KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Evaluation of generative machine learning models

Judging the quality of generated data with the use of neural networks

SAM YOUSEFZADEGAN HEDIN

Stockholm, Sweden 2022

Evaluation of generative machine learning models

Judging the quality of generated data with the use of neural networks

SAM YOUSEFZADEGAN HEDIN

Master's Programme, Machine Learning, 120 credits
Date: June 26, 2022

Supervisor: Ludvig Ericson
Examiner: Patric Jensfelt
School of Electrical Engineering and Computer Science
Swedish title: Evaluering av generativa maskininlärningsmodeller
Swedish subtitle: Evaluering av genererad data med hjälp av neurala nätverk

Abstract

Generative machine learning models are capable of generating remarkably realistic samples. Some models generate images that look entirely natural, and others generate text that reads as if a human wrote it. However, judging the quality of these models is a major challenge. Today, the most convincing method is to use humans to evaluate the quality of generated samples. However, humans are biased, costly, and inefficient. Therefore, there is a great need for automatic methods.

MAUVE is a recent advancement in the evaluation of generative text models. It compares generated data with real data and returns a score that quantifies their similarity. This is accomplished with the help of a neural network, which provides the understanding of text required to evaluate its quality. MAUVE is motivated by its correspondence with human judgment, and this is shown in multiple experiments.

This thesis contributes in two significant ways: First, we complement experiments and discussions made in the original paper. Importantly, we demonstrate that MAUVE sometimes fails to recognize quality differences between generative models. This failure is due to the choice of neural network. Later, we demonstrate that MAUVE can be used for more than just text evaluation. Specifically, we show that it can be applied to images. This is accomplished by using a neural network specialized in image recognition. However, the steps can be repeated for any data type, meaning that MAUVE can potentially become a more generalized measurement than suggested in the original paper.

Our second contribution is an extension to MAUVE called Sequence-MAUVE (S-MAUVE). The score MAUVE produces can be seen as an average of the overall quality of generated text. However, some generative models initially produce excellent text, but see drops in quality as the sequences grow longer. Therefore, a single score that represents entire sequences is likely to omit important details. Instead, S-MAUVE evaluates generated text at the smallest possible level. The result is a sequence of scores, which give users more detailed feedback about the behavior of a generative model.

Keywords

Generative Modeling, MAUVE, Deep Learning, GPT-2, evaluation

Sammanfattning

Generativa maskininlärningsmodeller kan generera data av enastående kvalitet. Vissa modeller genererar bilder av ansikten som ser helt realistiska ut, och andra genererar text som verkar varit skriven av en människa. Trots detta så är det inte klart hur dessa modeller ska evalueras. Idag så är den främsta metoden mänsklig evaluering: En person får utgöra huruvida generade data verkar realistisk eller inte. Mänsklig evaluering har flera nackdelar. Människor är partiska, dyra och långsamma. Därför behövs det automatiska evalueringsverktyg.

MAUVE är ett ny metod för att evaluera generative textmodeller som jämför hur lik genererad data är med äkta data. Detta åstadkoms med hjälp av ett neuralt nätverk, som bidrar med den förståelse av text som krävs för att evaluera den. MAUVE är motiverat av att dess omdömen överensstämmer med mänsklig evaluering.

Den här uppsatsen bidrar på två sätt. Till att börja med komplementerar vi experiment och diskussioner gjorda i den ursprungliga rapporten om MAUVE. Till exempel så visar vi att MAUVE ibland inte lyckas känna av kvalitetsskillnader mellan olika generativa modeller. Detta på grund av val av neuralt nätverk. Efteråt så demonstrerar vi att MAUVE kan appliceras på andra typer av data än text. Mer specifikt så applicerar vi MAUVE på bilder. Detta åstadkoms genom att använda ett neuralt nätverk specialiserat på bildigenkänning, istället för text. Stegen vi följer kan upprepas för vilken typ av data som helst, vilket innebär att MAUVE kan användas som ett mer generellt mått än vad den ursprungliga artikeln ger sken för.

Vårt andra bidrag är att utveckla MAUVE till det vi kallar för S-MAUVE. MAUVE använder bara sammanfattningar av hela texter som bas för sina jämförelser. En konsekvens av det är att den endast gör påståenden om textdatas genomsnittliga kvalitet. Men, det är välkänt att kvaliteten hos genererad textdata kan variera beroende på var i texten man befinner sig. Många generativa textmodeller producerar sekvenser som är verklighetstrogna i början, men blir sämre och repetitiva senare. Till skillnad från MAUVE så evaluerar S-MAUVE genererad text på minsta möjliga detaljnivå. Resultaten är en sekvens av poäng, som ger användare mer information om egenskaperna hos den studerade generativa modellen.

Nyckelord

Generativ modellering, MAUVE, Djupinlärning, GPT-2, evaluering

Acknowledgments

Thanks to my supervisor Ludvig Ericson, for continuously providing insightful feedback and ideas, as well as giving me access to his hardware.

Further, additional thanks to Ludvig as well as my examiner Patric Jensfelt: Thank you for allowing me to pursue this thesis, even when it was evident that the connection to your respective fields was questionable at best.

Thanks to the authors of [1] for allowing me to use their figures in this report. Reading their paper made me interested in this topic in the first place.

And lastly, thanks to the creators of MAUVE [2], who have quickly and thoroughly responded to all of my questions.

Stockholm, June 2022
Sam Yousefzadegan Hedin

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

FID	Fréchet Inception Distance
IS	Inception Distance
IV3	Inception V3
S-MAUVE	Sequence-MAUVE

Chapter 1

Introduction

Generative modeling is one of the most fundamental tasks in machine learning. While one of its main applications is the generation of realistic data, like in Figure 1.1, it is far from the only one. To generate realistic data, the needs intimate knowledge of it and such knowledge facilitate many other tasks. For example, a model that generates images of cats or dogs can easily be turned into a classifier that decides if a given image contains a cat or a dog [3].

As impressive as contemporary generative models are, it is not clear how to evaluate their performance. This is unfortunate as, without the ability to evaluate generative models, there is no way to compare them. This makes research in the field difficult, as there is no way to know if a suggested method is effective or not.

As an example, the generative image model PixelCNN [5] was released in 2016. Just a year later, PixelCNN++ [6] was announced, claiming to improve on the older method. However, supporting this claim with hard numbers is

Figure 1.1: A dog generated by the imagen generative model [4].

today almost impossible.

Currently, the most common method to evaluate generative models is using humans. For example, a human evaluates a generative image model by inspecting generated images. If the images look natural, the model is good. Meanwhile, the model is not good if they are easily recognized as fake.

There are numerous issues with this approach: Humans are slow and expensive to work with, and each individual has their own set of biases and preconceptions. Moreover, humans are mainly good at judging the quality of data types such as images or text. A human judge may be of little use if the task is to generate sensor data. Even for images and text, humans are not perfect, as they may fail to recognize subtle but genuine differences in quality.

Because of these issues, this thesis is about automatic methods to evaluate generative models. Many such methods exist, but all of them are, at best, complementary to human evaluation.

1.1 Contributions

MAUVE [2] is a recent method for evaluating the quality of generated text. It takes two sets of text samples as input and produces a single score to indicate how similar they are. In general, one set consists of text written by humans, and the other of text from a generative model [2]. The soundness of the method is motivated by its correlation to human judgment. In multiple experiments, human evaluators and MAUVE agree on to which extent generated text samples appear human [2].

