k-nn balls that contain a generated sample, as opposed to the number of real samples within a generated sample's k-nn ball. The authors also note that although k-NN maybe a computationally difficult task. The computation only needs to be done once per dataset as opposed to repeatedly per model. This is because practitioners do not need to construct manifolds for the generated samples, only the real ones [20]. This greatly alleviates computational complexity for these tasks.

Now the authors state that for the IPR framework there does not exist closed form analytical solutions for the expected value of precision and recall [20]. In contrast, the authors state that for identical generated and real distributions the expected value of density and coverage approaches a value of 1, which is desired behaviour. The proof comes from the expectation of the density; this translates to a probability of the event where the generated sample is the kth smallest among the real values. The proof idea is simple, each ranking (of distances between the N real points and the generated sample) has probability $\frac{1}{N!}$ (which they state are equally likely) and there are $(N-1)!$ possible rankings with the generated sample in the kth position, multiplied by k for k possible rankings of distances in the k-nn computation. Therefore the authors state that the probability of a generated point (sampled from the same distribution as the real points) that is in the knn ball of a

real point is:

$$\Pr(y \in B_N^k) = k \cdot \frac{(N-1)!}{N!} = \frac{k}{N}$$

Therefore the expectation of the density when the real and generated distribution is the same is:

$$\mathbb{E}[density] =$$

$$= \frac{1}{k} \sum_{i=1}^{N} \Pr(y \in B_N^k)$$

$$= \frac{1}{k} \sum_{i=1}^{N} \frac{k}{N} = 1$$

They apply a similar argument for expectation of coverage.

2.3.3 Experiments

Finally, we move to the experiments of this paper. The paper shows that for a toy model of gaussians in 64-dimensional space for a generated Gaussian that matches the real distribution whereas IPR does not perform well (precision = 0.68 and recall = 0.67) [20], whereas the new metric does pretty well with density=1.06 and coverage =0.97. Also for the varying truncation (see truncation trick in the prior section of IPR) ψ density and coverage exhibit 'better' behaviour than IPR, that is as ψ increases more diversity so coverage increases linearly and density decreases

linearly. Similar behaviour is exhibited in the VAE setup by truncating the latent variable, and better diagnosis of fidelity-diversity trade-off is seen. The authors finally make a point to talk about future work. In particular, they make a point that the future for work in making metrics to evaluate generative models requires getting better embeddings of high dimensional data. As we discussed, almost all generative metrics first take high dimensional data and embed them into a lower dimensional latent space. Then, in this embedding we run the metrics themselves. Thus if we have accurate embeddings, we can get a better idea of the fidelity and diversity of the samples.

2.3.4 Summary/Critique

Although the paper addresses concerns from prior work, and seemingly does do a good job of correcting issues from previous work. There are several issues with the methodology. Firstly, the metric has values not restricted between 0 and 1. In the case where density is above 1, the authors do not indicate what this behavior entails. A key property of any measure for generative models should be how interpretable it is, otherwise diagnosing issues becomes a lot more difficult. Also, it is key to note that both the density and coverage score are not symmetric. That is, density provides a summary of the fidelity of the samples, and coverage provides a summary of the diversity. But if we were to swap the two distributions; call the generated

distribution the real distribution and vice versa, now the density does not become coverage nor vice versa. A more serious issue is in the proofs of the analytic results in the paper. The authors state that they show the expectation of the density is 1 when the two distributions are identical. There is a serious issue with this proof. The authors construct a set of random variables; the set of absolute values of differences between the i.i.d. random variables of two points sampled from the real distribution:

$$S = \{|X_1 - X_2|, |X_1 - X_3|, ..., |X_1 - X_N|\}$$

Then they create a new random variable; the absolute value of the difference between a point from the generated distribution and a fixed point from the real distribution (they use X_1); $|Y - X_1|$. They claim that the set of random variables $S \cup |Y - X_1|$ is i.i.d., however this is clearly false as all these values are dependent on the choice of X_1, and therefore they are not independent of each other. This assumption is thus invalid, and the rest of the proof depends on this fact. Although the method does seem to have a few issues, it seems to be a step in the right direction with regards to prior work. From the experiments, DC seems to be less accepting of outliers than IPR.

3 PR Cover Algorithm

3.1 Precision Recall Cover

In this section of this chapter we introduce some preliminaries and notation. Then we discuss motivation and consequently defined our precision recall cover measure for two distributions on the population level. In the next subsection, we then present an empirical analogue to our measure.

3.1.1 Preliminaries

We start by listing some technical lemmas and tools that we use in our analysis. The following lemma is taken from [18].

Lemma 1. *Let B denote the class of balls on $\mathcal{X}$, with VC-dimension $\mathcal{V}_B$. Let $0 < \delta < 1$, and define $\alpha_n = (\mathcal{V}_B \ln 2n + \ln(\frac{8}{\delta}))/n$. The following holds with probability at least $1 - \delta$ for all balls in B. Pick any $a \geq \alpha_n$. Then $\mu(B) \geq 3a \implies \mu_n(B) \geq a$ and $\mu_n(B) \geq 3a \implies \mu(B) \geq a$.*

Note that, for d-dimensional euclidian spaces the VC-dimension of the class of balls is $d + 1$.

The following are commonly used inequalities. They can be found in the appendix of [25].

Lemma 2 (Lemmas A.1. and A.2. in [25])**.**

- *Let* $a \geq 0$. *Then* $x \geq 2a \ln(a)$ *implies* $x \geq a \ln(x)$.
- *Let* $a \geq 1$ *and* $b > 0$. *Then* $x \geq 4a \ln(2a) + 2b$ *implies* $x \geq a \ln(x) + b$.

We use the above to show the following inequality:

Lemma 3. *Let* $a \geq 0$ *and for* $x > 8$. *Then* $x \geq 4a \ln(a)$ *implies* $x \geq a \ln(2x)$.

Proof. Using Lemma 2, the ineqality $x \geq 4a \ln(a)$ implies $x/2 \geq a \ln(x/2)$, which implies

$$x \geq a \ln(x^2/4).$$

Now note that $x > 8$, implies $x^2 > 8x$, and thus $x^2/4 > 2x$. Combining this with the above inequality, we get

$$x \geq a \ln(2x)$$

as claimed. □

3.1.2 Setup and definitions

We consider a space $X \subseteq \mathbb{R}^d$ and two distributions P and Q over X. We will think of these as the true underlying data generating distribution P, and the generative distribution Q that is induced by some learned generative model. We use the notation $x \sim P$ to mean a point x sampled from distribution P. We use the notation $\hat{P} = (x_1, x_2, \ldots, x_n)$ and $\hat{Q} = (y_1, y_2, \ldots, y_m)$ for samples from P and Q respectively. The empirical distributions induced by the samples (uniform distributions over the sample points) are then denoted by P_n and Q_m. For simplicity of presentation, we will assume that both P and Q admit continuous density functions $d_P : X \to \mathbb{R}$ and $d_Q : X \to \mathbb{R}$ respectively. We will use the notation $\text{supp}(P)$ and $\text{supp}(Q)$ to denote the supports of the distributions P and Q respectively (regions with nonzero probability mass w.r.t. the distributions).

We will also use notation such as $Q(A)$ to denote the probability of event A occurring with respect to probability distribution Q. For our proposed evaluation measure, the events of importance will be balls in $\mathbb{R}^d$. We will let $B_P(x, \beta)$ denote the smallest ball of probability mass at least β with respect to probability distribution P centered at point x. Since we assume that our distributions have a continuous density function, there always exists balls $B_P(x, \beta)$ of mass exactly β. For the empirical distributions, $B_{\hat{P}}(x, \frac{k}{n})$ will denote a k-nearest neighbor ball with

respect to sample set $\hat{P}$ around domain point x.

Several recent studies, as discussed in Chapter 2 attempt to highlight the fidelity and diversity of generated samples [19, 20, 23]. This typically entails measuring how much each distribution covers the support of the other. Usually, *precision* is defined as the portion of the support of Q that is in the support of P as well. Conversely, *recall* is defined as the portion of the support of P that is in the support of Q as well. Our notions of *precision cover* and *recall cover* aim to capture the intuition that we should only care about a region being covered (by the other distribution) if it has at least certain probability mass. In the context of a learned generative model, if an area has a negligible probability in terms of the true distribution, then we may not care about the generative model not being able to generate instances in that area. Conversely, if the generative model has a negligible probability of generating a certain type of instances, then we may not be too concerned about how likely such instances are generated by the true process.

These considerations are accounted for by the two parameters α and β in the definition (below) of our precision-recall cover measure. The parameters α and β provide "knobs" on the cut-off for an area to be considered "negligible" and correspondingly for coverage to be considered "sufficient".

Definition 1. *Let P and Q be two probability distributions over some space X, and let $\alpha, \beta \in [0, 1]$ with $\alpha \leq \beta$ be given. Then we define the* (α, β)-precision coverage

of P by Q *by*

$$\mathrm{PC}_{\alpha,\beta}(P,Q) = \mathop{\mathbb{P}}_{y\sim Q}[P(B_Q(y,\beta))) \geq \alpha] \tag{3.1}$$

Conversely, we define the (α,β)-recall coverage of P by Q *by*

$$\mathrm{RC}_{\alpha,\beta}(P,Q) = \mathop{\mathbb{P}}_{x\sim P}[Q(B_P(x,\beta))) \geq \alpha] \tag{3.2}$$

The precision version of this measure takes two distributions, and for each point y from Q considers a ball of probability mass β. The measure reflects what portion of these balls have probability mass at least α with respect to P. The choice of α and β allow for generality and for the user to fine tune the sensitivity of the evaluation measure. For instance, perhaps our distribution P is completely disjoint from Q however the points from each distribution are very close in $\mathbb{R}^d$, we may not want to punish this behaviour as badly as for points that are completely disjoint and very far from each other. Recall is the analogous measure with the role of the distributions swapped.

3.1.3 Population level properties of precision recall cover

To further motivate our proposed measures, we start by stating some simple properties of our proposed measure.

Observation 4. *Let* $X \subseteq \mathbb{R}^d$ *be some domain and* P *and* Q *be two distributions over* X*. Then:*

1. $\mathrm{PC}_{\alpha,\beta}(P,P) = \mathrm{RC}_{\alpha,\beta}(P,P) = 1$ *for all* $0 < \alpha \leq \beta \leq 1$

2. $\mathrm{PC}_{\alpha,\beta}(P,Q) = \mathrm{RC}_{\alpha,\beta}(Q,P)$

3. $0 \leq \mathrm{PC}_{\alpha,\beta}(P,Q) \leq 1$ *and* $0 \leq \mathrm{RC}_{\alpha,\beta}(P,Q) \leq 1$ *for all* $0 < \alpha \leq \beta \leq 1$

4. *If* $\mathrm{supp}(P) \cap \mathrm{supp}(Q) = \emptyset$*, then there exist* $\alpha \leq \beta$ *sufficiently small such that* $\mathrm{PC}_{\alpha,\beta}(P,Q) = 0$ *and* $\mathrm{RC}_{\alpha,\beta}(P,Q) = 0$

The last statement shows that, as the parameters α and β get smaller (the measure thus more refined) the precision recall cover will correctly identify that distributions have disjoint support. Moreover, as we will show next, it will correctly identify how much mass Q assigns to the support of P and vice versa. The first three statements follow from the definition 1. We now give proofs of the last statement below:

