mathematics

MDPI

Article

A Mathematical Interpretation of Autoregressive Generative Pre-Trained Transformer and Self-Supervised Learning

Minhyeok Lee

School of Electrical and Electronics Engineering, Chung-Ang University, Seoul 06974, Republic of Korea; mlee@cau.ac.kr

Abstract: In this paper, we present a rigorous mathematical examination of generative pre-trained transformer (GPT) models and their autoregressive self-supervised learning mechanisms. We begin by defining natural language space and knowledge space, which are two key concepts for understanding the dimensionality reduction process in GPT-based large language models (LLMs). By exploring projection functions and their inverses, we establish a framework for analyzing the language generation capabilities of these models. We then investigate the GPT representation space, examining its implications for the models' approximation properties. Finally, we discuss the limitations and challenges of GPT models and their learning mechanisms, considering trade-offs between complexity and generalization, as well as the implications of incomplete inverse projection functions. Our findings demonstrate that GPT models possess the capability to encode knowledge into low-dimensional vectors through their autoregressive self-supervised learning mechanism. This comprehensive analysis provides a solid mathematical foundation for future advancements in GPT-based LLMs, promising advancements in natural language processing tasks such as language translation, text summarization, and question answering due to improved understanding and optimization of model training and performance.

Keywords: generative pre-trained transformer; GPT; ChatGPT; self-supervised learning; deep learning; natural language processing; NLP

MSC: 68T27

Citation: Lee, M. A Mathematical Interpretation of Autoregressive Generative Pre-Trained Transformer and Self-Supervised Learning. *Mathematics* **2023**, *11*, 2451. https://doi.org/10.3390/math11112451

Academic Editor: Florentina Hristea

Received: 29 April 2023
Revised: 23 May 2023
Accepted: 23 May 2023
Published: 25 May 2023

1. Introduction

The recent advent of generative pre-trained transformer (GPT) models [1–4], a class of large-scale deep learning models, has led to an increasing focus on their applications in the field of natural language processing (NLP) [5–11] and artificial intelligence (AI) [12–15]. These GPT-based models, including notable instances such as ChatGPT [2], have demonstrated remarkable capabilities in generating human-like text and understanding intricate linguistic patterns. As a result, their potential to transform various domains, ranging from machine translation to question-answering systems, has garnered significant attention from both academia and industry. Despite the widespread interest in GPT-based large language models (LLMs) and their impressive performance [16,17], the mathematical underpinnings of these models and their training methods, specifically autoregressive self-supervised learning, remain relatively unexplored.

This paper aims to bridge the gap between the empirical success of GPT-based LLMs and the theoretical understanding of their fundamental properties. To this end, we delve into the mathematical framework that underlies GPT models and autoregressive self-supervised learning [18–20], seeking a mathematical interpretation of the mechanisms that contribute to their impressive language generation capabilities. By providing a formal analysis of GPT-based LLMs, we hope to lay the groundwork for a more systematic exploration of their properties, limitations, and potential improvements.

In order to establish a comprehensive mathematical foundation, we begin by defining the concept of a natural language space, which encompasses all possible human language expressions. We then introduce the notion of a knowledge space, a lower-dimensional space containing abstract representations of the information conveyed by expressions in the natural language space. By considering projection functions and their inverses, we develop a framework for understanding the process of dimensionality reduction in the context of GPT-based LLMs while preserving the meaning of linguistic expressions.

Furthermore, we examine the representation space of GPT models, which is used to encode input sentences or expressions as high-dimensional vectors. We discuss the smoothness of this space and its implications for the functionality of GPT models as approximations of projection functions. By considering the interplay between the natural language space, the knowledge space, and the GPT representation space, we provide insights into the inner workings of these models and their capacity to capture and manipulate linguistic information.

Additionally, we address the challenges and limitations of GPT models and autoregressive self-supervised learning, focusing on the trade-offs between model complexity and generalization capabilities. We also investigate the implications of incomplete inverse projection functions, which may hinder the ability of GPT-based models to accurately represent certain types of knowledge or meaning.