Our work investigates and extends MAUVE. We have divided our contributions into two chapters.

Chapter 4 investigates potential use-cases. We complement experiments and discussions made in the original paper with our own. The chapter continues by adapting MAUVE to work with image data. By showing that MAUVE agrees with humans in situations where human evaluation is available, we increase confidence that it is a reliable metric that can be applied when it is not.

Our second major contribution is in Chapter 5. Although made explicitly for text, MAUVE does not consider its sequential nature. However, the quality of generated text can vary wildly over the length of a sequence [7]. For example, it may initially be high quality but then become repetitive and predictable. Therefore, we introduce S-MAUVE. Rather than producing a single score, S-MAUVE produces a sequence of scores, which indicate the quality at all positions in generated sequences.

1.2 Problem

Evaluating the quality of generated data is of massive importance for research in the field. Without the ability to evaluate generative models, there is no way to know which methods work and which ones do not. Therefore, the development of the entire field would stop. This would be unfortunate, as generative modeling enables important tasks that are otherwise difficult, like detecting outliers in datasets [8].

The most reliable method today is the use of humans for evaluation. However, humans are biased, slow, and expensive. Can we produce automated metrics that reflect, or improve upon, the judgment of humans?

1.3 Purpose

This thesis aims to improve on existing methods for evaluating generative models. A robust method for evaluating generative models helps compare different models. There are many competing classes of generative models [9, 10, 11], and for each class, many different improvements have been suggested [12, 13, 14]. A robust way to compare classes of models, as well as improvements to those models, will guide future research by showing researchers what works and what does not.

1.4 Ethics and sustainability

Machine learning models inherit the ethical and social bias of their training data [15]. They are further biased due to their training process. Therefore, using them to judge the quality of generated data must be made carefully. However, humans are also biased, and investigating their biases is just as, if not more, challenging than investigating the bias of a machine learning model. Because of this, the task of creating good measurements for generative models, and being aware of the biases inherent in the process, remains vital.

The importance of machine learning in research continues to increase: Better machine learning enables better research in all data-driven areas, including sustainability. As will be explained in this thesis, generative modeling is of fundamental importance for all machine learning. Therefore, despite the prohibitive computational costs of developing and training them, they are vital to enable further research into environmental matters [16].

1.5 Delimitations

A fundamental assumption in this paper is that, while flawed, human evaluation is the best-existing method to evaluate the quality of a generative model. Therefore, this thesis discusses methods for evaluation in light of how well they correspond to human judgment. In some cases, the original paper on MAUVE [2] performed proper human experiments. In other cases, samples are published in this thesis, and the authors will make their judgments about the quality of some generated data. Any such judgments on the part of the authors will be made clear, and readers are free to disagree.

1.6 Structure of the thesis

The background in Chapter 2 begins with an introduction to the field of generative modeling and why it is important. Later, an introduction to the GPT-2 family of generative language models, as they are used extensively in this thesis. The chapter ends by listing several existing methods for evaluating generative models. By examining these, we both show what goals and priorities researchers have when they create generative models and demonstrate why evaluation is so difficult.

Once the introduction to other methods for evaluating generative methods is complete, MAUVE and its most closely related work are introduced in Chapter 3.

Chapter 4 is the first chapter with new experiments. Here, we complement experiments and discussion made in the original paper on MAUVE [2]. Specifically, we perform two new experiments: First, investigate the usage of MAUVE for comparing different generative models. Second, we apply it to image data and discuss why doing so is desirable.

In the next Chapter 5 we introduce S-MAUVE, which gives more insight into generated sequential data than MAUVE.

Finally, the thesis ends with conclusions, discussion, and suggestions for future work in Chapter 6.

Chapter 2

Background

This section explains generative modeling and why it is an important field. A presentation is then given to the GPT-2 family of generative models, as they are frequently used in this thesis. The chapter ends with an introduction to existing strategies used to evaluate generative models.

2.1 Generative modeling

Generative modeling is concerned with approximating the distribution of some training data. This approximation enables data generation [13], classification [3], and out-of-distribution detection [8]. It is one of the most challenging fields in machine learning, as few other tasks require such intricate knowledge of the training data.

Formally, this approximation is of $P(x)$, $P(x|y)$ or $P(x, y)$, where x is the training data and y the labels. For example, when approximating $P(x)$, x may be an image. If x is an image, then the y in $P(x, y)$ might be a description of what is on the image. Meanwhile, generative text models usually model $P(x|y)$. Here, x is the next word in a sequence, and y all previous words.

Generative models can be adapted to perform many different tasks. For example, a generative model that has approximated $P(x|y)$ or $P(x, y)$ can be turned into a classifier, which models $P(y|x)$. This is done using Baye's rule or with the law of total probability, as in Eq. 2.1. In many cases the normalized probabilities are not needed, and $P(x)$ can therefore be considered constant. Furthermore, $P(y)$ is easy to estimate from training data by counting the frequency of labels.

$$P(y|x) = \frac{P(x,y)}{P(x)} \propto P(x,y) \tag{2.1}$$

$$P(y|x) = \frac{P(x|y) * P(y)}{P(x)} \propto P(x|y) * P(y) \tag{2.2}$$

Figure 2.1 visualizes why this conversion from a generative model to a classifier is possible. In the figure, the approximation $P(x, y)$, represented by the shaded areas, is used to fit a line for classification. The line classifies data points above it as circles and below as x's. The reverse is impossible: The shaded areas can not be produced given only the classification line. In this sense, generative modeling is strictly more complex than classification.

Training a generative model and using it for classification has both benefits and drawbacks over simply creating a classifier from the start. A generative model can detect out-of-distribution samples, warning that potential classifications are unreliable. It can also fill in missing features in the input data. However, generative models require more training data and computational resources to train.

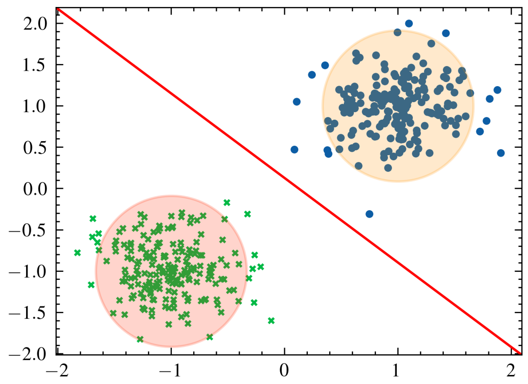

Figure 2.1: A generative model is turned into a discriminative model.

This thesis mainly discusses generative modeling from the perspective of data generation.

2.1.1 Precision and recall

Consider the case of using a generative model to generate new samples. On the one hand, the generated samples should be as varied as the training data;

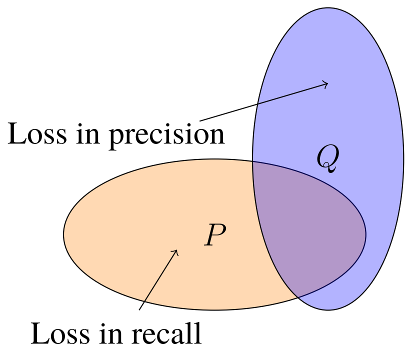

Figure 2.2: Precision and recall between the true distribution P and approximated distribution Q

simply repeating one sample is unacceptable. A model that generates data that is as varied as the training data has high *recall* [2].