Proof of Observation 4. The first three claims follow directly from the definitions. For the last item, we will argue why it is true for the recall cover version. The claims for the precision cover then follow analogously. Let $(\beta_i)_{i\in\mathbb{N}}$ be a monotonically decreasing sequence of values for β in $(0,1)$ converging to 0. Note that for any point $x \in \mathrm{supp}(P)$ we have

$$\bigcap_{i=1}^{\infty} B_P(x, \beta_i) = \{x\}$$

and thus we get

$$\bigcap_{i=1}^{\infty} \bigcup_{x \in \mathrm{supp}(P)} B_P(x, \beta_i) = \overline{\mathrm{supp}(P)},$$

where $\overline{\mathrm{supp}(P)}$ denotes the (topological) closure of $\mathrm{supp}(P)$. Since the Lebesgue measure of $\overline{\mathrm{supp}(P)} \setminus \mathrm{supp}(P)$ is 0 and we are assuming our distributions to admit density functions, we get that for any $\alpha_0 > 0$, since the support of P and of Q are disjoint, there exists an index $i \in \mathbb{N}$ such that

$$Q\Big(\bigcup_{x \in \mathrm{supp}(P)} B_P(x, \beta_i) \Big) < \alpha_0.$$

This implies that, for any $\beta < \beta_i$ and $\alpha < \min\{\alpha_0, \beta\}$, we have

$$\mathrm{RC}_{\alpha,\beta}(P, Q) = \mathop{\mathbb{P}}_{x \sim P}[Q(B_P(x, \beta))) \geq \alpha] = 0$$

and the statements follow. □

Theorem 5. *Let P and Q be distributions over a space X. For any $\epsilon > 0$ there exist (sufficiently small) values $0 \leq \alpha < \beta \leq 1$, such that*

$$|\mathrm{PC}_{\alpha,\beta}(P, Q) - Q(\mathrm{supp}(P))| \ \leq \ \epsilon$$

and

$$|\mathrm{RC}_{\alpha,\beta}(P, Q) - P(\mathrm{supp}(Q))| \ \leq \ \epsilon.$$

Proof of Theorem 5. As in the proof of Observation 4, one can choose α_0 and β_0 sufficiently small so that

$$Q(\bigcup_{x\in(\mathrm{supp}(P)\setminus\mathrm{supp}(Q))} B_P(x,\beta_0)) < \alpha_0$$

and thus, for any $\beta \leq \beta_0$ and $\alpha \leq \alpha_0$ only balls around points in the support of Q will be α-covered by Q. However note, that not all of these will necessarily be α-covered by Q. It remains to show that we can choose α sufficiently small so that the P-mass of balls that are not α-covered by Q is at most ϵ.

Let some $\epsilon > 0$ be given and let's fix some $\beta_1 < \beta_0$. For every $x \in \mathrm{supp}(Q)$ we have $Q(B_P(x,\beta_0))) > 0$. We consider a sequence of values $(\alpha_i)_{i\in\mathbb{N}}$ that converges to 0. Then we get that the sets

$$A_i := \{x \in \mathrm{supp}(Q) \mid Q(B_P(x,\beta_1)) > \alpha_i\}$$

are ordered by inclusion, namely $A_i \subseteq A_j$ for $j \geq i$ and

$$\bigcup_{i=1}^{\infty}\{x \in \mathrm{supp}(Q) \mid Q(B_P(x,\beta_1)) > \alpha_i\} = \mathrm{supp}(Q).$$

Thus, there exists a $j \in \mathbb{N}$ such that for any $i \geq j$, we get

$$P(A_i) = P(\{x \in \mathrm{supp}(Q) \mid Q(B_P(x,\beta_1)) > \alpha_i\}) \geq P(\mathrm{supp}(Q)) - \epsilon.$$

This completes the proof. □

3.1.4 PR Cover algorithm

In practice, we generally do not have access to the actual distributions P and Q, but rather have sets of samples $\hat{P} \sim P^n$ and $\hat{Q} \sim Q^m$ from them. We now define an empirical analogue of the PR-cover measure.

Given n samples $\hat{P} = (x_1, \dots x_n)$ and m samples $\hat{Q} = (y_1, \dots, y_m)$, we can apply our measure over the discrete distributions that are induced by sample sets. We will use $\alpha = \frac{k}{m}$ and $\beta = \frac{k'}{n}$ where $k' = Ck$ for some $C \in \mathbb{N}$. That is, the measure will be based on k'-nearest neighbor balls over the sample sets.

Definition 2. *Let $\hat{P} = (x_1, \dots x_n)$ and $\hat{Q} = (y_1, \dots, y_m)$ be two sample sets, and let $k, k' \in \mathbb{N}$ with $k \leq k'$ be given. Then we define the* (k, k')-precision coverage of $\hat{P}$ by $\hat{Q}$ *by*

$$\mathrm{PC}_{k,k'}(\hat{P}, \hat{Q}) = \frac{1}{m} \sum_{j=1}^{m} 1\left[P_n(B_{\hat{Q}}(y_j, \frac{k'}{m})) \geq \frac{k}{n} \right] \tag{3.3}$$

Conversely, we define the (α, β)-recall coverage of $\hat{P}$ by $\hat{Q}$ *by*

$$\mathrm{RC}_{k,k'}(\hat{P}, \hat{Q}) = \frac{1}{n} \sum_{i=1}^{n} 1\left[Q_m(B_{\hat{P}}(x_i, \frac{k'}{n})) \geq \frac{k}{m} \right] \tag{3.4}$$

In the next section, we will show analysis for our algorithm.

Algorithm 1 outlines how to compute (k, k')-cover measures over two samples. The algorithm takes two parameters k and k' and samples $\hat{P}$ and $\hat{Q}$. It constructs k'-nearest neighbor balls over the sample set $\hat{Q}$ and then computes the total number of k'-nearest neighbor balls that contain at least k points from $\hat{P}$, and divides this

number by the total number of sample points in $\hat{Q}$, this gives us our precision cover metric. The recall metric is symmetric so it is simply the same procedure with the distributions swapped. In the following pages we outline our procedure: the main Precision-Recall Cover algorithm and its subroutine. We only show the algorithm for the precision style measure as the recall algorithm is identical with the distributions swapped.

Algorithm 1 Precision-Recall Cover

Input Sample Sets $\hat{P}$, $\hat{Q}$ as well as choice of k, and C

$\hat{Q}_\beta \leftarrow \emptyset$

$k' \leftarrow C * k$

$\hat{r}_{\hat{Q}} \leftarrow$ k'-NearestNeighborDistances$(\hat{Q}, k')$

for $y \in \hat{Q}$ **do**

  val $\leftarrow$ PR-Cover-Indicator$(y, \hat{r}_y, \hat{P})$

  if $val = 1$ **then**

    $\hat{Q}_\beta \leftarrow \hat{Q}_\beta \cup y$

  end

end

PC $\leftarrow \frac{|\hat{Q}_\beta|}{|\hat{Q}|}$

return PC

Algorithm 2 Precision-Recall Cover Indicator subroutine

Input Sample Set $\hat{P}$, sample point $y \in \hat{Q}$, (k')-nearest neighbor distance $\hat{r}_y$, and k

$\mathsf{val} \leftarrow 0$

$i \leftarrow 0$

for $x \in \hat{P}$ **do**

  if $||y - x|| \leq \hat{r}_y$ **then**

    $i \leftarrow i + 1$

  end

end

if $i \geq k$ **then**

  $\mathsf{val} \leftarrow 1$

end

return val

3.2 Analysis

Now we outline some of the key formal performance guarantees that our algorithm and PRC measure enjoy. We will phrase all our results in terms of the precision cover measure. By symmetry, they will hold for recall cover when the arguments are switched. The results are obtained by applying techniques of VC-analysis to

k-nearest neighbour balls [1, 6, 18, 29].

3.2.1 Identical distributions

We start by presenting a type of sanity check analysis for the cases where the two distributions are identical. In that case, we have $\mathrm{PC}_{\alpha,\beta}(P, Q) = \mathrm{PC}_{\alpha,\beta}(P, P) = 1$ for all α, β, see Observation 4. We show that, given sufficiently large samples, the empirical k, k'-cover measure will also attain value 1 with high probability.

Theorem 6. *Let $\delta > 0$, $X \subseteq \mathbb{R}^d$ a domain and $\hat{P}$ and $\hat{Q}$ denote samples from distributions P and Q over X of sizes n and m respectively. Then there exist finite sample sizes N and M, depending only on the dimension of the space and the confidence parameter δ, and appropriate choices of k and k' such that the following holds: if $P = Q$, with probability at least $1 - 2\delta$, over the sampling of $\hat{P}$ of size $n \geq N$ and $\hat{Q}$ of size $m \geq M$, we have*

$$\mathrm{PC}_{k,k'}(\hat{P}, \hat{Q}) = 1$$

The proof is given below, it uses a lemma from [18] which is adapted from [3].

Proof of Theorem 6. We let $\mathcal{V}_B$ denote the VC-dimension of balls in the space X. Note that if X is a d-dimensional euclidean space, then $\mathcal{V}_B = d + 1$. Let some $\delta > 0$ be given. We here assume that the two distributions $P = Q$ are identical,

and consider two samples $\hat{P}$ and $\hat{Q}$, both of size

$$n \geq 36\mathcal{V}_B \ln(36\mathcal{V}_B) + 18\ln(8/\delta)$$

from this distribution. The above implies $2n \geq 72\mathcal{V}_B \ln(36\mathcal{V}_B) + 36\ln(8/\delta)$, and applying the second item of Lemma 2 to this inequality yields $2n \geq 18\mathcal{V}_B \ln(2n) + 18\ln(8/\delta)$ and thus

$$n \geq 9\mathcal{V}_B \ln(2n) + 9\ln(8/\delta) \tag{3.5}$$

We will assume that these samples are so that the implications in Lemma 1 are valid, and note that this holds with probability at least $1 - 2\delta$ over drawing $\hat{P} \sim P^n$ and $\hat{Q} \sim Q^n$. We set $k = \mathcal{V}_B \ln(2n) + \ln(8/\delta)$ and let $k' = 9k = 9(\mathcal{V}_B \ln(2n) + \ln(8/\delta))$, so that with Equation 3.5 we get $n \geq k'$, and thus k'-nearest neighbor balls are well defined for samples $\hat{P}$ and $\hat{Q}$ and $k'/n \leq 1$.

Now consider any point $y \in \hat{Q}$ and the k'-nearest neighbor ball $B_{\hat{Q}}(y, \frac{k'}{n})$ with respect to $\hat{Q}$ around this point. By definition of k' and this ball, we have

$$Q_n(B_{\hat{Q}}(y, \frac{k'}{n})) = \frac{k'}{n} = \frac{9(\mathcal{V}_B \ln(2n) + \ln(8/\delta)}{n}$$

Invoking Lemma 1, this yields

$$Q(B_{\hat{Q}}(y, \frac{k'}{n})) = P(B_{\hat{Q}}(y, \frac{k'}{n})) \geq \frac{k'}{3n} = \frac{3(\mathcal{V}_B \ln(2n) + \ln(8/\delta)}{n}$$

Invoking the other implication of the same lemma, this yields

$$P_n(B_{\hat{Q}}(y, \frac{k'}{n})) \geq \frac{k'}{9n} = \frac{(\mathcal{V}_B \ln(2n) + \ln(8/\delta)}{n} = \frac{k}{n}$$

Thus, with probability at least $1-2\delta$, every k' nearest neighbor ball in $\hat{Q}$ contains at least k points from $\hat{P}$ for the stated values of k and k' and identical sample sizes $n = m \geq 36\mathcal{V}_B \ln(36\mathcal{V}_B) + 18\ln(8/\delta)$. This immediately implies $\mathrm{RC}_{k,k'}(\hat{P},\hat{Q}) = 1$. The corresponding statement for $\mathrm{PC}_{k,k'}(\hat{P},\hat{Q})$ follows analogously. □

3.2.2 Local consistency

Of course, in general, we don't expect the two input distributions to be exactly identical. We now show that our k, k' mechanism will correctly identify local regions that are sufficiently covered by the other distribution: For a given local density ratio $\frac{P(B(x,k'))}{Q(B(x,k'))} > \omega$ there is a true distribution sample size such that all regions with sufficient density ratio will be recognized by our measure.