Given the context, this paper is dedicated to exploring several critical research aspects:

- First, it seeks to formally define the natural language space and the knowledge space to foster an understanding of the dimensionality reduction process in GPT-based LLMs.
- Second, it aims to elucidate the role of projection functions and their inverses in the language generation capabilities of these models.
- Third, it endeavors to identify the properties of the GPT representation space and analyze how these properties affect the models' approximation capabilities.
- Lastly, it contemplates the limitations and challenges of GPT models and autoregressive self-supervised learning, with a special focus on aspects such as model complexity, generalization capabilities, and the completeness of inverse projection functions.

Through this comprehensive analysis, we hope to contribute to the ongoing discourse surrounding GPT-based LLMs and provide a solid mathematical foundation for future advancements in the field.

2. Preliminaries

In this section, we establish a solid foundation on the underlying principles and mechanics of deep learning and GPT models, which is vital for readers who may not be intimately familiar with these concepts. This understanding is indispensable for the more advanced discourse that follows in the subsequent sections, where we delve into the functional intricacies of GPT models in the context of autoregressive self-supervised learning. We introduce formalized definitions and concepts, which are instrumental in these discussions, to provide a rigorous, mathematical depiction of how GPT-based models work. This includes a thorough understanding of how GPT models, comprised of stacked non-linear neural network layers with attention mechanisms, map the natural language space into the knowledge space via the intermediate layers, thus enabling these models to effectively capture and represent the semantic essence of natural language.

2.1. Deep Learning Models

Deep learning models are a class of machine learning models that consist of multiple layers of interconnected artificial neurons [21]. These models are designed to learn hierarchical representations from input data by minimizing a loss function that quantifies the discrepancy between the model's predictions and the ground truth. A deep learning model $f : \mathcal{X} \to \mathcal{Y}$ is a function that maps input data from a space $\mathcal{X}$ to a target space $\mathcal{Y}$. The model is composed of L layers, and each layer $l \in \{1, \ldots, L\}$ consists of a set of neurons $N^l = \{n_1^l, \ldots, n_{k_l}^l\}$, where k_l is the number of neurons in layer l.

Definition 1 (Activation function [21]). *An activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is a non-linear function applied element-wise to the output of a neuron. It is assumed to be differentiable and monotonic.*

For each neuron $n_i^l \in N^l$, let a_i^l denote its pre-activation value and z_i^l denote its post-activation value. The pre-activation value is a linear combination of the outputs from the neurons in the previous layer, and the post-activation value is obtained by applying the activation function to the pre-activation value:

$$a_i^l = \sum_{j=1}^{k_{l-1}} w_{ij}^l z_j^{l-1} + b_i^l, \tag{1}$$

$$z_i^l = \sigma(a_i^l), \tag{2}$$

where w_{ij}^l is the weight connecting neuron n_j^{l-1} to neuron n_i^l, and b_i^l is the bias term for neuron n_i^l. In this context, $\sigma(\cdot)$ represents the activation function as defined in Definition 1.

A deep learning model can be viewed as a composition of functions $f = f^L \circ \cdots \circ f^1$, where each function $f^l : \mathbb{R}^{k_{l-1}} \to \mathbb{R}^{k_l}$ represents the operation performed by layer l. The model's parameters, $\Theta = \{W^l, b^l\}_{l=1}^L$, where $W^l \in \mathbb{R}^{k_l \times kl-1}$ is the weight matrix for layer l and $b^l \in \mathbb{R}^{k_l}$ is the bias vector for layer l, are optimized by minimizing a loss function $\mathcal{L}(\Theta; \mathcal{D})$, where $\mathcal{D}$ denotes the training dataset.