On the other hand, the generated samples should look like they came from the training data; samples should be high quality. A model that generates samples of high quality has high *precision* [2].

Formally, if P is a distribution that Q approximates, Q has poor precision if it gives a high likelihood to data points that are unlikely, or impossible, under P. Likewise, Q has poor recall if there are likely samples in P that are highly unlikely, or impossible, under Q. This is demonstrated in Figure 2.1.

The objectives of high precision and high recall are often in opposition, and the tradeoff between them depends on the application [17]. For example, if the task is to generate a few data points, precision becomes more important as the samples should be of high quality. However, if many data points should be generated, recall is more important to avoid repetition.

2.2 The GPT-2 family of generative models

The GPT-2 family of generative language models are referenced frequently in this thesis, and therefore an introduction is given in this section.

In recent years, huge attention-based [18] models such as GPT-2 [19] have entirely taken over the field of text generation. GPT-2 generates answers to questions, translates from one language to another, and continues human-written text. The creators of GPT-2 give this example of the model completing an article, where the beginning is written by a human*:

* https://openai.com/blog/better-language-models/#sample2

> **Human**: A train carriage containing controlled nuclear materials was stolen in Cincinnati today. Its whereabouts are unknown.
>
> **GPT-2**: The incident occurred on the downtown train line, which runs from Covington and Ashland stations. In an email to Ohio news outlets, the U.S. Department of Energy said it is working with the Federal Railroad Administration to find the thief. [...]

Note that there are many valid continuations to the human introduction above. In text generation, there is usually no single, correct answer.

GPT-2 exists in multiple sizes depending on the computational constraints of the application. Reference implementations of GPT-2 begin with the GPT-2 xl model, followed by GPT-2 large, and finally GPT-2 small [19]. Human evaluation has found that the quality of generated samples increase with model size [2]. In other words xl $>$ large $>$ small. Even smaller, unofficial versions such as DistilGPT-2* exist [20]. Humans have not thoroughly evaluated DistilGPT-2. However, since it is smaller than GPT-2 small, the expectation is that it will generate worse samples.

2.3 Text generation with GPT-2

This section gives a high level description of text generation with GPT-2.

GPT-2 does not work on plain text. Instead, any input is converted to a sequence of tokens as part of pre-processing. A token represents a word, a part of a word, or has a special meaning, such as the <end-of-sequence> token. GPT-2 has a *vocabulary*, containing all of its known tokens. It takes as input zero or more tokens and outputs a mapping between all tokens in its vocabulary and how probable they are to appear next [21]. Thanks to this, it can both be used to generate new sequences and to evaluate how probable an entire sequence is.

To generate a new sequence, GPT-2 is applied to a possibly empty sequence of tokens. If the sequence is empty, GPT-2 will generate text from nothing. A non-empty sequence may be the start of a story written by a human or a question that the model should answer. In response to the sequence of tokens, GPT-2 produces the token-probability mapping. Then, a decoding method selects the next token based on the mapping [21]. Selecting the most likely token is called greedy decoding, which is a straightforward but ineffective method. Section 2.3.1 outlines the decoding methods investigated in this paper.

* `https://huggingface.co/distilgpt2`

The old sequence, together with the newly selected token, is then used as a new sequence of tokens for GPT-2. This process repeats until GPT-2 has generated an <end-of-sequence> token or the user-specified maximum number of tokens has been reached.

2.3.1 Decoding methods

A decoding method selects the next token in a sequence when presented with the vocabulary-probability mapping given by the generative model. The choice of decoding method is one of the most important ones when designing a generative language model [7]. There is a tradeoff between generating bland and repetitive text and text where words seem to have been chosen at random. This section outlines the decoding methods mentioned in this paper.

Greedy Decoding

The most straightforward decoding strategy is greedy decoding, which samples the most likely next token given all earlier tokens. As noted in [7], methods like greedy decoding that involves maximizing the likelihood generate text that is repetitive and bland.

Top-k sampling

When generating text, only a subset of all possible words is a reasonable choice at any given point. A language model reflects this by giving a high probability to those words. However, if there are few such words, a significant portion of the probability mass will still be placed at words that are unreasonable choices [21].

Top-k sampling addresses this issue [21], and it is one of the methods used by GPT-2 [19]. Top-k sampling collects the k most probable tokens and only samples from that collection. [21]. Limiting the algorithm to only sample from the top-k words ensures that a reasonably likely word is always selected. Choosing k is not an exact science. A k that is too small will act like greedy decoding and lead to repetitive text, and top-k decoding with $k = 1$ is precisely greedy decoding. Meanwhile, a k that is too large will generate nonsensical text [7].

Temperature sampling

Temperature sampling is motivated similarly as top-k sampling: In a large vocabulary, there will be many low-probability words that should not be sampled. However, unlike top-k sampling, temperature sampling does not make a hard cut-off at the k most likely words. Instead, it skews their distribution towards higher probability words, making low-probability words even more unlikely [7]. A temperature parameter controls how significant the skew is. Similar to top-k sampling, choosing the temperature is challenging. A low skew leads to diverse but possibly nonsensical text, whereas a low temperature leads to repetitive and bland text [7]. Along with top-k sampling, temperature sampling was used in GPT-2 [22].

Top-p sampling

The final decoding strategy is top-p, a.k.a nucleus, sampling. Like top-k sampling, top-p sampling takes a subset of the vocabulary and samples from that subset [7]. However, rather than taking the top-k samples, top-p sampling takes the smallest possible subset of the vocabulary such that the subset makes up p of the total probability mass [7]. In scenarios where only a few tokens make sense, top-p sampling will only sample from these to avoid nonsensical text. However, if the scenario is more open-ended and there are many potential candidates, top-p sampling will collect all of them, which decreases the chances of high repetition.

2.4 Evaluating generative models

When evaluating images or text, the current gold standard is the human judgment: A good model generates data that humans can not distinguish from actual data. However, humans are slow and cumbersome to work with, and there is a need for more automatic methods. Existing automatic methods differ in generality: Some work with any data, whereas others only work with specific data types such as images or text. In addition, some methods require access to the generative model, whereas others only require generated samples.

Methods that work with generated samples often fail to recognize overfitting [17]. An overfit generative model simply repeats training data or repeats it with minor modifications.

2.4.1 Evaluation with log-likelihood

The log-likelihood of test data is a common way to evaluate generative models [17]. Regular likelihood can usually not be used, as it often involves intractable integrals [9, 23] or otherwise hard to compute terms [17].

A generative model is evaluated with log-likelihood by using it to determine how likely unseen test samples are. A good model should award real samples with a high log-likelihood. However, if the sample dimensionality is high, a model that produces good samples can have low log-likelihood or vice-versa, as explained in [17]:

Poor log-likelihood but good samples: A lookup table that stores training data and simply repeats it during generation will "generate" great samples, but will have a poor log-likelihood on unseen test data.