Theorem 7. *Let $\delta > 0$,$C > 1$, $X \subseteq \mathbb{R}^d$ a domain and $\hat{P}$ and $\hat{Q}$ be samples from distributions P and Q over X of sizes n and m respectively. Let $\mathcal{V}_B$ be the VC dimension of hyperspheres in $\mathbb{R}^d$. Let $\omega > 0$, $k' = Ck$ and $k = 9(\mathcal{V}_B \ln(2n) + \ln(\frac{8}{\delta}))$,*

$$m \geq \max\{36C\mathcal{V}_B \ln(36\mathcal{V}_B) + 18C\ln(\frac{8}{\delta}), \frac{8Ck\omega}{18\mathcal{V}_B}\}$$

and

$$n \geq \frac{72\ln(8/\delta)}{C\omega}\ln(\frac{18m}{C\omega}).$$

Then with probability $1-2\delta$ the precision cover (algorithm 1) will give a score of 1 to all points $x \in \hat{Q}$ with $(\frac{P(B_{\hat{Q}}(x,\frac{k'}{m}))}{Q(B_{\hat{Q}}(x,\frac{k'}{m}))}) > \omega$.

This theorem is key in our analysis. It shows that for all balls that satisfy the density requirement we will have our measure recognize those points. So for a given minimum density all k-nn balls that are above the density threshold our measure's indicator function will assign a value of 1. This result goes hand in hand with our continuous measure (Definition 1). For a ball of mass β w.r.t. distribution Q, if this ball has mass at least α w.r.t. distribution P we can use the above theorem to show that there exists a number of samples from the true distribution such that our measure will recognize this ball as a sufficiently covered ball in the precision cover measure. The proof is given below:

Proof. This proof has been adapted from [1].

We assume a condition on m:

$$m \geq \max\{36C\mathcal{V}_B \ln(36\mathcal{V}_B) + 18C\ln(\frac{8}{\delta}), \frac{8Ck\omega}{18\mathcal{V}_B}\}$$

which implies that $m \geq k'$ via lemma 3. The condition on k also gives $\frac{\mathcal{V}_B}{k} < 1$. Given all of these along with the condition on n implies

$$n \geq \max\left\{\frac{72m\mathcal{V}_B}{Ck\omega}\ln\left(\frac{18m\mathcal{V}_B}{Ck\omega}\right), \frac{18m\ln(\frac{8}{\delta})}{Ck\omega}, \frac{9m}{C\omega}\right\}. \tag{3.6}$$

It is trivial to see that Equation 3.6 gives

$$\Rightarrow n \geq \frac{9m}{C\omega} \tag{3.7}$$

$$\Rightarrow n \geq \frac{18m\ln(\frac{8}{\delta})}{Ck\omega} \tag{3.8}$$

$$\Rightarrow n \geq \frac{72m\mathcal{V}_B}{Ck\omega}\ln(\frac{18m\mathcal{V}_B}{Ck\omega}) \tag{3.9}$$

The first condition in the max in Equation 3.6 combined with Lemma 3 (this is valid as the second term in Equation 3.6 satisfies the conditions for Lemma 3) yields

$$n \geq 2\Big(4 \times \Big(\frac{9m\mathcal{V}_B}{Ck\omega}\ln(\frac{9m\mathcal{V}_B}{Ck\omega})\Big)\Big)$$

$$\Rightarrow n \geq 2\Big(\frac{9m\mathcal{V}_B}{Ck\omega}\ln(2n)\Big) \tag{3.10}$$

adding inequalities (3.8) and (3.10) we get:

$$2n \geq \frac{18m\mathcal{V}_B}{Ck\omega}\ln(2n) + \frac{18m\ln(\frac{8}{\delta})}{Ck\omega}$$

$$\Rightarrow n \geq \frac{9m\mathcal{V}_B}{Ck\omega}\ln(2n) + \frac{9m\ln\frac{8}{\delta}}{Ck\omega}$$

$$\Rightarrow n\frac{Ck\omega}{3m} \geq 3(\mathcal{V}_B\ln(2n) + \ln(\frac{8}{\delta}))$$

$$\Rightarrow \frac{Ck\omega}{3m} \geq \frac{3(\mathcal{V}_B\ln(2n) + \ln(\frac{8}{\delta}))}{n} \tag{3.11}$$

Now note that by the definition of the k' nearest neighbor balls over the generated samples we have (and recall that $k' = Ck$):

$$\hat{Q}_m(B_{\hat{Q}}(y, \frac{k'}{m})) = \frac{Ck}{m}$$

Note that we choose k such that:

$$k \geq 9(\mathcal{V}_B \ln(2n) + \ln(\frac{8}{\delta}))$$

plugging this choice of k into the definition of k'-nn ball over the generated samples:

$$\hat{Q}_m(B_{\hat{Q}}(y, \frac{k'}{m})) = \frac{Ck}{m} \geq \frac{9C(\mathcal{V}_B \ln(2n) + \ln(\frac{8}{\delta}))}{m}$$

by Lemma 1 we have:

$$\Rightarrow Q(B_{\hat{Q}}(y, \frac{k'}{m})) \geq \frac{Ck}{3m}$$

using the density ratio we obtain:

$$\Rightarrow P(B_{\hat{Q}}(y, \frac{k'}{m})) \geq \frac{Ck\omega}{3m}$$

plugging inequality 3.11 into the above, we get:

$$\Rightarrow P(B_{\hat{Q}}(y, \frac{k'}{m})) \geq \frac{Ck\omega}{3m} \geq \frac{3(\mathcal{V}_B \ln(2n) + \ln(\frac{8}{\delta}))}{n}$$

using Lemma 1 again, we get:

$$\Rightarrow \hat{P}_n(B_{\hat{Q}}(y, \frac{k'}{m})) \geq \frac{Ck\omega}{9m}$$

plugging inequality (3.7) into the above, we readily obtain:

$$\hat{P}_n(B_{\hat{Q}}(y, \frac{k'}{m})) \geq \frac{k}{n}$$

so with probability $1 - 2\delta$ we have that there are k or more points in the k'-nn ball centered around a generated sample $y \in Q$. This then ensures w.h.p. that the precision measure will recognize this k'-nn ball and thus give it a perfect score; ie. the indicator function in the definition of the precision measure is 1. $\square$

3.2.3 Support consistency

Next we show that there is a number of samples n from distribution P that will fully cover the samples of a certain size from Q that fall into the joint support. The following theorem states this more precisely:

Theorem 8. *For $\delta > 0$, and let m, k, and k' satisfy the conditions of Theorem 7. Then there exists a value N for the number of samples such that for all $n > N$, with probability $1 - 3\delta$ over the generated samples, $\hat{P}$ will cover all points from $\hat{Q}$ that fall into* $\text{supp}(P) \cap \text{supp}(Q)$.

As for the other results in this section, it should be noted that due to symmetry the theorem above holds true for when the true distribution is swapped with the generated distribution. For that case the above statement provides an attractive guarantee: For a given dataset size from the true distribution, if we generate sufficiently many points from the generative model's indued distribution, then all points in the intersection of the two supports will be covered that is, our measure will eventually correctly identify all those points from the true distribution that can also be produced with the learned model. The proof is given below:

Proof. We let $\mathcal{X}_Q(\epsilon, \omega) \subseteq \text{supp}(Q)$ denote the subset of points x in the support of Q whose ϵ-ball has weight ratio smaller than w, that is w e define the set $\mathcal{X}_Q(\epsilon, \omega) = \{y \in \mathcal{X}_Q | \frac{P(B_Q(y,\epsilon)}{Q(B_Q(y,\epsilon)} < \omega\}$ (where $Q(B_Q(y, \epsilon)) = \epsilon$). Recall that in our

setup we assumed that both P and Q have continuous density functions d_P and d_Q respectively. This implies that for any sample from either distribution we can pick a ball of mass exactly ϵ, for $\epsilon \in (0, 1]$.

We now introduce a Lemma useful to our analysis, the proof is shown later.

Lemma 9.

$$\lim_{\omega \to 0} Q(\mathcal{X}_Q(\epsilon, \omega) \cap \mathcal{X}_P) = 0$$

Using Lemma 9 there exists a choice of ω small enough such that there is a generated sample size m, so no generated samples y will be in $\mathcal{X}_Q(\epsilon, \omega) \cap \mathcal{X}_P$, in similar fashion to Theorem 7. Pick $\epsilon = \frac{Ck}{3m}$. also note that:

$$Q(B_Q(y, \epsilon)) = \frac{Ck}{3m}$$

There exists a sample size, N, from P we can pick such that this sample set will have perfect precision on $\mathcal{X}_P \backslash \mathcal{X}_Q(\epsilon, \omega)$. Recall our choice of k from prior: $k \geq 9(\mathcal{V}_B \ln(2m) + \ln(\frac{8}{\delta}))$. This then implies:

$$\frac{Ck}{m} \geq \frac{9C(\mathcal{V}_B \ln(2m) + \ln(\frac{8}{\delta}))}{m}$$

By employing the contrapositive of Lemma 1 and the above, we have:

$$\hat{Q}_m(B_Q(y, \epsilon)) \leq \frac{Ck}{m}$$

This implies that there are at most Ck points in the ball $B_Q(y, \epsilon)$ from Q.

Using Theorem 7 (assuming we satisfy the conditions to use), for our choice of ω, C, δ, and m we have a value of N such that:

$$n \geq N = \frac{72m \ln(\frac{8}{\delta})}{C\omega} \ln(\frac{9m}{C\omega})$$

by Theorem 7, this sample size implies that the ball $B_Q(y, \epsilon)$ will have at least k points from P: that is: $\hat{P}_n(B_Q(y, \epsilon)) \geq \frac{k}{n}$. This thus implies (with probability $1 - 3\delta$) that $\hat{P}$ will cover all points from $\hat{Q}$ that have joint support. Our precision measure will give these generated points a perfect score. Thus concluding the proof.

Now we prove Lemma 9:

Proof. First call the set of samples from the generated distribution that have density less than ω: $\mathcal{X}_Q(\epsilon, \omega) = \{y \in \mathcal{X}_Q | \frac{P(B_Q(y,\epsilon)}{Q_Q(B(y,\epsilon)} < \omega\}$, where $B_Q(y, \epsilon)$ is a ball centered at point $y \in Q$ with mass ϵ w.r.t. distribution Q ie: $(Q(B_Q(y, \epsilon)) = \epsilon)$. Now consider an ordered sequence of densities in decreasing order called $[\omega_i]_{i \in \mathbb{N}}$, let this monotonically decreasing sequence converge to 0. We clearly get:

$$\lim_{n \to \infty} \omega_n = 0$$

Now putting this sequence into the set we defined earlier we have:

$$\lim_{n \to \infty} \mathcal{X}_Q(\epsilon, \omega_n) = \bigcap_{n=1}^{\infty} \mathcal{X}_Q(\epsilon, \omega_n)$$

$$\bigcap_{n=1}^{\infty} \mathcal{X}_Q(\epsilon, \omega_n) = \mathcal{X}_Q \backslash \mathcal{X}_P$$

That is, the intersection of the set of all points from Q that have no neighboring points within radius ϵ of them for progressively smaller values of ω converges to the points in support Q that are not in support P.

Taking the probability of the limit sequence w.r.t. probability distribution Q:

$$\lim_{n\to\infty} Q(\mathcal{X}_Q(\epsilon, \omega_n)) = Q(\bigcap_{n=1}^{\infty} \mathcal{X}_Q(\epsilon, \omega_n))$$

$$Q(\bigcap_{n=1}^{\infty} \mathcal{X}_Q(\epsilon, \omega_n)) \leq Q(\mathcal{X}_Q \backslash \mathcal{X}_P)$$

Note that if we take the intersection with $\mathcal{X}_P$ we obtain the limit we are trying to prove:

$$\lim_{n\to\infty} Q(\mathcal{X}_Q(\epsilon, \omega_n) \cap \mathcal{X}_P) \leq Q((\mathcal{X}_Q \backslash \mathcal{X}_P) \cap \mathcal{X}_P) = 0$$

□

□

4 Experiments

In this section we present and discuss a variety of experiments that showcase our measure's behaviour alongside other metrics. This experimental section aims to supplement our theoretical analysis. We present two types of experimental setups: First, through carefully designed tests on synthetic data, we compare our algorithm's (ie. empirical measure's) behavior to previously established measures (Subsection 4.1). In particular, we highlight some failure cases of the earlier measures where our metric succeeds as well as compare the approximation quality in terms of convergence to ground truth. Second, we establish that our measure provides correct assessments in real world settings (Subsection 4.2). Through a variety of experimental setups where we systematically vary the diversity or quality of one of the distributions, we show that our measure adequately follows our design. We do this by tracing the training stages of a VAE, dropping digits from MNIST [7], truncating a GAN or blurring images. In addition, we also verify that our (k, k')-measure is robust to varying the choice of k as long as the ratio between k and

k' remains constant 1/3. This is in contrast to earlier k-nn based measures whose values we show to be highly sensitive to the choice of k.