Remark 1. *The backpropagation algorithm computes the gradient of the loss function with respect to the model's parameters, $\nabla_\Theta \mathcal{L}(\Theta; \mathcal{D})$, by applying the chain rule of differentiation. This gradient is used to update the model's parameters iteratively via gradient-based optimization methods, such as stochastic gradient descent (SGD) or variants thereof.*

Theorem 1 (Universal Approximation Theorem [22]). *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a continuous and nonconstant activation function. Given any continuous function $g : \mathbb{R}^d \to \mathbb{R}$ defined on a compact set $\mathcal{C} \subseteq \mathbb{R}^d$ and any $\epsilon > 0$, there exists a feedforward neural network with one hidden layer and width k_1, using activation function $\sigma(\cdot)$, such that the neural network can approximate $g(\cdot)$ with an error less than ϵ, i.e.,*

$$\sup_{\mathbf{x} \in \mathcal{K}} |f(\mathbf{x}) - g(\mathbf{x})| < \epsilon. \tag{3}$$

Theorem 1 states that a sufficiently wide neural network with a single hidden layer can approximate any continuous function with arbitrary accuracy. This theorem highlights the expressive power of deep learning models.

2.2. Generative Pre-Trained Transformer

A GPT model is composed of several layers of neural network modules, each consisting of layer normalization, multi-head attention, and dropout functionalities. The model integrates L number of transformer blocks, each of which houses a residual module encapsulating layer normalization, multi-head attention, dropout, and a fully connected layer. This systematic arrangement and interaction of modules and layers contribute to the robust performance of the GPT model. Figure 1 illustrates the architecture of the GPT model, and Definition 2 provides a formal description of the GPT model.

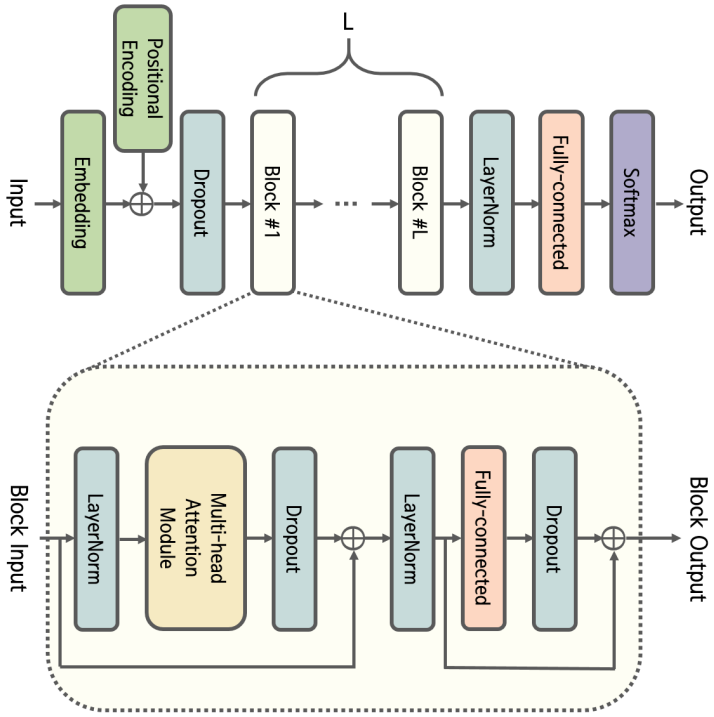

Figure 1. Neural Network Architecture of GPT.

Definition 2 (Generative Pre-trained Transformer (GPT) [4]). *A GPT is an autoregressive self-supervised learning model that employs the Transformer architecture for natural language processing tasks. Given an input sequence of tokens* $\mathbf{x} = (x_1, x_2, \ldots, x_T)$*, where* $x_t \in \mathcal{X}$ *for* $t = 1, 2, \ldots, T$*, the GPT model learns a probability distribution* $P(x_t|x_{<t})$ *over the vocabulary* $\mathcal{X}$*, which is conditioned on the preceding tokens* $x_{<t} = (x_1, x_2, \ldots, x_{t-1})$*. The conditional probability distribution* $P(x_t|x_{<t})$ *esimated by a GPT can be represented by*

$$P(x_t|x_{<t}) = f_\Theta(x_t|x_{<t}), \quad \forall t = 1, 2, \ldots, T. \tag{4}$$

In the GPT model, the position information of tokens in the input sequence is crucial for maintaining the autoregressive property and capturing the sequential dependencies between tokens. To incorporate this information, the GPT model utilizes a technique called positional encoding, which adds a fixed sinusoidal encoding to the input embeddings.