Good log-likelihood but poor samples: Assume p is the density of a good model with respect to likelihood, and q is a model that generates noise. If you generate samples from the model $0.01p(x) + 0.99q(x)$, then samples will be almost complete noise, but the log-likelihood will not change much if the dimensionality of x is large, Eq. 2.3.

$$\tag{2.3}$$

$$\log(a + b) \geq \log(a)$$

$$\log\left[0.01p(x) + 0.99q(x)\right] \geq \log\left[0.01p(x)\right] \tag{2.4}$$

$$\log\left(\tfrac{a}{b}\right) = \log(a) - \log(b)$$

$$\log\left[0.01p(x)\right] = \log p(x) - \log 100 \tag{2.5}$$

$\log p(x)$ is proportional to the dimensionality of x, while $\log 100$ is constant. Modern generative models deal with very high-dimensional data [2], and [17] shows that this issue surfaces even for relatively small 32×32 images. In general, the log-likelihood is a poor measure of the quality of a generative model.

2.4.2 Evaluation based on distances

Many methods to evaluate generative models rely on the use of distances. The hope is that one could measure the distance between some generated data and some test data. If the distance is small enough but not too small, this would indicate an excellent generative model. If the distance is too small, this may indicate overfitting. A fundamental challenge for any method that relies on distance metrics is that basic measures such as euclidean distance are not immediately meaningful [17]. For example, just negating an image leads to a very large distance in "pixel space" between the modified version and the original, even if they remain semantically similar, as seen in Figure 2.3.

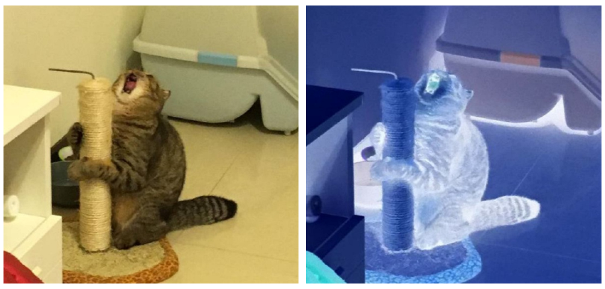
(a) A cat (b) Arguably the same cat

Figure 2.3: These images are far apart in “pixel space”, even if they represent the same cat.

To remedy this, methods such as Fréchet Inception Distance (FID) [24] and Inception Distance (IS) [25] forward samples through a neural network and study their activations rather than the original data. These activations are called *embeddings*. The hope is that forwarding images into this “embedding space” retains semantic information in a way that facilitates measurements and mathematical manipulation. In this space, the images in Figure 2.3 should be close to each other.

2.4.3 Inception score

IS is one of the most used measurements in generative modeling, and was introduced in [25]. It uses the Inception V3 (IV3) network [26] to judge the quality of generated images. Initially, the IV3 network was created to classify images and was trained on ImageNet [27].

IS is calculated by forwarding generated samples through the IV3 network and inspecting the activations in the final layer. The final layer encodes the confidence that an image belongs to a specific class. For example, the final layer can show that an animal on an image has a 20% probability of being a particular breed of cat, 70% probability of a specific dog breed, and 10% probability shared among the many other classes in ImageNet. The authors encode their preferences for generative models as follows:

- A generative model should generate images that result in a conditional label distribution $p(y|x)$ with low entropy: When shown a single image, IV3 should be sure about what is on the image.
- A generative model should generate images from many different classes. Formally, the marginal $\int p(y|x = \mathcal{M}(z))\ dz$, where z is the input to a

generative model $\mathcal{M}$, should have high entropy.

Several flaws with IS are highlighted in [1]: While the classifications of IV3 are robust to training details, i.e., different implementations tend to classify images the same, their confidence in the classifications is very different. The authors demonstrate inception scores that differ 12%, just based on minor implementation details of IV3.

Another highlighted flaw is that the IS is only reasonably applicable to generative models trained on ImageNet. Despite that, many papers apply it to CIFAR-10 [28] or other image datasets. The authors demonstrate some of the shortcomings of IS by generating low-quality images with an incredibly high inception score, as seen in Figure 2.4.

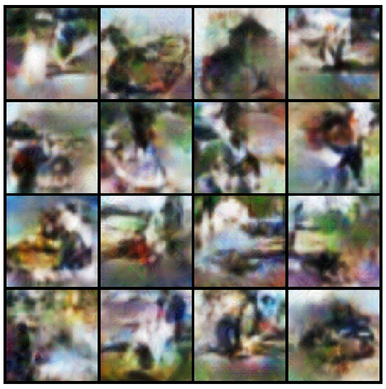

Figure 2.4: From [1, Figure 1]. These samples receive an Inception score of 900.15. Meanwhile, the best-performing networks according to human evaluation receive scores on the order of 10.

2.4.4 Evaluating text

There exist several methods that only work with text. A typical example is BLEU [29], often used in translation tasks where a human translator provides a correct translation. BLEU compares the overlap in n-grams between the generated text and correct text [29]. N-grams are all slices of a text sequence of length n. For example, the 2-grams of the sentence “The quick dog jumps.” are

- The quick
- quick dog
- dog jumps

However, [30] notes that n-grams only look at surface-level similarity and instead propose BERTscore, which uses a BERT network [31] to get embeddings of data samples. Their method compares the meaning of a generated sample with one proper sample. For example, it quantifies the semantic similarity between the sentences *the weather is cold today* and *it is freezing today*.

The authors of MAUVE note that n-grams can not be used to evaluate the quality of text when there is no single correct answer, such as when continuing a story started by a human [2].

Chapter 3

Related work

The primary basis for this thesis is MAUVE, created to evaluate the quality of generated text. MAUVE uses a neural network to get embeddings of samples, similar to IS as described in Section 2.4.3,. However, how it obtains embeddings and what it does with them is more similar to FID. Therefore, this chapter begins with an explanation of FID. After, we finally introduce MAUVE.

3.1 Fréchet Inception Distance

FID was introduced in response to some of the issues identified with IS [24]. Today, it is one of the most common metrics used to evaluate generative image models [32]. FID compares how similar a set of generated samples is to a set of reference samples, where the reference samples are usually real, natural samples. The sets of samples are used to approximate two distributions, which are compared using Fréchet distance.

An intuitive explanation of Fréchet distance is to imagine a human walking their dog [33]: The human walks on one curve and the dog on the other at the same speed. At the start, the human makes the leash as short as possible while still being long enough such that they will not have to extend it as the two are walking. The length of the leash is the Fréchet distance.

The approximations of the distributions are created by forwarding the sets of samples through the IV3 network. The activations in one of the last layers are the sample embeddings used to fit two multivariate normal distributions. The FID is the Fréchet distance between the two distributions [24].

Notably, FID does not account for precision and recall. A good FID might mean high precision, recall, or a combination thereof. The authors of [34] argue that a good score function should consider both. In light of that, they construct a precision-recall curve, which describes whether a generative model is good at precision, recall, or both. Precision-recall curves are one of the fundamental building blocks of MAUVE.

3.2 MAUVE

MAUVE was introduced in [2] and primarily dealt with open-ended text generation. Despite being released recently, it has already seen use in multiple papers [35, 36].

Similar to FID described in Section 3.1, a model is applied to two sets of data to create embeddings. Two distributions are approximated with the embeddings and are then compared in the context of precision and recall. The final score ranges between 0 and 1, where 1 means perfect similarity.

MAUVE makes no attempts to detect overfitting, and a model that simply repeats training data during generation will not be penalized for it.

3.2.1 Approximating P and Q

There is no way to access the actual distributions P and Q. Instead, they are approximated by embedding samples in a language model, like GPT-2.