4.1 Experiments on synthetic data

4.1.1 Sanity checks

We devised several simple toy experiments as sanity checks to see the results of our and earlier measures in a simple and interpretable setting. For a variety of pairs of (true and generative) distributions we assessed their similarity through PR, IPR, DC, as well as our measure PRC.

For this, we used a combination of normal and uniform distributions with varying the degree of overlap and dimensions ranging from 1 to 3. In the most benign settings, we see mostly consistent behaviour among all measures in terms of the degree of overlap, ie matching distributions get perfect scores and as the degree of overlap decreases scores get progressively worse reflecting this. However, the metrics exhibit drastically different behavior in select experiments. For some distributions with disjoint supports, IPR is particularly prone to being overly accepting of samples outside the distribution's support. We also provide an example where the disjoint supports are not recognized by any metric other than PRC. Details and illustration of these experiments have been moved to the appendix for space

reasons.

4.1.2 Convergence quality

We now present comparisons of the three nearest neighbor based measures, namely the improved precision/recall (IPR), density/coverage (DC), and precision/recall cover (PRC) in terms of the convergence behavior and quality of approximation to ground truth. By varying the degree of overlap of two uniform distributions, we establish settings of clear ground truth for precision and recall (namely the probability masses of the joint support [20][19][23][8]).

We use the same value of k for each measure (and the same k-nn computation), and we set the number of samples from the "true distribution" and "generative distribution" to be equal throughout these experiments. For PRC we use a value of k such that $k' = 3k$, as suggested by our analysis. Also, as guided by the analysis, we let the value of k (or k' in the case of PRC) change and grow logarithmically with the sample size; order $(\log(n))$ see appendix for more technical details.

The results for the recall version are shown in Figure 4.1. All measures shown converge to an approximation of the ground truth (orange line). However, as dimension increases our measure converges faster and clearly to a better approximation of the ground truth.

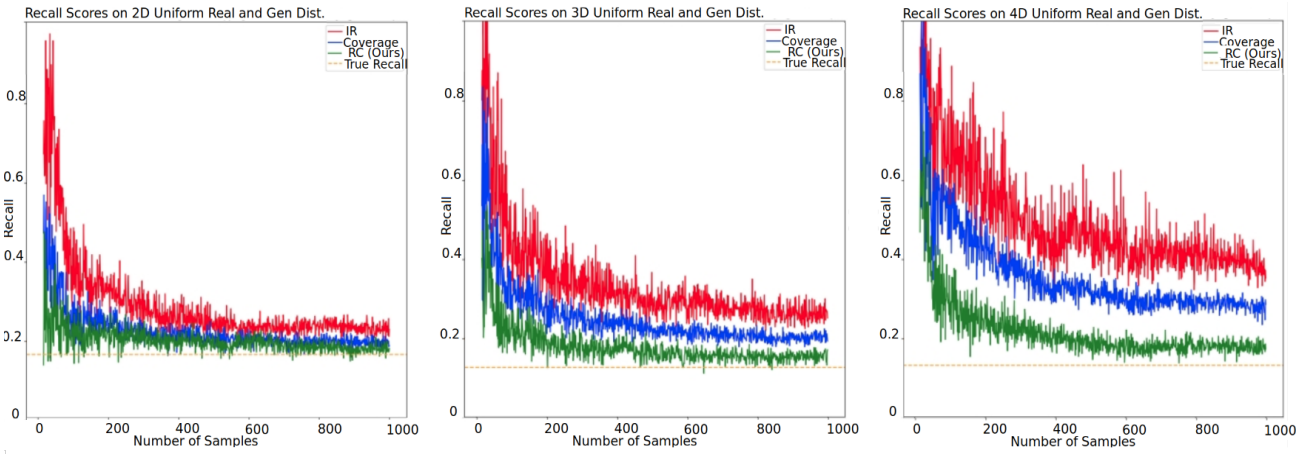

Figure 4.1: Recall convergence experiments for well defined uniform distributions. The orange dashed line is the true recall, the green line is our metric RC (from PRC), the blue line is Coverage (from DC), and the red line is improved recall (IR from IPR). The plots show the results of recall-type metrics from dimensions 2 to 4 (left to right). These plots show the recall-type scores with changing number of samples. All measures seem to converge to an approximation of the true recall (orange-dashed line), but as dimension increases our measure converges faster and to a better approximation.

4.2 Experiments in real world settings

4.2.1 Application to VAEs

In this section we present experiments on a popular generative model that, unlike GANs, gives rise to an explicit generative probability distribution. VAEs are trained by minimizing KL divergence and, if training succeeds, the quality of generated samples gets better as the KL divergence decreases. This suggests an intuitive set of experiments for the precision type measures: training on MNIST image data [7], we track the precision-based scores per training epoch as image quality gets better. Figure 4.2 shows that, among the k-nn based measures, our measure PC is the only one effectively reflecting the increase in image quality.

4.2.1.1 Application to image data directly

We first discuss another set of experiments over the MNIST data: to track the change in recall-type scores with respect to a loss in diversity of generated samples, we successively dropped a digit from the image collection.

We simply start off with original MNIST data as the "real set" and the "generated set". Then, a certain label (a specific digit) is dropped from the "generated set", while randomly taking out samples from the "real set" to ensure the overall sample sizes remain balanced. Figure 4.3 shows that, as we progressively drop

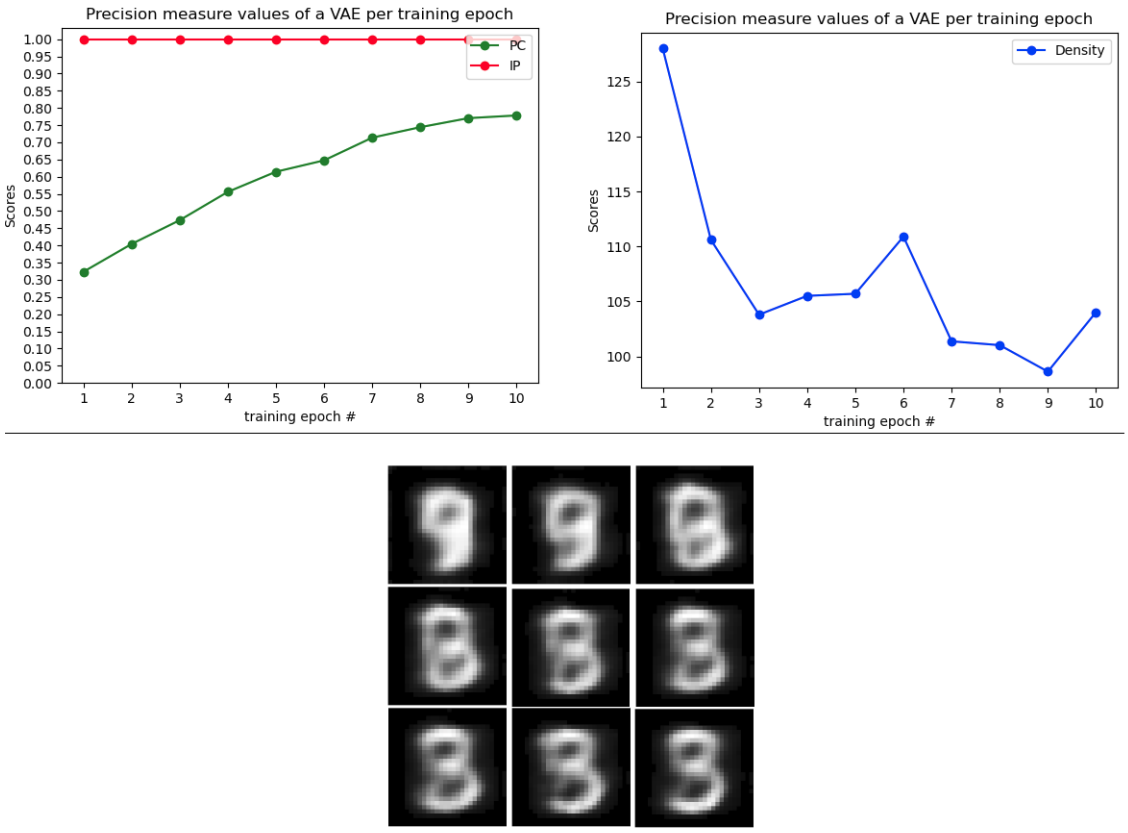

Figure 4.2: Training a VAE, we track the precision type scores per epoch. The plot on the left shows our measure PC (green, ours) and IP (red). The plot on the right shows Density (blue) (from DC). The image collection (bottom image) shows one generated sample image from epoch 1 to epoch 9 (top left to bottom right). These image samples confirm that quality improves as training progresses. Only our measure PC accurately reflects this. Density gives scores of 125 (the measure is not normalized) and decreases non-monotonically and IP stays constant 1 due to its tendency for over-acceptance.

digits, our recall cover measure (RC) correctly reflects the loss in diversity of the "generated set". The RC score drops approximately a 10% for each dropped digit (one out of ten digits dropped at each step).

For a second set of experiments we used the FFHQ dataset, a high resolution 1024x1024 image dataset of human faces [14]. Here we progressively added noise to an image in a fashion similar to that of the forward process of diffusion models [26]. This allows us to control the gradual degradation of quality of the images from a high quality image of a person to pure noise. We expect to see the scores of precision type measures decrease as we add noise and the image quality decreases. The first plot in Figure 4.4 shows results of this setup. As the noise increases, the scores of all measures decreases. We again see the IP has a tendency to be overaccepting and that Density is not normalized (the score starts off with a value larger than 1).

As a second means to induce degradation of image quality and its effect on precision scores, we downsample the images ("generated set") and compare to the original images ("real set").

A downsampling factor of 2 takes an image of resolution 1024x1024 and averages the pixel values of every 2x2 square over the image and replaces the pixel values of each pixel in that square with the average pixel value. The bottom plot of Figure 4.4 shows the scores of PC, IP and Density as we repeatedly downsample. Again we

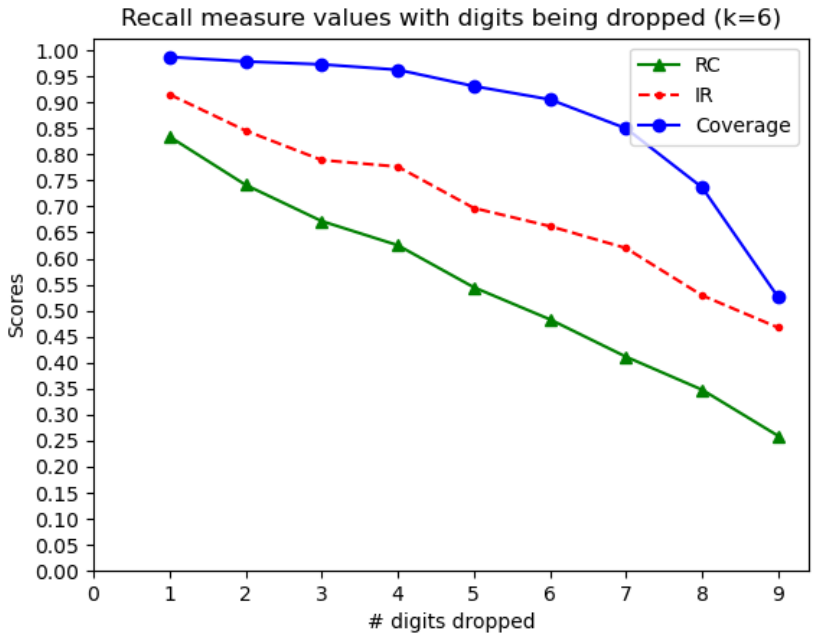

Figure 4.3: Tracking RC (green), IR (red) and Coverage (blue), as we successively drop a digit from MNIST data (comparing to an equal size sample from all digits in MNIST). RC and IR correctly reflect the loss in diversity in the "generated set", while Coverage remains almost constant for quite a while (until more than two thirds of the digits are dropped).

observe that Density is not normalized and IP has a tendency to be overaccepting (while all measures indicate the decrease in quality eventually).