Definition 3 (Positional Encoding [23]). *The positional encoding function* $\mathrm{PE} : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ *computes the position encoding for each position* $p \in \mathbb{N}$ *and each dimension* $i \in \mathbb{N}$ *in the input embedding space as follows [24,25]:*

$$\mathrm{PE}(p, i) = \begin{cases} \sin\left(\frac{p}{10000^{\frac{2i}{d}}}\right) & \text{if } i \text{ is even,} \\ \cos\left(\frac{p}{10000^{\frac{2i-1}{d}}}\right) & \text{if } i \text{ is odd.} \end{cases} \tag{5}$$

The positional encoding function incorporates a 10,000 constant, which serves as a hyperparameter. This value was determined empirically during the original development of the Transformer model. It is designed to create a balance between the higher and lower frequency components of the positional encoding. To facilitate the optimization process, the GPT model employs layer normalization, which is applied to the input of both the multi-head self-attention and position-wise feedforward sublayers. Layer normalization helps alleviate the vanishing and exploding gradient problems and accelerates training.

Definition 4 (Layer Normalization [26]). *Given an input matrix $H \in \mathbb{R}^{T\times d}$, layer normalization computes the normalized output matrix $\hat{H} \in \mathbb{R}^{T\times d}$ as follows:*

$$\hat{H}_{ij} = \frac{H_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}, \tag{6}$$

where $\mu_j = \frac{1}{T}\sum_{i=1}^{T} H_{ij}$ and $\sigma_j^2 = \frac{1}{T}\sum_{i=1}^{T}(H_{ij} - \mu_j)^2$ are the mean and variance of the j-th feature across all positions, respectively, and $\epsilon > 0$ is a small constant for numerical stability.

Definition 5 (Scaled Dot-Product Attention [23]). *The scaled dot-product attention function* Attention $: \mathbb{R}^{T\times d} \times \mathbb{R}^{T\times d} \times \mathbb{R}^{T\times d} \to \mathbb{R}^{T\times d}$ *takes as input a query matrix $Q \in \mathbb{R}^{T\times d}$, a key matrix $K \in \mathbb{R}^{T\times d}$, and a value matrix $V \in \mathbb{R}^{T\times d}$, and it computes the output as follows:*

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V, \tag{7}$$

where $\sqrt{d}$ is a scaling factor that prevents the dot products from becoming too large.

In addition to the scaled dot-product attention mechanism, the GPT model employs the multi-head attention mechanism to capture different aspects of the input sequence. By having multiple attention heads, the model can learn different types of relationships between tokens in parallel.

The GPT model consists of a stack of L identical layers, each containing a multi-head self-attention mechanism, followed by position-wise feedforward networks. The self-attention mechanism computes a weighted sum of the input embeddings, where the weights are determined by the compatibility between input tokens.

Definition 6 (Multi-Head Attention [23]). *The multi-head attention function* MultiHead $: \mathbb{R}^{T\times d} \times \mathbb{R}^{T\times d} \times \mathbb{R}^{T\times d} \to \mathbb{R}^{T\times d}$ *takes as input a query matrix $Q \in \mathbb{R}^{T\times d}$, a key matrix $K \in \mathbb{R}^{T\times d}$, and a value matrix $V \in \mathbb{R}^{T\times d}$, and it computes the output as follows [27,28]:*

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O, \tag{8}$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ *for $i = 1, 2, \ldots, h$, with $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d\times \frac{d}{h}}$ being learnable weight matrices, h being the number of attention heads, and $W^O \in \mathbb{R}^{d\times d}$ being a learnable output weight matrix.*

Beyond the attention mechanism, the GPT model utilizes position-wise feedforward networks (FFNs) within each layer. These FFNs consist of two linear layers with a non-linear activation function in between, which allows the model to learn complex non-linear transformations of the input sequence.