Each sample consists of a sequence of tokens. As discussed in Section 2.3, GPT-2 predicts the next token based on all previous tokens. Consequently, each token is embedded in the context of all previous tokens. Therefore, the embedding of the last token represents the entire sequence, and this embedding is used by MAUVE.

The embeddings are clustered with K-means [37], and the centroid assignments are counted. The result is two histograms: One created from the embeddings of P_{text}, representing P, and one from Q_{text}, representing Q.

3.2.2 Precision-Recall curves

MAUVE compares the approximations of P and Q from the perspective of precision and recall.

A model may generate realistic but repetitive samples, unrealistic but varied samples, or anything in between. High-quality samples imply high precision, while varied samples imply high recall. For more information on precision and recall, see Section 2.1.1.

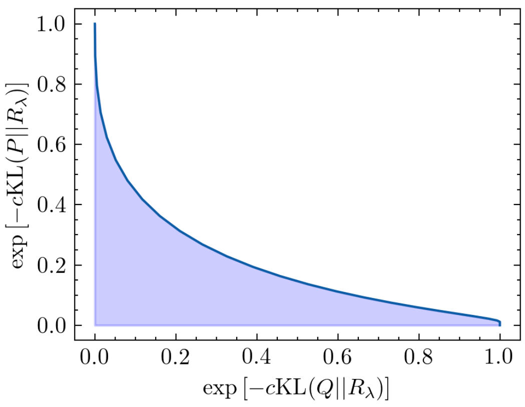

Figure 3.1: MAUVE is the area under the curve. This model leans towards higher recall than precision

Given the target distribution P and approximating distribution Q, the authors express precision as $\mathrm{KL}(P||Q)$ and recall as $\mathrm{KL}(Q||P)$. KL is the Kullback-Leibler divergence, a measure of the difference between two probability distributions. Note that $\mathrm{KL}(P||Q) \neq \mathrm{KL}(Q||P)$.

In many cases, the support for P and Q are not identical, leading to $\mathrm{KL}(P, Q)$ being infinite. The authors overcome this by creating the mixture distribution

$$R_\lambda = \lambda P + (1 - \lambda)Q, \, \lambda \in (0, 1) \tag{3.1}$$

And re-define precision and recall as

$$\text{recall} = \mathrm{KL}(Q||R_\lambda) \tag{3.2}$$

$$\text{precision} = \mathrm{KL}(P||R_\lambda) \tag{3.3}$$

These are then used to create the precision-recall curve

$$\mathcal{C}(P, Q) = \left\{\exp\left(-c\,\mathrm{KL}(Q||R_\lambda)\right), \exp\left(-c\,\mathrm{KL}(P||R_\lambda)\right)\right\} \tag{3.4}$$

where $c > 0$ is a hyperparameter used for scaling. Changing c does not change the ordering of different mauve scores, but changes how close they are. To create interpretable results, the authors suggest using $c = 5$ [2], which is used in this thesis.

The final score is the area under $\mathcal{C}(P, Q)$, found by integrating over λ, as seen in Figure 3.1. The score ranges between 0 and 1, where 1 means perfect similarity and $P = Q$.

Chapter 4

Use cases of MAUVE

This chapter investigates potential applications of MAUVE by extending the experiments done in the original report. The first experiment considers the use of MAUVE for comparing the quality of different generative models. Meanwhile, the second section applies MAUVE to images. The purpose is to demonstrate that MAUVE can be applied sensibly to other data types than text.

4.1 Comparing generative models

One of the main applications of automatic measurements for generative models is for comparison, briefly discussed in Chapter 1. Consequently, MAUVE should recognize quality differences between generative models, and this should be independent of which embedding network it uses. If it is not independent of the embedding network, this raises the question of what embedding network should be used.

If MAUVE is to evaluate data generated from networks A and B, should A or B be used to create embeddings? If A is used, does that give an unfair advantage to A? What if both A and B are used in different trials and come to opposing conclusions?

One could argue that a third model, C, should be used as an impartial judge. However, C is biased just as A and B, though it may be less clear to what extent that affects its qualities as an embedding network.

The original paper briefly investigates how the choice embedding network impacts the MAUVE score [2]. In particular, RoBERTa [38] is used as an alternative embedding network to evaluate text generated by GPT-2 models. RoBERTa specializes in text understanding rather than text generation. Therefore, that experiment corresponds to using a third model C, to compare models A and B. When evaluating data generated by the GPT-2 family of models, they find that there is a high correlation between rankings when using RoBERTa and when using GPT-2 [2].

This experiment investigates the former scenario, where A and B are compared using A and B as embedding networks.

4.1.1 Method

Three pieces for MAUVE are of interest in this section: The embedding model, the reference text P_{text}, and the generated text Q_{text}. Different variations of GPT-2 were used both to generate text and as embedding models for MAUVE..

Models

As described in Section 2.2, language understanding and generational capabilities increase with the size of the model. Therefore, the MAUVE score for text generated by the largest model should be better than for all other models, independent of the embedding network used. The standard models investigated here were, ordered by size, GPT-2 large, GPT-2 small, and DistilGPT-2. In addition, the new model GPT-2 random was created just for this experiment. GPT-2 random is a version of GPT-2 small initialized with random weights and never trained. Samples from it are just random words, and it is the worst of the investigated models.

Text samples

P_{text} was always the test split of the published data from the web-text dataset [19], written by humans.

While officially generated samples are available for some networks and settings of those networks, there are not enough to cover all scenarios investigated in this experiment. Therefore, new samples were generated. This was done with top-p sampling, with $p = 0.92$, and temperature sampling, with $t = 1$. Five human-written words from the webtext dataset were used as initial input in both cases. Official samples were generated using temperature sampling without initial input, also with $t = 1$. The new samples generated for this experiment are available at[*]

[*] https://github.com/samhedin/evaluation-of-gen-models_data

4.1.2 Results

In this section, data generated by a network is denoted as $Q_{\text{text-type-*}}$, where *type* means which type of GPT-2 network was used to generate the data. As described in the previous section, *type* is either large, small, distil or random. The star indicates whether the samples were from official sources or newly generated and what decoding method was used.

Similarly, MAUVE and the embedding network used is denoted by $\text{MAUVE}_{\text{type}}$, such as $\text{MAUVE}_{\text{small}}$ to indicate that MAUVE was used with GPT-2 small as embedding network.

First, GPT-2 large and GPT-2 small were compared with both new and official samples. As hoped, both GPT-2 large and GPT-2 small recognize that GPT-2 large is the more powerful model, Table 4.1.

Table 4.1: GPT-2 small compared with GPT-2 large

	$\text{MAUVE}_{\text{small}}$	$\text{MAUVE}_{\text{large}}$
$Q_{\text{text-small-official-temperature}}$	0.799	0.586
$Q_{\text{text-large-official-temperature}}$	0.891	0.854
$Q_{\text{text-small-temperature}}$	0.789	0.609
$Q_{\text{text-large-temperature}}$	0.889	0.851
$Q_{\text{text-small-top-p}}$	0.850	0.64
$Q_{\text{text-large-top-p}}$	0.958	0.931

However, the situation is different when comparing DistilGPT-2 and GPT-2 large. DistilGPT-2 incorrectly rates itself higher than GPT-2 large, as seen in Table 4.2.

Table 4.2: DistilGPT-2 compared with GPT-2 large.