4.2.1.2 Application to GANs

Finally, we explore our measure's performance on evaluating generative adversarial networks (GANs).

Prior work [19, 4, 20] discusses an artificial way of controlling diversity and fidelity via truncation. It is accepted in the literature [4, 19] that the more truncation that is applied the higher the quality of the generated images; however this comes at the cost of less diversity in generated samples. Conversely, with less truncation, it is expected to see more diversity in samples, but worse quality. We design experiments to show that our measure will accurately reflect this diversity-quality trade off with StyleGAN [14] (see Figure 4.5).

4.2.1.3 Robustness w.r.t choice of k

Finally we compare the k-nn based measures on popular datasets while varying k, to test their sensitivity to the choice of k. We again use FFHQ for our real dataset and samples from StyleGAN2 [15]. Figure 4.6 shows the results. While varying the choice of k, (the same bottleneck computation of k nearest distances is used for all measures) our PC and RC measure results in consistent scores. DC and IPR, on

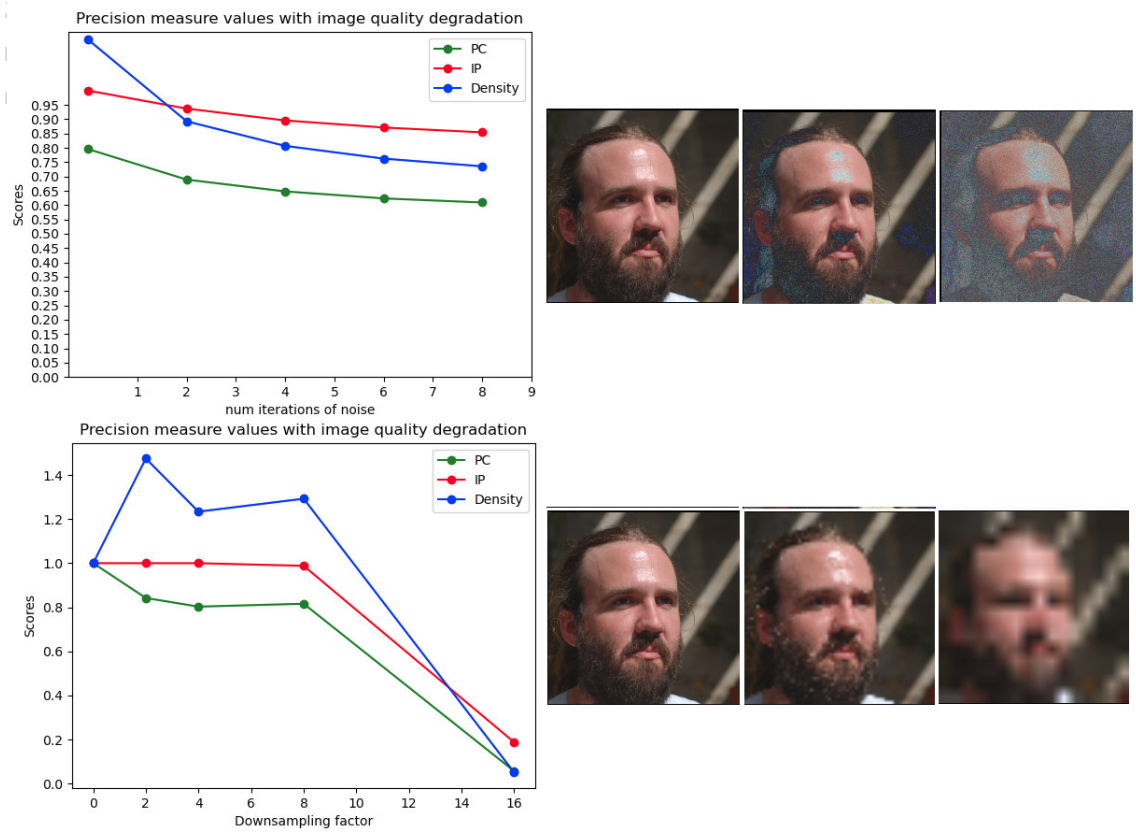

Figure 4.4: We track PC (green), IP (red), and Density (blue) as we decrease image quality of the "generated set" by increasing noise levels (top left graph) or repeated downsampling (bottom left graph). All measures indicate the degradation in image quality, but Density is not normalized and IP has a tendency to be overaccepting. Note that when downsampling, IP gives perfect scores Density remains above 1 until a downsampling factor of 16. The image samples on the right illustrate the gradual degradation in quality by adding noise (top row) or downsampling (bottom row).

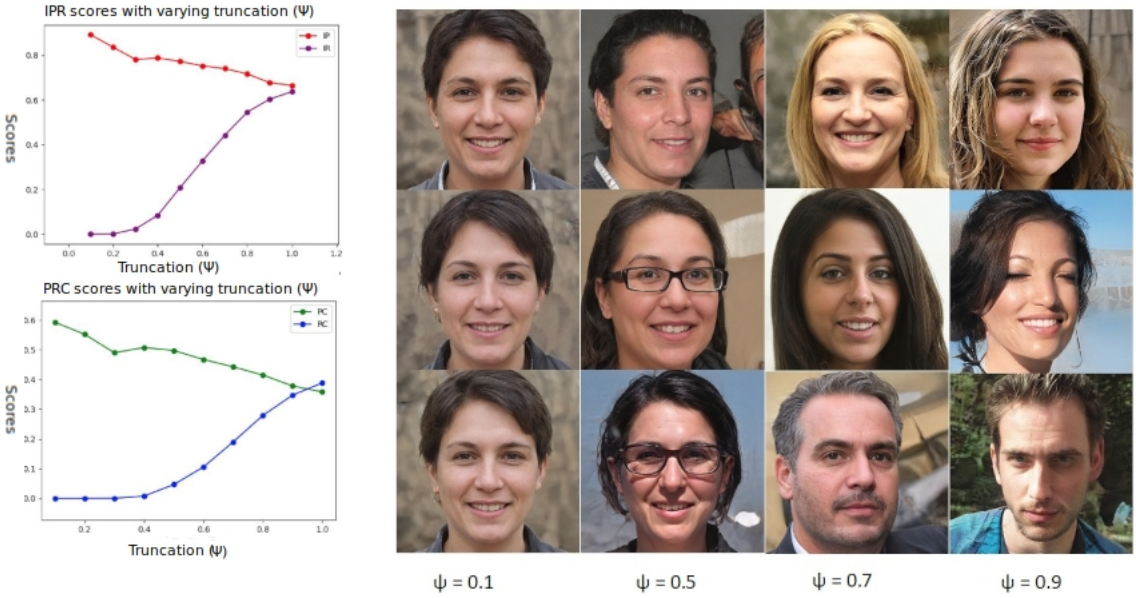

Figure 4.5: Some samples generated by StyleGAN [14] with varying degrees of truncation. We plot our measures RC and PC as a function of the amount of truncation. As desired, we observe that as truncation decreases (maximum truncation at $\psi = 0$, minimal at $\psi = 1$) PC (green) decreases and RC (blue) increases. We also conduct this experiment while tracking IPR (top left), this also gives expected behavior.

the other hand are shown to be sensitive to the choice of k. Our measure enjoys better statistical consistency through the interplay between k and k' (whose ratio remains constant $1/3$, as suggested by our analysis). We believe that a measure not showing brittleness with respect to hyper-parameter choices (such as k in k-nn based measures) provides an important reassurance for practitioners.

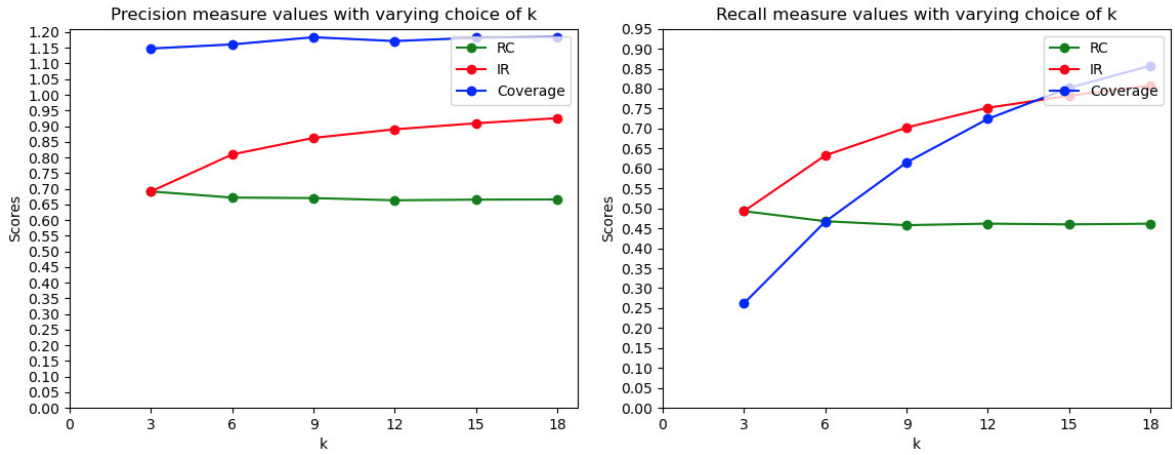

Figure 4.6: The plot on the left shows precision-type measures' results on the FFHQ dataset and StyleGAN2 samples with varying k. The plot on the right shows recall-type measures' results with a varying k on the same dataset and GAN. While DC (blue) and IPR (red) are sensitive to the choice of k, the scores of PRC (green) remain constant when varying k. Our measure enjoys better statistical consistency through the interplay between k and k' (their ratio remains 1/3, as suggested by our analysis).

5 Conclusion

In this section, we will summarize the work and conclude the thesis. As with all works there are some limitations to keep in mind, we will address these in this section. We will also discuss some potential new research directions that can follow from our work.

5.1 Limitations

Our measure along with the IPR, and DC uses the k-nn computations as a core element. It is well known that the k-nn computation suffers from the curse of dimensionality and thus with higher dimensions it becomes a bottleneck in the computation of each method. Both FID and IS do not use a k-nn algorithm and as seen in other works can be faster than other metrics in computing scores to grade generative models [15]. However, also keep in mind there are other drawbacks of using these methods even though they are faster to compute, as many prior works note [19, 20, 23].

Several researchers have suggested to use projections to lower dimension latent spaces for high dimensional data such as image data. These image embeddings are used in almost all previous works for generative modelling metrics, and we stay consistent with the embedding method used by other papers (VGG16) [11, 19, 20, 23, 24]. A drawback for all methods that use embeddings is how accurate those embeddings can be. Whether it is text data, image data, sound synthesis etc. meaningful embeddings are needed for any of these evaluation methods that implement embeddings in their evaluation pipeline. Hence, this is another bottleneck in our method and all comparable methods' reliability.

5.2 Future Work

There are some relevant avenues to explore after our work. One interesting research question would be to conduct a large survey of all evaluation metrics for generative models. Currently, there are many novel methods with very few works to understand what each of these methods is measuring and how well. This work can be used to guide the development on generative models and clear up what are the pertinent methods that should be used when evaluating generative models.