Definition 7 (Position-wise Feedforward Networks [23]). *Given an input matrix $H \in \mathbb{R}^{T\times d}$, the position-wise feedforward network computes the output matrix $H' \in \mathbb{R}^{T\times d}$ as follows:*

$$H' = \sigma_H(HW^1 + b^1)W^2 + b^2, \tag{9}$$

where $W^1 \in \mathbb{R}^{d\times d'}$, $W^2 \in \mathbb{R}^{d'\times d}$ are learnable weight matrices, and $b^1 \in \mathbb{R}^{d'}$, $b^2 \in \mathbb{R}^d$ are learnable bias vectors. The activation function σ_H introduces non-linearity.

The GPT model is trained to maximize the likelihood of the target tokens given the context tokens. The objective function is the negative log-likelihood of the target tokens:

$$\mathcal{L}(\Theta; \mathcal{D}) = -\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \sum_{t=1}^{T} \log P(y_t | x_{<t}; \Theta), \tag{10}$$

where $\mathcal{D}$ is the dataset of input-target pairs $(\mathbf{x}, \mathbf{y})$ with $\mathbf{y} = (y_1, y_2, \ldots, y_T)$ being the target sequence.

A multitude of research endeavors have been undertaken to investigate the architecture of GPT models and their practical applications. However, the majority of these studies' pursuits emphasize the exploration of architectural alterations and the development of fine-tuning methodologies that enhance empirical results [1,3,4,29].

For instance, Radford et al. [4] revealed that language models, when subjected to training on a novel dataset consisting of millions of webpages, exhibit an inherent ability to learn specific tasks without the need for any explicit supervision. They identified the significance of the language model's capacity in the success of zero-shot task transfer, with performance improving in a log-linear fashion across tasks. Their most extensive model, GPT-2, showcased an unparalleled performance on a majority of tested language modeling datasets, underscoring the potential of language processing systems that learn to perform tasks via naturally occurring demonstrations.

Similarly, Brown et al. [3] discovered that scaling up language models substantially enhances task-agnostic, few-shot, and GPT-3. They trained an autoregressive language model with a large parameter set, which was significantly larger than any previous non-sparse language model, and tested its performance in the few-shot setting. Despite acknowledging certain areas where GPT-3's few-shot learning struggles, their findings attest to the remarkable performance of GPT-3 across a range of NLP tasks.

Kaplan et al. [29] investigated the empirical scaling laws pertinent to language model performance on the cross-entropy loss, observing that the loss scales as a power-law with model, dataset size, and the amount of computation used in the training process. They noted the minimal effects of other features such as network width, despite the fact that larger models are significantly more sample-efficient.

However, notwithstanding the compelling empirical results of these studies, there exists a conspicuous gap in our understanding of the model's behavior from a mathematical perspective. This gap is exactly what our research aspires to fill. We aim to provide a functional analysis of GPT, contributing to a deeper comprehension of the model's properties and elucidating the mechanisms that drive its performance.

2.3. Autoregressive Self-Supervised Learning

Definition 8 (Autoregressive Self-Supervised Learning [4])**.** *Autoregressive self-supervised learning is a learning paradigm in which a model is trained to predict the next token in a sequence, given the preceding tokens, without using any labeled data [19]. The model learns a probability distribution $P(x_t|x_{<t}; \Theta)$ over the vocabulary $\mathcal{X}$, conditioned on the context tokens $x_{<t} = (x_1, x_2, \ldots, x_{t-1})$, where Θ denotes the model's parameters.*

Definition 9 (Token Probability Estimation [4])**.** *In autoregressive self-supervised learning, token probability estimation refers to the process of computing the probability of a token $x_t \in \mathcal{X}$ given the context tokens $x_{<t}$. Given a model with parameters Θ, the token probability estimation can be defined as:*

$$\hat{P}(x_t|x_{<t}; \Theta) = f_\Theta(x_{<t}), \tag{11}$$

where f_Θ is a function parameterized by Θ that maps the context tokens $x_{<t}$ to the estimated probability distribution over the vocabulary $\mathcal{X}$.