	$\text{MAUVE}_{\text{distil}}$	$\text{MAUVE}_{\text{large}}$
$Q_{\text{text-distil-top-p}}$	0.324	0.903
$Q_{\text{text-large-top-p}}$	0.252	0.931

The same thing is observed when comparing GPT-2 random with GPT-2 large. GPT-2 random incorrectly believes that it generates better data than GPT-2 large, Table 4.3.

Table 4.3: GPT-2 random compared with GPT-2 large.

	$\text{MAUVE}_{\text{random}}$	$\text{MAUVE}_{\text{large}}$
$Q_{\text{text-random-top-p}}$	0.019	0.26
$Q_{\text{text-large-top-p}}$	0.004	0.931

4.1.3 Discussion

Both DistilGPT-2 and GPT-2 random failed to recognize that their samples are worse than GPT-2 large.

In opposition to the trend, GPT-2 small correctly finds that its samples are worse than those generated by GPT-2 large. This happens both when using official samples and when generating new samples. The reason for this is unclear. However, the networks used, GPT-2 small and GPT-2 large, are closer in quality than the networks used in other trials.

In any case, our results show that MAUVE does not always correspond to human judgment, which favors larger models [2]. Instead, our results show that models are often biased in favor of their own samples when used as embedding networks.

This result complements results in the original paper, where using RoBERTa as an embedding network was shown to correlate with human judgment. Since RoBERTa was not used to generate text, this corresponds to using a third embedding network for evaluation.

It may be argued, then, that a third model, C, should always be used for evaluation. However, no embedding model is guaranteed to detect qualities or defects in generated data. While it might be clear that A is biased in favor of A, C will also be biased, even if in what way might be less obvious. Is it better to use an embedding model with unknown bias than one with known bias?

Furthermore, RoBERTa and GPT-2 were released the same year and are both based on algorithms of that time. Therefore, RoBERTa may be limited in its understanding in the same way that the GPT-2 models are limited. There is currently no reason to believe that RoBERTa would recognize the quality difference between GPT-2 and models that have come after, such as GPT-3 [22] or PaLM [39].

Luckily for MAUVE, humans are also biased and limited: A human may not be able to detect errors in generated data. We believe that if an embedding network has better reading comprehension than whatever human judges are available, it will be a better judge of quality. Even in cases with comparable reading comprehension, it is at least more efficient and standardized.

4.2 MAUVE for images

While the original paper focuses exclusively on text generation, there is nothing fundamental about MAUVE that prevents it from being used for other types of data. Here we show that MAUVE can be used to evaluate images. The only required modification is to use an image model to create embeddings rather than a language model.

4.2.1 Method

The IV3 [26] image classification network was used as embedding network, similar to FID as described in Section 3.1 and IS in Section 2.4.3.

$\text{MAUVE}_{\text{IV3}}$ was first used to compare the similarity of the train and validation sets of Tiny ImageNet [40]. These sets are very similar, and this should be reflected with a high score.

Then, $\text{MAUVE}_{\text{IV3}}$ was used to compare Tiny ImageNet with CIFAR-10 [28]. Since they are different datasets, the distribution of images from them should also differ. In particular, Tiny ImageNet contains 200 different classes, whereas CIFAR-10 only contains 10. Since Tiny ImageNet is more varied, some of its images should be highly unlikely to exist in CIFAR-10. This should be reflected with a relatively low score.

Since the current standard for evaluating generated images is FID, it is used for comparison.

4.2.2 Result

In line with expectations, both FID and $\text{MAUVE}_{\text{IV3}}$ identify that the two subsets of Tiny ImageNet are more similar than Tiny ImageNet and CIFAR-10, see Table 4.4. Note that a lower FID means that the sets are more similar.

Table 4.4: Tiny ImageNet compared with CIFAR-10.

	$\text{MAUVE}_{\text{IV3}}$	FID
TI_{train}, TI_{val}	0.927	62
TI_{train}, CIFAR-10	0.004	309

4.2.3 Discussion

In this experiment, MAUVE was modified to work with image data, and it successfully recognized the difference between Tiny ImageNet and CIFAR-10.

The score between Tiny ImageNet and CIFAR-10 might appear too low. This may be due to the c hyperperamater as explained in Section 3.2.2. The c used here was 5, as suggested by the authors of MAUVE. However, this suggestion is based on the assumption that the application is text evaluation. For images, a different c may be more demonstrative.

Why adapt MAUVE to work with images when FID already exists? Unlike MAUVE, FID does not consider precision and recall, as discussed in Section 3.1. Furthermore, we hope that in the end, there will be only one metric for all types of generative models. Having MAUVE for text, FID for images, and some other metric for sensor data creates unnecessary complexity and confusion.

In particular, researchers that work with data types where no good pre-existing metrics exist often come up with their own. These metrics are often an afterthought, and their strengths and weaknesses are usually not well understood.

MAUVE could become a standardized metric, as we expect its behavior and properties to carry over between data types.

Chapter 5

S-MAUVE

As explained in Section 3.2, MAUVE only saves the embedding of the last token in the sequences that it evaluates. As a result, finding out why a model receives a particular score is difficult. Is a poor score because the model generates bad data overall, or are only tiny subsections poor? Consider the following continuation to user input, generated by GPT-2 small with greedy decoding:

> **Human**: Clinton talks about her time
>
> **GPT-2 small**: as a child star in the '80s, and how she's been able to keep her career going. "I was a child star", she says. "I was a child star. I was a child star. I was a child star. I was a child star. [. . .]"

The first sentence is believable, but the entire text is highly repetitive. Since MAUVE only looks at a summary of the entire sequence, this information is lost. Therefore, we introduce S-MAUVE, which evaluates the quality of text at the token level. The result is a sequence of scores, indicating the quality at all possible positions.

S-MAUVE works with data other than text, as long as the data is sequential. For example, it can be adapted to work with images, similar to MAUVE in Section 4.2. This would be accomplished with image models like PixelCNN [5], which see images as a sequence of pixels, starting at the top left and ending at the bottom right.

Since S-MAUVE is so closely related to MAUVE, we expect that most of the observable behavior of MAUVE to carry over to S-MAUVE, strengths and weaknesses alike.

5.1 Implementation

Only minor modifications to MAUVE are needed to create S-MAUVE.

In MAUVE, each sequence is represented by the embedding of its last token. As a result, the embedding step of MAUVE returns a matrix of shape (n, e), where n is the number of sequences and e the dimensionality of the embedding of the last token.

Meanwhile, S-MAUVE uses the embedding of each token. Therefore, the embedding step returns a matrix of shape (t, n, e), where t denotes the token index and ranges from 0 to the length of the longest sequence. For any sequence s shorter than the longest sequence, s is padded with the embedding of its final token. Note that as a result of this, S-MAUVE at $t = T$ is different from MAUVE unless all sequences are of equal length.

Standard MAUVE subroutines are then applied to the embeddings from each token index, resulting in a sequence of scores. This sequence represents the similarity between evaluated sets of text on a token-by-token basis.

S-MAUVE requires more computational resources than MAUVE, as k-means has to be performed for each token index rather than just once. However, for models such as GPT-2, the computational complexity of embedding a sequence is over quadratic [41]. Therefore, as the length of sequences increases, the embedding step begins to dominate the computational time. Here, S-MAUVE is as fast as MAUVE.