Another interesting research question is diving deeper into the theoretical analysis of our population level measure and algorithm. Although we provide theory throughout this project, a fully extensive analysis of both the algorithm and mea-

sure would be informative. For instance, seeing how the population measure behaves with respect to more well established measures like f-divergences and the KL-divergence.

5.3 Concluding remarks

We have introduced a new evaluation measure for computing and diagnosing differences and similarities between two distributions with access to samples only. Our population level (α, β)-PRC measure and its empirical (k, k')-PRC counterpart are statistically sound while the empirical measure is also algorithmically simple to evaluate. We believe that this combination, namely a sound framework that directly corresponds to the algorithmic implementation, is crucial and prior measures have lacked in offering both of these aspects simultaneously. Our population level measure has some ideal properties that readily pertain to the high level definition of Precision and Recall for evaluating generative models, namely identifying joint support of two distributions. We then give an algorithm that follows from our population level measure. We show some theoretical properties of our algorithm that act as safeties; ensuring our algorithm is well behaved in a variety of settings. These settings include identifying if two distributions are equal eventually (giving a score of 1), identifying local regions of support given enough samples, and finally identifying support consistency between the two distributions. On top, our mea-

sure is based on a natural local test of coverage that can naturally be used as a diagnostic tool, eg to improve a generative model's performance.

In addition to the development, theoretical motivation, and analysis, we have here presented a variety of promising empirical results from applying and comparing our measure in synthetic and real world settings. Starting from simple, easy to explain, experiments we can see glaring issues in other comparable methods. Issues such as overcoverage of IPR and DC, or incorrectly identifying disjoint distributions as similar. We see that in these simple scenarios our method does not face these same issues and is reflective of the expected phenomenon. We then move into more complex experiments where we can see if our algorithm is able to judge the quality and diversity of generated samples. We start with basic real world settings where we train a VAE on MNIST (handwritten digits) that should increase in quality throughout its training process. We see how all relevant quality-type scores fare throughout the training process. Notably, only our method accurately reflects the expected increase in quality, where the other methods give uninformative values (giving a perfect score, or values above 1). We then have a similar experiment to see how comparable diversity-type methods do in a controlled real world setting to gauge diversity. In this experiment, we have the MNIST dataset and randomly pick a digit and drop it. We progressively drop digits and keep repeating until 9 of the 10 digits are dropped. Each time we drop a digit the remaining data is called

the generated samples, we then compute diversity-type scores. We note that all methods show a loss of diversity, however our method shows the most accurate loss of diversity each time a digit is dropped. Again, IPR and DC seem to give generous scores each time a digit is dropped. Finally, in the last set of experiments we move into one of the most relevant settings; high quality images. Again we would like to see if our quality type score is accurate in a real world scenario. We artificially destroy the quality of images via downsampling and adding noise. We see that all comparable quality type scores do show a loss of quality but again the other methods are overly accepting and sometimes even increase in score before going down, counter to the expected deterioration of the images. We also verify another expected behaviour in generative models; the quality-diversity trade-off in GANs with truncation. We note that, as with IPR, our method also reflects this expected behaviour. This is promising as it shows in a setting for high level generative models where we expect to see a quality-diversity trade-off our quality/diversity scores accurately reflect this and catch this phenomenon. Finally, we have an experiment that shows our method is not sensitive to a specific choice of k in the k-nn computation. We compute all comparable scores after changing the k values, to see how this will affect the scores. We see that other methods inflate their scores as k increases whereas our scores remain robust to the changing of k. This shows another promising trait of our method where practitioners need not worry about

the choice of k and how sensitive their results will be dependent on that choice.

In conclusion, we introduce a theoretically founded population level measure evaluated on two distributions. We then derive an algorithm from this measure and prove some theoretical properties of our algorithm. Finally, we show experiments where can verify if our method is measuring the quality or diversity of generated samples alongside comparable methods. We see that our algorithm shows very promising behavior in many applicable settings where other state-of-the-art methods do not. We hope to see this method be implemented and used by practitioners in the field of generative AI.

Bibliography

[1] Christopher Berlind and Ruth Urner. Active nearest neighbors in changing environments. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1870–1879. PMLR, 2015.

[2] Ali Borji. Pros and cons of GAN evaluation measures: New developments. *Comput. Vis. Image Underst.*, 215:103329, 2022.

[3] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning, ML Summer Schools 2003*, volume 3176 of *Lecture Notes in Computer Science*, pages 169–207. Springer, 2003.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

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

[6] Sanjoy Dasgupta and Samory Kpotufe. Optimal rates for k-nn density and mode estimation. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[7] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[8] Josip Djolonga, Mario Lucic, Marco Cuturi, Olivier Bachem, Olivier Bousquet, and Sylvain Gelly. Precision-recall curves using information divergence frontiers. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020*, pages 2550–2559, 2020.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[10] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *JSTOR: Applied Statistics*, 28(1):100–108, 1979.

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[12] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. doi: 10.1126/science.1127647.

[13] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018.

[14] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. Computer Vision Foundation / IEEE, 2019.

[15] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12104–12114. Curran Associates, Inc., 2020.

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[17] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, nov 2021.

[18] Samory Kpotufe. k-nn regression adapts to local intrinsic dimension. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24, 2011.

[19] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019.

[20] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pages 7176–7185. PMLR, 2020.

[21] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021.

[22] Murray Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27:832 – 837, 1956.

[23] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, 2018.

[24] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[25] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press, 2014. ISBN 978-1-10-705713-5.

[26] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In

Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, Lille, France, 2015. PMLR.

[27] Esteban G. Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood . *Communications in Mathematical Sciences*, 8:217–233, 2010.

[28] Tim van Erven and Peter Harremoes. Ré nyi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, jul 2014.

[29] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. doi: 10.1137/1116025.

[30] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks, 2018.

6 Appendix

6.1 Experiments

For all experiments we intend to make the code publicly available on GitHub. In the vein of reproducibility we choose to provide a detailed summary of experiments here in addition to extra images and results.

6.1.1 Toy Models

As stated in the main work, we devised several toy experiments as a sanity check for our measure. In these settings, we start off with two synthetic distributions as stand-ins for the "real distribution" and the "generated distribution". The real distribution and generated distributions are either both uniform distributions or Gaussian distributions in most of our setups. We use numpy to randomly sample points from their respective distributions. We create the distributions in 1 to 3 dimensions for visualization purposes and use the same random seed throughout the experiments. We have conducted a wide set of experiments where we vary

many factors in a controlled setting; we vary the amount of overlap between the two distributions, the number of samples for each distribution, the dimension, and the shape of the distributions. We will show several figures to illustrate the setup of the experiment and some of the results. In most of these toy models we show a visualization of the data (on the left) and on the right the Precision-Recall Curve from [23] with the scores from the other measures on the top right of the figure. For the toy models shown here we have opted to use a choice of $k = 9$ for DC and IPR, and for PRC we use a choice of $k' = 9$ and $k = 3$. See Figure 6.1 for a 1 dimensional case of matching distributions. See Figure 6.2 for a case of 2-dimensional distributions that are partially overlapping, in this example we also have the true distribution have 3 times as many samples as the generated distribution has. In fig 6.3 we have 3 dimensional uniform distributions that are essentially disjoint.

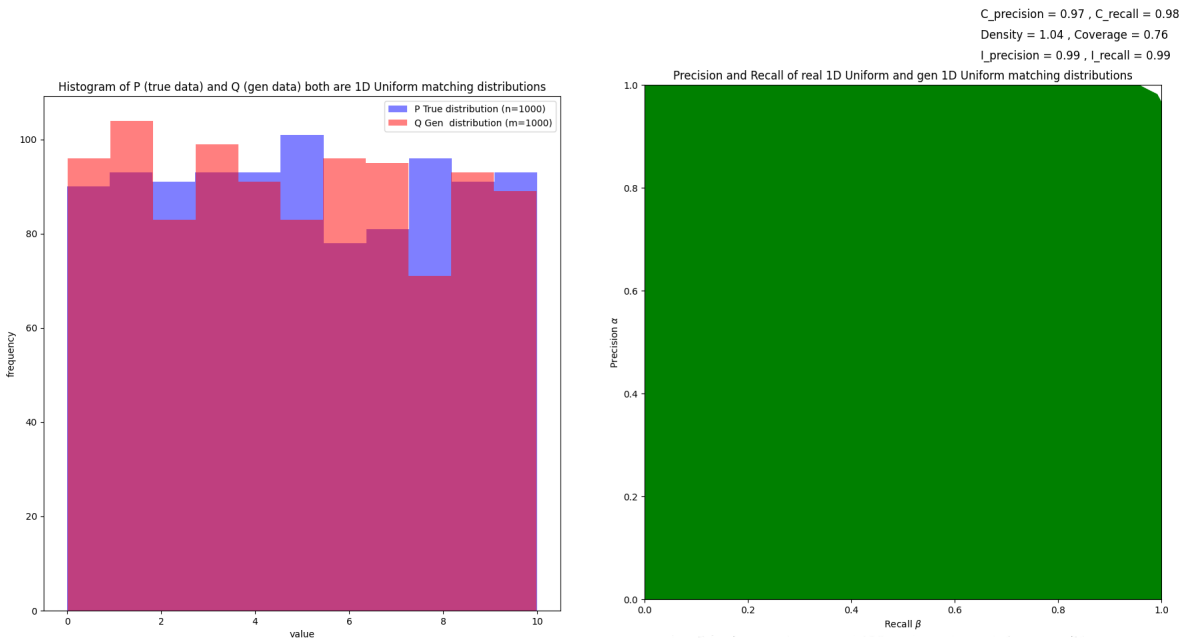

Figure 6.1: The true distribution (blue) and the generated distribution (salmon) are both matching (uniform from 0 to 10). See visualization of the 1-dimensional data (in form of a histogram of the samples) on the left. Both sample sets have the same size (1,000 samples). We use $k = 9$ for DC and IPR and $k' = 9$ and $k = 3$ for PRC. The PR curve (right) is filled in implying perfect precision and recall. All other measures also reflect the matching distributions (see top right for scores). We observe that density and coverage (DC) is not normalized, while density takes on a value larger than 1 (even if only slightly), coverage is significantly below 1. This shows that the scores of this measure are difficult to interpret, and thus might be misleading.

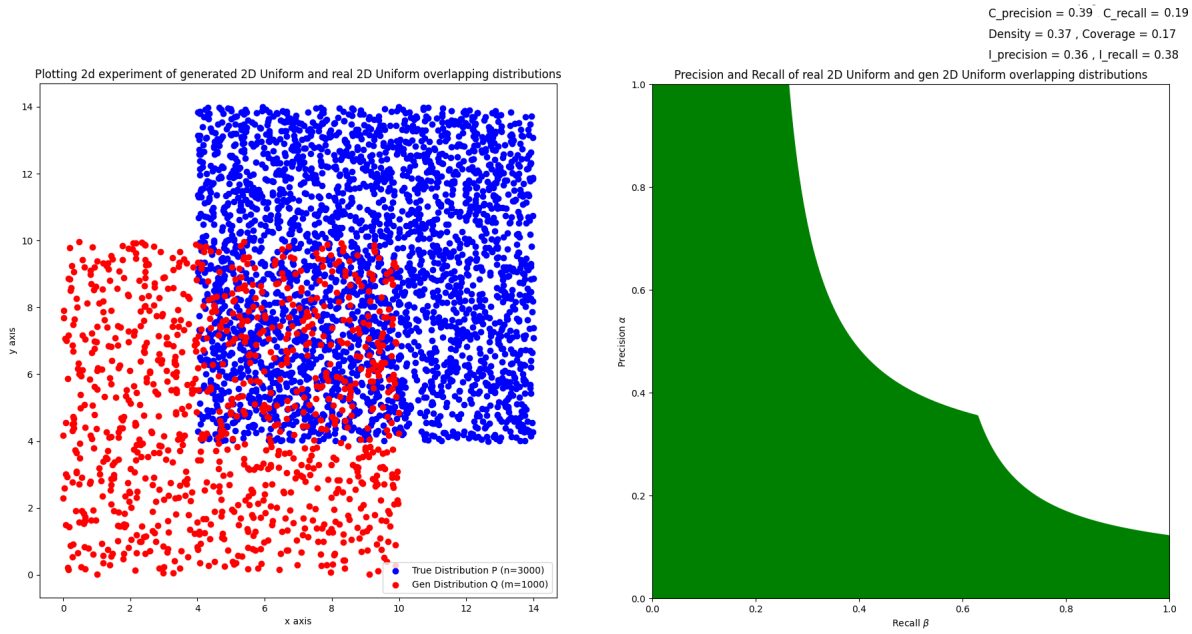

Figure 6.2: In this example we sample from two distributions; a 2 dimensional uniform square from 0 to 10 for both x and y axis (gen distribution in red), and a uniform square from 4 to 14 on both x and y axis (real distribution in blue). The scores mostly all reflect the partial overlap (see the PR curve right). Notice that in this example we have far more samples from the true distribution than the real (3000 samples from real dist. vs 1000 samples from gen dist.). This is reflected in the precision type scores of all measures (see top right). Note that the two distributions have equal area and have an overlapping area of 36%, as supported in the theory, since there are more true samples than gen samples we see that the precision scores are closer to the overlapping area amount than the recall-type scores.

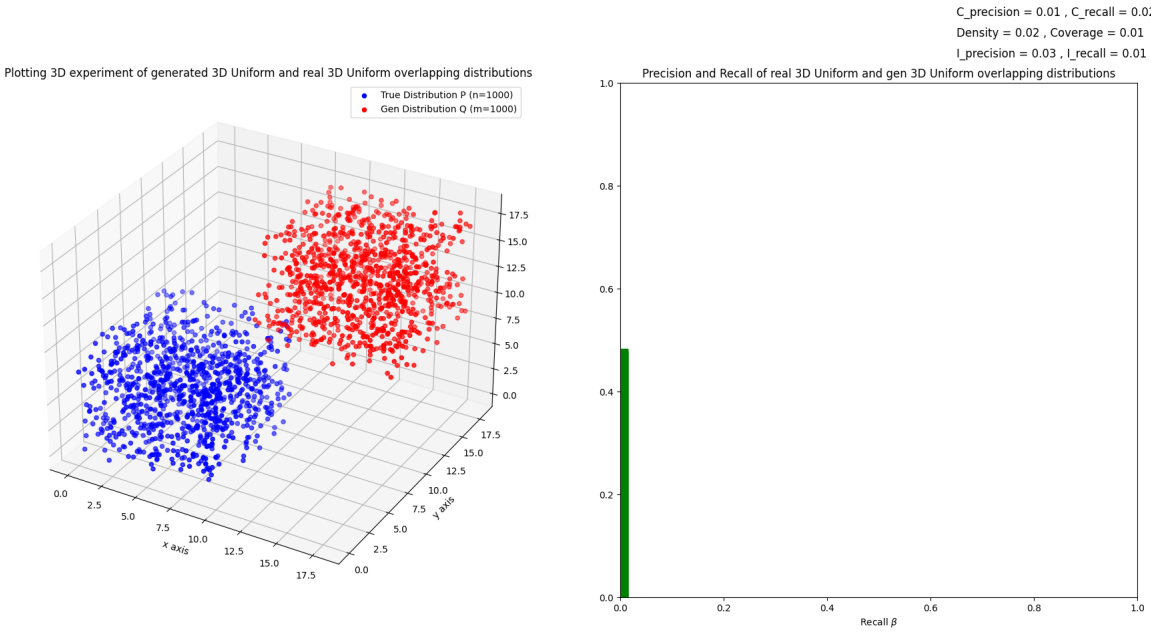

Figure 6.3: In this example we have two 3-dimensional cubes (see data on the left); a true uniform distribution from 0 to 10 on the three axes (in blue) and a generated uniform distribution from 10 to 20 on the three axes (in red). In this case the 2 distributions are essentially disjoint (one shared boundary points exists), and all measures reflect this. The PR curve (right) is empty implying no precision or recall, and the other measures almost all give scores of 0 (top right).

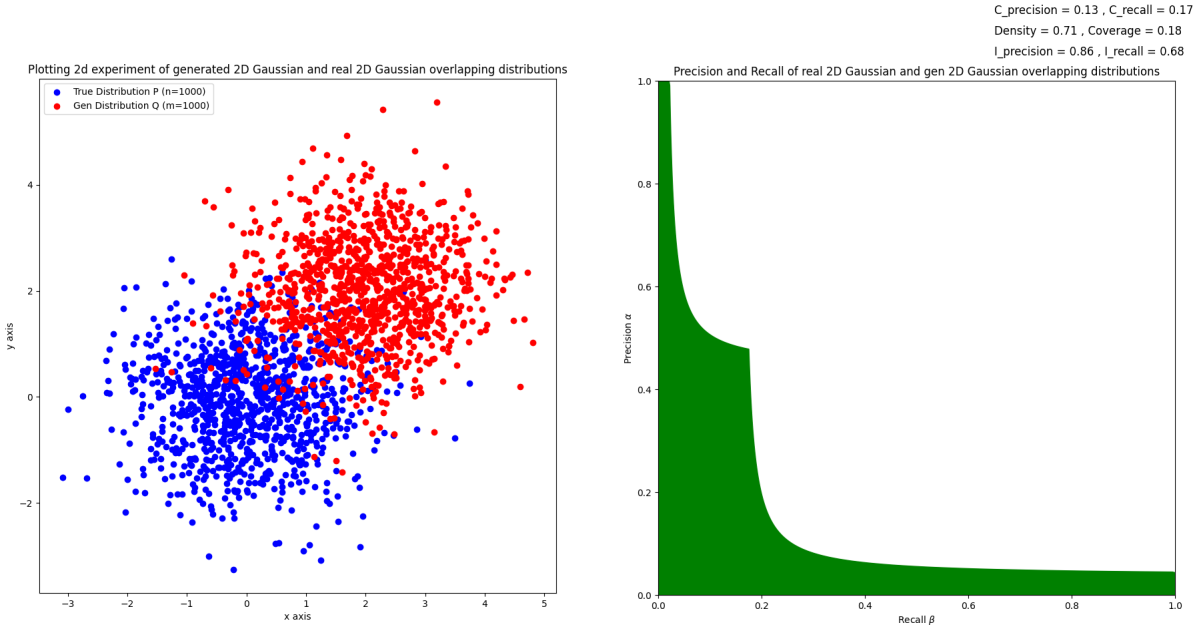

Figure 6.4: In this example we have a 2D Gaussian with mean 0 and std of 1 for both x and y axes as the true dist. (blue) and a Gaussian with mean 2 and std of 1 for both axes as the gen dist. (red). As we can see in the figure (left), although the data is overlapping most points from both distributions do not overlap and there is a clear distinction between the two distributions. On the right the PR curve shows that there is a discrepancy between the two distributions (seen in the lack of the PR curve being filled in), our measure PRC keeps a lower score (about 0.15) reflecting the lack of overlap between the distributions. In contrast, DC has a high density score yet retains a lower coverage score (top right). IPR has scores above 65% which is not appropriate for these two distributions. An example of the over-accepting nature of this method.

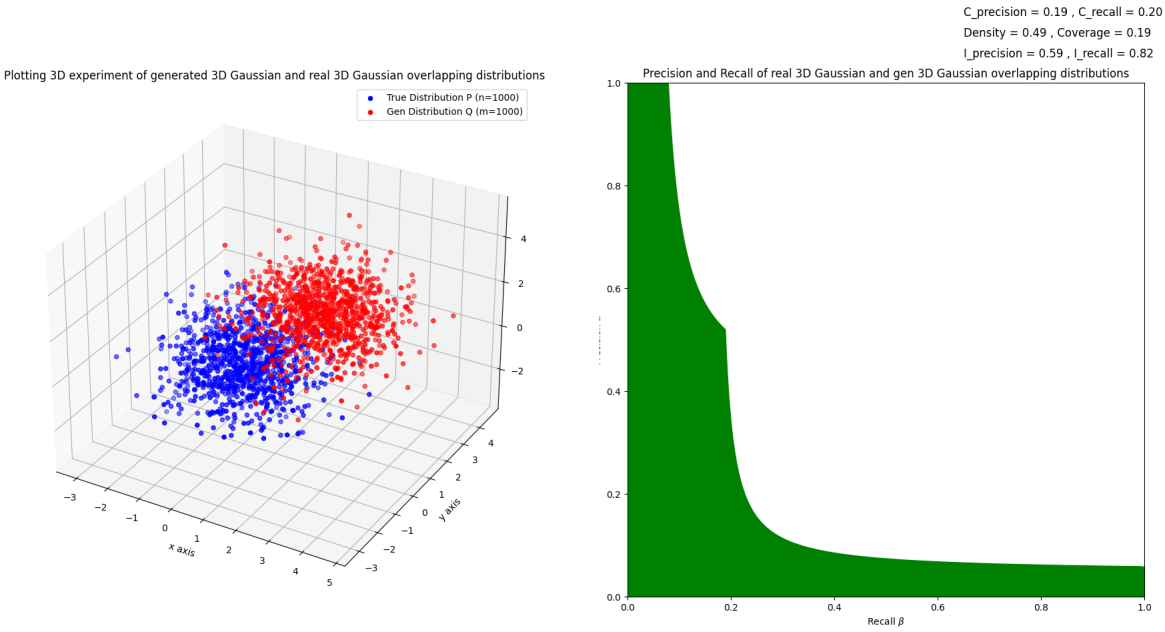

Figure 6.5: In this example we have a 3-D Gaussian with mean 0 and std of 1 all three axes as the true distribution (blue) and a Gaussian with mean 2 and std of 1 for all three axes as the gen distribution (red). Although there is overlap in the data from these two distributions, most points are located in distinct areas, clearly belonging one of the two distributions. On the right the PR curve shows there is a discrepancy between the two distributions (seen in the lack of the PR curve being filled in), our measure PRC keeps a lower score (about 0.20) reflecting the lack of overlap between the distributions. In contrast, DC has a high density score yet retains a lower coverage score (top right). IPR has the highest scores above 50% which is not the nature of the two distributions. This is another example of the over-accepting nature of this measure.

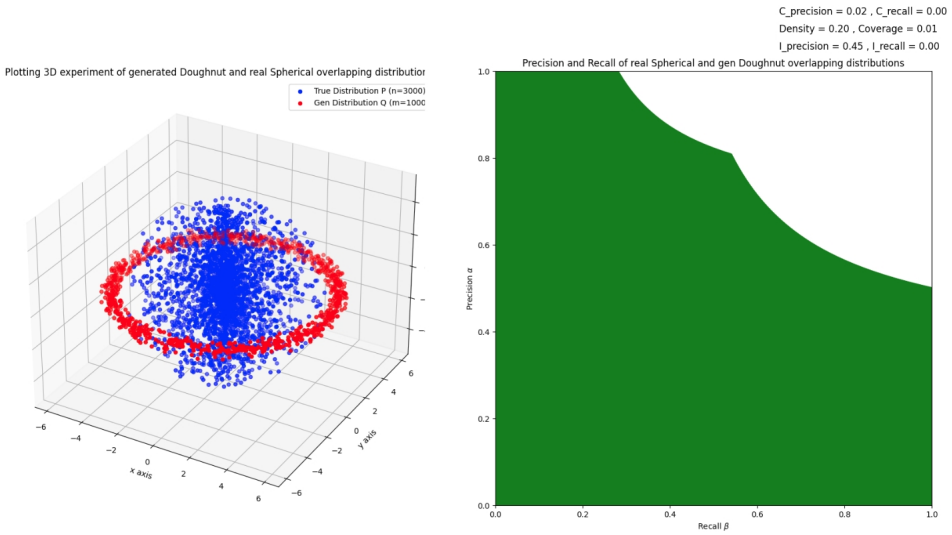

Figure 6.6: The PR Curve (right) gives a very high scores on both precision and recall (indicating similar distributions with large overlap) while the two distributions are in fact disjoint (data on left). The PR curve is almost fully filled in implying both high precision and recall. The doughnut (red; generated samples) does not overlap with the sphere (blue: real samples). Other metrics have the same issue: Density = 0.20, Coverage = 0.06, Improved Precision = 0.45, Improved Recall = 0.80. Only PRC correctly scores these sets as disjoint with PC = 0.02, and RC = 0.00

We observe many interesting behaviors that support our theory and highlight issues of other measures. Some interesting observations we observe is that if we sample from the true distribution more than the gen distribution or vice versa we can see scores be skewed (this supports some of the theory we have developed

see theorem 7), this behaviour of our measure is empirically reflected in fig 6.2. Another behaviour we have noted in this simplistic setting is that DC and IPR are prone to be more over-accepting than our measure. This is also true in settings where the distributions have significant discrepancy between them, but DC and IPR incorrectly say they are similar distributions (see fig 6.4, 6.5, 6.6). In fig 6.6 we can see a case of where two distributions are completely disjoint yet all other measures identify these distributions as similar. This again adds to the point of PR, IPR and DC being prone to over-accepting. Only our measure accurately reflects the disjoint nature of the two distributions. This experiment highlights this worrying behavior that will be seen again in later experiments in real world data.