5.2 A sanity check for S-MAUVE

This section demonstrates the basic functionality of S-MAUVE with a toy experiment, where text samples were corrupted with random noise at a fixed position.

P_{text} and Q_{text} were non-overlapping subsets of the webtext dataset of size 2000. As corruption, five random words were inserted 100 words into each of the samples in Q_{text} to create $\mathrm{Q}'_{\text{text}}$. Since one word can result in multiple tokens, the corruption's exact start and end position in the resulting token sequences vary slightly.

MAUVE correctly registers the corruption

$$\mathrm{MAUVE}(\mathrm{P}_{\text{text}}, \mathrm{Q}_{\text{text}}) = 0.958 \tag{5.1}$$

$$\mathrm{MAUVE}(\mathrm{P}_{\text{text}}, \mathrm{Q}'_{\text{text}}) = 0.949 \tag{5.2}$$

However, this decrease in the score is less informative than the result of

S-MAUVE, which is visualized in Figure 5.1.

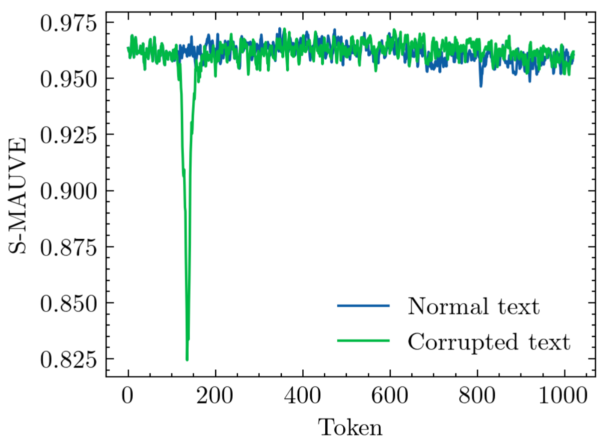

Figure 5.1: S-MAUVE identifies the location of the corruption. 4-step moving average.

S-MAUVE, unlike MAUVE, provides information about where the corruption is located. Until the start of the corruption, the sequences are identical, which is recognized by S-MAUVE. During the corruption, the scores drop significantly. Since corrupted tokens affect the embeddings of subsequent tokens, the two scores do not return to being identical.

The scores may seem close in later stages. However, note that the corruption is relatively minor. Only about 5-10 out of 1000 tokens are noise in each sequence. Therefore, the effects are likely to be averaged out in later stages. Any increase in score thanks to the corruption may be due to chance, and we expect that, on average, the corrupted text should perform worse than the normal text.

The precise information given by S-MAUVE guides developers and researchers. For example, if a developer of a generative model sees Figure 5.1, they know that the main issue with the generated text is not that it becomes repetitive at later stages.

5.3 S-MAUVE for comparing decoding methods

As described in Section 2.3.1, different decoding methods generate sequences of varying quality. In particular, a decoding method may be more or less liable to begin repeating itself or generate nonsensical text. This section applies S-MAUVE to data generated with greedy decoding, top-k sampling and top-p sampling. Greedy decoding is universally regarded as the worst of the three [7, 2]. However, the relation between top-k and top-k sampling is more subtle, and often depends on the application [2].

MAUVE already recognizes the by-humans perceived quality differences between decoding strategies [2], but S-MAUVE is more informative.

5.3.1 Method

S-MAUVE was applied to samples generated by GPT-2 small with greedy decoding, top-k sampling and top-p sampling, which were introduced in Section 2.3.1. top-k sampling was used with $k = 50$, and top-p sampling with $p = 0.92$. The reference text P_{text} was the webtext-test split from [19]. All sets were of size 1000. Temperature sampling was not included in this experiment due to computational constraints.

5.3.2 Result

A visualization of S-MAUVE is available in Figure 5.2. Greedy decoding produces decent text for a short while, but the quality quickly drops as the decoding strategy begins to repeat itself. The sharp increase at $t = 1000$ is expected, as it is always an <end-of-sequence> token. Top-k sampling maintains an overall high score, with minor drops as the sequences get longer. Top-p sampling produces high-quality sequences, and no drop in quality can be seen as the sequences reach 1000 tokens in length.

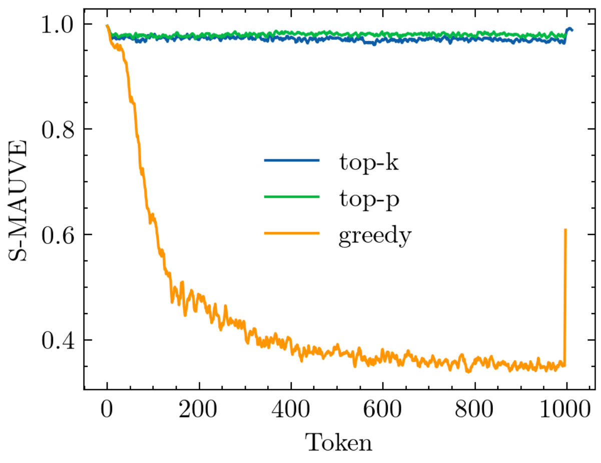

Figure 5.2: S-MAUVE for different decoding methods, 5-step moving average.

5.3.3 Discussion

Greedy decoding is by far the worst of the decoding methods. To a minor extent, the experiment suggests that p-sampling has an advantage over top-k sampling, especially as the sequences grow longer. However, since the performance of these methods is significantly affected by the choice of p and k, it is unwise to draw hard conclusions without further experiments.

None of the results are surprising: Greedy decoding is known to generate bad data, whereas top-k and top-p decoding are closer in quality [2, 7]. S-MAUVE visualizes what was already known in a new way, and we hope that it will be helpful as an alternative to human evaluation when selecting a decoding method. Our implementation of S-MAUVE is available at[*].

[*] https://github.com/samhedin/mauve

Chapter 6

Conclusions and Future work

This chapter begins with a summary of our contributions, along with some closing thoughts. While we have shown that MAUVE is not perfect, it might still be preferable to human evaluation. Later, we list the limitations of the project. Notably, the scale of the experiments was limited. In the very end, we suggest future work for MAUVE.

6.1 Conclusions

MAUVE has quickly become one of the most discussed metrics in generative language modeling. While the original paper only applies MAUVE to measure the quality of open-ended text generation, we have shown that MAUVE also applies to other data types.

As impressive as MAUVE is, we have shown that is limited by the choice of embedding model. Since embedding models are biased and limited, any metric that uses them will also be biased and limited. Being aware of the limitations of the embedding network remains a major challenge in evaluating neural networks. To make matters worse, by the time the limitations of an embedding model are understood, it is likely that a new and improved model will already be available, with its own set of limitations.

However, the current main alternative, human judgment, is also limited. So the question is, can MAUVE be used as an alternative to flawed human judgment? As generative models become better and better, with reading comprehension that matches or exceeds that of humans, the answer either is or is likely to become yes.

We also introduced S-MAUVE, which uses the sequential nature of text to give deeper insight into where a model produces similar or dissimilar tokens to some reference text. In Section 5.3 we showed that S-MAUVE quantifies the difference in quality between decoding methods in a way that regular MAUVE does not. S-MAUVE can be applied to other types of data than text, as long as they are sequential.