6.1.2 Convergence Experiments

In these sets of experiments we use simple uniform distributions in increasing dimensions to define true values of precision and recall. We construct 2 uniform hypercube distributions for the real and generated distributions and vary the degree of overlap. In this case true precision becomes the overlapping volume over the generated distribution's volume, and conversely true recall is the overlapping volume over the true distribution's volume. The setup of this experiment is as follows: we fix a configuration for a true distribution and generated distribution with a certain amount of overlap. We set the number of samples from the true and

gen distribution to be the equal $n = m$. We start at 15 samples for both distributions, obtain $n = m$ samples randomly for both distributions given a particular configuration of true and generated distributions (we use a random seed for reproducibility). Then, we compute all k-nn based measures (IPR, DC, and PRC) for the current sample size. We compute k-nn based measures only because they are directly comparable as the same bottleneck computation is done for all 3 measures (the distances between sets and within sets). Then we increment the sample size and sample from both distributions again for the given configuration of real and gen distribution and compute measures. We repeat this process until the terminal number of samples from both distributions (we choose to stop the experiment when $n = m = 1000$). as $n = m$ increases we adjust our value of k. We use a value of k inspired from the theory of order log(n). This value of k is the same for DC and IPR and PRC uses this same value for k' while keeping the ratio fixed of $C = 3$ for our (k, k')-based measure. More specifically, k ranges from 9 to 13 in our set of experiments. Figure 4.1 in the main part of the paper shows that our measure converges faster to the true values of precision and recall, it can also be seen that some of the other measures do not seem to converge to the true values at all (see higher dimensional setting of experiment). We also have included the experiments for the precision type experiments from increasing dimensions below.

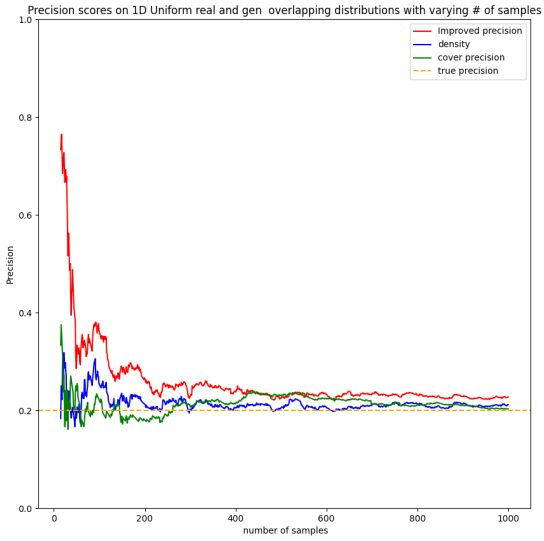

Figure 6.7: We compute improved precision, precision cover, and density for a fixed configuration of 1-dimensional uniform distributions. The true distribution is a uniform distribution over the interval from 0 to 10. The gen distribution is a uniform distribution from 8 to 18. This corresponds to a true precision of 0.2 for this configuration of distributions. We set $n = m$ start at 15 samples and go to 1000 samples. We can see IP (red), Density (blue), and PC (green) all converge to the true value of precision (dashed orange line).

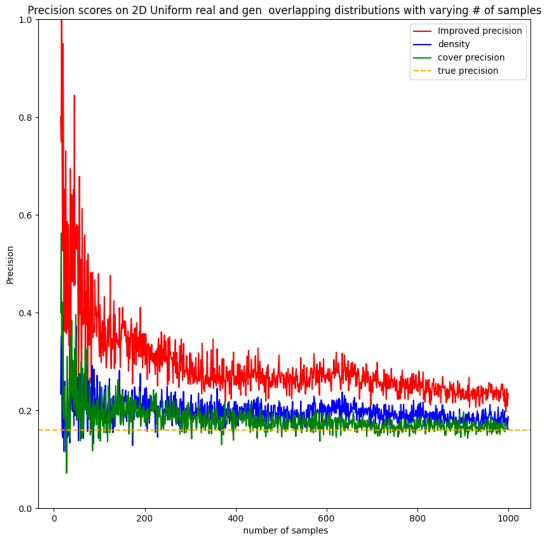

Figure 6.8: We compute improved precision, precision cover, and density for a fixed configuration of 2-dimensional uniform distributions. The true distribution is a uniform distribution over the interval from 0 to 10 in both x and y axes. The gen distribution is a uniform distribution from 6 to 16 for both x and y axes. This corresponds to a true precision of 0.16 for this configuration of distributions. We set $n = m$ start at 15 samples and go to 1000 samples. We can see IP (red), Density (blue), and PC (green) all converge to the true value of precision (dashed orange line).

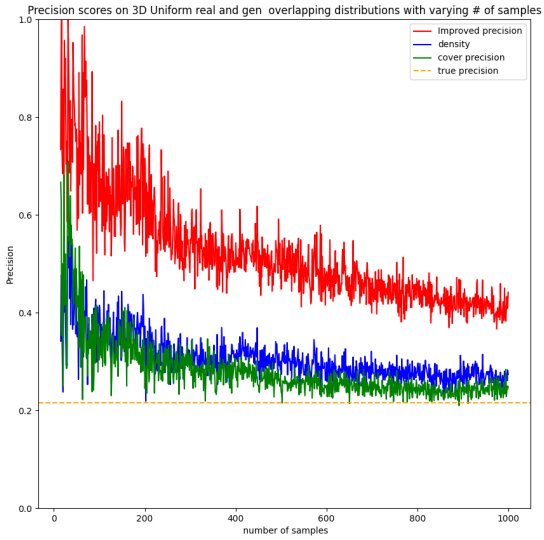

Figure 6.9: We compute improved precision, precision cover, and density for a fixed configuration of 3-dimensional uniform distributions. The true distribution is a uniform distribution over the interval from 0 to 10 in x, y and z axes. The gen distribution is a uniform distribution from 4 to 14 for x, y and z axes. This corresponds to a true precision of 0.216 for this configuration of distributions. We set $n = m$ start at 15 samples and go to 1000 samples. We can see IP (red), Density (blue), and PC (green) all converge to the true value of precision (dashed orange line).

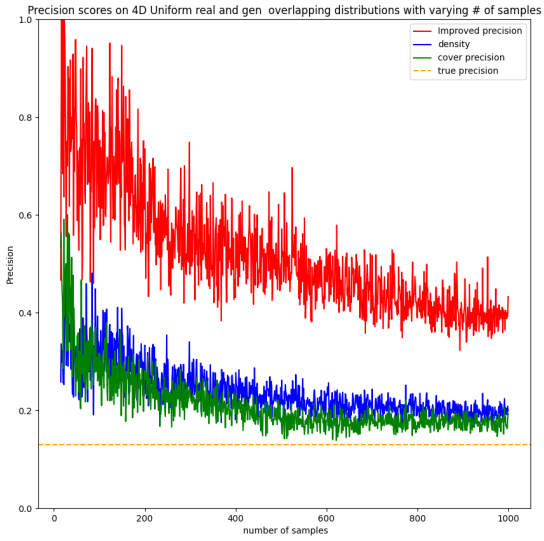

Figure 6.10: We compute improved precision, precision cover, and density for a fixed configuration of 4-dimensional uniform distributions. The true distribution is a uniform distribution over the interval from 0 to 10 in all 4 axes. The gen distribution is a uniform distribution from 4 to 14 for all axes. This corresponds to a true precision of 0.1296 for this configuration of distributions. We set $n = m$ start at 15 samples and go to 1000 samples. We can see IP (red), Density (blue), and PC (green) all converge to the true value of precision (dashed orange line).

6.1.3 VAE experiments

For the VAE training experiments we use the MNIST data set [7]. We use a simple VAE that uses the MNIST dataset for training (60,000 samples) and testing (10,000

samples). We use the adam optimizer and train the VAE for 10 epochs. The test set is used as the real distribution and we generate 10,000 samples from the model (to match the testing set size) as the generated distribution. We convert these images to gray-scale, so they are of dimension 28x28x1. In these sets of experiments we expect the measures to reflect the increase of quality as epochs increase. From fig 4.2 it is clear only our measure shows the desired behavior of increasing precision (quality) scores.

6.1.4 Digit Dropping

Since we have a nice set of controlled experiments to see an increase in quality in a real world setting with the VAE training experiment, we wish to do the same for our diversity based measures (recall-type scores). In this setting we design an experiment to artificially control for diversity of a dataset. We first start off with the complete MNIST dataset (70,000 samples). Then we choose a number of samples for the real and generated dataset ($n = m$), in this case 5,000 samples, for both real and gen distributions. From the complete dataset we randomly select 5,000 samples and call this the real sample set. We then randomly pick 1 digit to drop, and use a function to eliminate all samples with that digit's label from the potential generated dataset pool. From the MNIST dataset (now excluding the dropped digit) we sample 5,000 samples and call this the generated sample set. We

compute improved recall, coverage, and recall cover over the real and generated sample sets. We then drop another digit and repeat the process until 9 digits (out of 10) are dropped. For this experiment every time prior to random sampling we use and record the value of the random seed used.

6.1.5 Experiments with GANs

For experiments with GANs we used the well developed pipeline of methods for assessing high resolution image data for quality metrics. Many other prior works have used this pipeline to test out measures for grading generative models [23, 19, 20]. The pipeline consists of a network or method to project the high dimensional (usually image) data into a lower dimensional latent space. In this latent space we then have our measure compare the data and obtain a final score for PRC. We use the pipeline available from [15], which uses VGG16 to obtain image embeddings. Our approach here follows consistent methodology in the literature [20, 23, 19]. We use the pretrained GAN and compute several other popular measures on the generated and real data. In our experiments we use StyleGAN and StyleGAN2.

For the noise experiments we add noise iteratively as done in diffusion model [26, 21]. We use a linear noise schedule starting from 0.000001 ending at 0.0000075 with a 300 steps. We use small amounts due to the sensitive nature of the pipeline; ie enough noise to degrade image quality to human perception but not enough such

that the feature extractors can no longer properly operate on the image. We notice that by adding a relatively significant amount of noise all measures give scores of 0.

Another set of experiments to control for quality or lack thereof is the downsampling set of experiments. In this set of experiments we use a value for the stride to downsample called s. We move an sxs square over the image (usually a high resolution 1024x1024 image) and average the pixel value over the square and assign each pixel in that square the value of the average. For instance, if we have a stride of 2, each 2x2 square is averaged and that average pixel value is given to each pixel in that square. This is akin to losing information by resizing an image of size 1024x1024 to size 512x512 and then resizing back to 1024x1024. This loss of pixel information can be seen to have a blurring effect on the quality of an image, see 4.4.