6.2 Limitations

Thorough human experiments, in addition to the ones done in MAUVE [2] to judge the quality of generated data is not possible due to monetary constraints.

Computational constraints limit several experiments. For example, GPT-2 xl [19] could not be used in any experiments. In other cases, sample sizes were limited.

Another limitation is the dataset used. The data used in this paper either comes from the webtext dataset [19] used to train GPT-2, or from data generated by GPT-2 itself. The generated data should reflect the webtext dataset. We believe that GPT-2 will react differently to these datasets, which it knows better than unseen datasets. However, running experiments on more datasets is outside the scope of this paper.

6.3 Future work

The choice of embedding network in MAUVE is a vital one, and we would like to see further research into this area. Are there properties of a model that makes it appropriate or inappropriate as an embedding network?

We would also like to see what happens when MAUVE is used as a loss function for training a neural network, similar to how FID was used in [32]. As a piece of advice for implementation, it might be preferable to regard the k-means clustering step as constant.

For S-MAUVE, we would like to see what would happen if it was used as a sampling function. It is possible that it would lead to repetitive text, similar to greedy decoding as described in Section 2.3.1. However, it may still be useful as a way to gain insight into its properties.

References

[1] S. Barratt and R. Sharma, “A note on the inception score,” 2018.

[2] K. Pillutla, S. Swayamdipta, R. Zellers, J. Thickstun, S. Welleck, Y. Choi, and Z. Harchaoui, “MAUVE: Measuring the gap between neural text and human text using divergence frontiers,” in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. [Online]. Available: https://openreview.net/forum?id=Tqx7nJp7PR

[3] R. S. Zimmermann, L. Schott, Y. Song, B. A. Dunn, and D. A. Klindt, “Score-based generative classifiers,” 2021. [Online]. Available: https://arxiv.org/abs/2110.00473

[4] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes *et al.*, “Photorealistic text-to-image diffusion models with deep language understanding,” *arXiv preprint arXiv:2205.11487*, 2022.

[5] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, “Conditional image generation with pixelcnn decoders,” *Advances in neural information processing systems*, vol. 29, 2016.

[6] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, “Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications,” *arXiv preprint arXiv:1701.05517*, 2017.

[7] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, “The curious case of neural text degeneration,” 2019. [Online]. Available: https://arxiv.org/abs/1904.09751

[8] H. Choi, E. Jang, and A. A. Alemi, “Waic, but why? generative ensembles for robust anomaly detection,” *arXiv preprint arXiv:1810.01392*, 2018.

[9] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *stat*, vol. 1050, p. 1, 2014.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.

[11] D. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538.

[12] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1x1 convolutions,” *Advances in neural information processing systems*, vol. 31, 2018.

[13] A. Vahdat and J. Kautz, “Nvae: A deep hierarchical variational autoencoder,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 667–19 679, 2020.

[14] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” 2017. [Online]. Available: https://arxiv.org/abs/1710.10196

[15] H. R. Kirk, F. Volpin, H. Iqbal, E. Benussi, F. Dreyer, A. Shtedritski, Y. Asano *et al.*, “Bias out-of-the-box: An empirical analysis of intersectional occupational biases in popular generative language models,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[16] S. Zhong, K. Zhang, M. Bagheri, J. G. Burken, A. Gu, B. Li, X. Ma, B. L. Marrone, Z. J. Ren, J. Schrier, W. Shi, H. Tan, T. Wang, X. Wang, B. M. Wong, X. Xiao, X. Yu, J.-J. Zhu, and H. Zhang, “Machine learning: New ideas and tools in environmental science and engineering,” *Environmental Science & Technology*, vol. 55, no. 19, pp. 12 741–12 754, 2021. doi: 10.1021/acs.est.1c01339 PMID: 34403250. [Online]. Available: https://doi.org/10.1021/acs.est.1c01339

[17] L. Theis, A. van den Oord, and M. Bethge, “A note on the evaluation of generative models,” 2016.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[20] T. Li, Y. E. Mesbahi, I. Kobyzev, A. Rashid, A. Mahmud, N. Anchuri, H. Hajimolahoseini, Y. Liu, and M. Rezagholizadeh, "A short study on compressing decoder-based language models," *arXiv preprint arXiv:2110.08460*, 2021.

[21] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," 2018. [Online]. Available: https://arxiv.org/abs/1805.04833

[22] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[23] L. Ruthotto and E. Haber, "An introduction to deep generative modeling," 2021. [Online]. Available: https://arxiv.org/abs/2103.05180

[24] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," 2018.

[25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[28] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[29] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002. doi: 10.3115/1073083.1073135 pp. 311–318. [Online]. Available: https://aclanthology.org/P02-1040

[30] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," 2020.

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[32] A. Mathiasen and F. Hvilshøj, "Backpropagating through fréchet inception distance," 2021.

[33] E. W. Chambers, Éric Colin de Verdière, J. Erickson, S. Lazard, F. Lazarus, and S. Thite, "Homotopic fréchet distance between curves or, walking your dog in the woods in polynomial time," *Computational Geometry*, vol. 43, no. 3, pp. 295–311, 2010. doi: https://doi.org/10.1016/j.comgeo.2009.02.008 Special Issue on 24th Annual Symposium on Computational Geometry (SoCG'08). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925772109000637

[34] M. S. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, "Assessing generative models via precision and recall," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[35] Y. Su, T. Lan, Y. Wang, D. Yogatama, L. Kong, and N. Collier, "A contrastive framework for neural text generation," 2022. [Online]. Available: https://arxiv.org/abs/2202.06417

[36] C. Meister, T. Pimentel, G. Wiher, and R. Cotterell, "Typical decoding for natural language generation," *arXiv preprint arXiv:2202.00666*, 2022.

[37] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.

[38] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[39] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," 2022. [Online]. Available: https://arxiv.org/abs/2204.02311

[40] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

TRITA –EECS-EX-2022:407

For DIVA

```
{
"Author1": {
            "Last name": "Yousefzadegan Hedin",
            "First name": "Sam",
            "Local User Id": "u100001",
            "E-mail": "samhedin@kth.se",
            "ORCiD": "0000–0002–00001–1234",
            "organisation": {"L1": "School of Electrical Engineering and Computer Science ",
                             }
           },
"Degree": {"Educational program": "Master’s Programme, Machine Learning, 120 credits"},
"Title": {
            "Main title": "Evaluation of generative machine learning models",
            "Subtitle": "Judging the quality of generated data with the use of neural networks",
            "Language": "eng" },
"Alternative title": {
            "Main title": "Evaluering av generativa maskininlärningsmodeller",
            "Subtitle": "Evaluering av genererad data med hjälp av neurala nätverk",
            "Language": "swe"
},
"Supervisor1": {
                  "Last name": "Ericson",
                  "First name": "Ludvig",
                  "Local User Id": "u100003",
                  "E-mail": "ludv@kth.se",
                  "organisation": {"L1": "School of Electrical Engineering and Computer Science ",
                                   "L2": "Computer Science" }
              },
"Examiner1": {
                  "Last name": "Jensfelt",
                  "First name": "Patric",
                  "Local User Id": "u100004",
                  "E-mail": "patric@kth.se",
                  "organisation": {"L1": "School of Electrical Engineering and Computer Science ",
                                   "L2": "Computer Science" }
              },
"Other information": {
"Year": "2022", "Number of pages": "xiii,39"}
}
```