In the context of autoregressive self-supervised learning, the function f_Θ in Definition 9 is often implemented using deep neural networks, such as transformers or recurrent neural networks (RNNs). These architectures are designed to capture complex dependencies between tokens and are capable of representing a wide range of probability distributions over the vocabulary $\mathcal{X}$. In particular, the choice of the function f_Θ and its parameterization can have a significant impact on the model's ability to learn the true underlying data-generating process.

Assumption 1 (Markov Assumption)**.** *For autoregressive self-supervised learning, it is assumed that the conditional probability of a token x_t depends only on a fixed number of preceding tokens $x_{<t}$. This is also known as the Markov assumption, which simplifies the modeling of the joint probability distribution over token sequences.*

Assumption 2 (Smoothness)**.** *For autoregressive self-supervised learning, we assume that the function f_Θ in Definition 9 is smooth with respect to the model parameters Θ. This implies that small changes in the model parameters will result in small changes in the estimated probability distribution over the vocabulary $\mathcal{X}$.*

A key implication of Assumption 2 is that the optimization landscape of the autoregressive self-supervised learning problem is characterized by a continuous and differentiable space with respect to the model parameters Θ. This property allows the use of gradient-based optimization techniques, such as Adam and RMSProp, to iteratively update the model parameters and minimize the objective function in Equation (15). Furthermore, under appropriate conditions, convergence to a local minimum or stationary point can be guaranteed.

Definition 10 (Token Context Window [4])**.** *A token context window of size $w \in \mathbb{N}$ is a fixed-size sliding window that captures the w most recent tokens in a sequence. Given a sequence $\mathbf{x} = (x_1, x_2, \ldots, x_T)$, the token context window $C_t^w(\mathbf{x})$ at position $t \in 1, \ldots, T$ is defined as $C_t^w(\mathbf{x}) = (x_{t-w+1}, x_{t-w+2}, \ldots, x_t)$, where $1 \leq t - w + 1$.*

Remark 2. *In practice, the token context window size w is often chosen to balance the trade-off between computational complexity and the capacity to capture long-range dependencies in the input sequences.*

It is important to note that the choice of the token context window size w in Definition 10 can be influenced by the inherent structure and dependencies in the data. A larger context window can potentially capture longer-range dependencies, but it may also increase the computational complexity of the model and lead to overfitting. Conversely, a smaller context window reduces computational complexity but may fail to capture important dependencies between tokens. In practice, the optimal context window size is often determined using model selection techniques, such as cross-validation.

Definition 11 (Token Autocorrelation)**.** *Token autocorrelation is a measure of the dependency between tokens at different positions in a sequence. Given a sequence $\mathbf{x} = (x_1, x_2, \ldots, x_T)$, the token autocorrelation at lag $k \in 1, 2, \ldots, T-1$ is defined as:*

$$\rho_k(\mathbf{x}) = \frac{\sum_{t=k+1}^{T}(x_t - \bar{x})(x_{t-k} - \bar{x})}{\sum_{t=1}^{T}(x_t - \bar{x})^2}, \tag{12}$$

where $\bar{x}$ denotes the mean of the sequence $\mathbf{x}$.

Remark 3. *The token autocorrelation can be used to analyze the statistical dependencies between tokens in a sequence, which can inform the choice of the token context window size w in Definition 10.*

Token autocorrelation, as defined in Definition 11, can also provide insights into the appropriate choice of the function f_Θ in Definition 9. For example, if the token autocorrelation decays rapidly with increasing lag, it may indicate that a simpler model, such as an RNN with a small hidden state, could be sufficient to capture the dependencies between tokens. On the other hand, if the token autocorrelation decays slowly or exhibits periodic patterns, more complex models, such as transformers with multiple layers and attention mechanisms, might be necessary to accurately represent the underlying data-generating process.

Definition 12 (Conditional Entropy [30]). *The conditional entropy $H(X_t|X_{<t})$ is a measure of the uncertainty in a random variable X_t given the values of $X_{<t}$. The conditional entropy is defined as*

$$H(X_t|X_{<t}) = -\sum_{x_t \in \mathcal{X}} \sum_{x_{<t} \in \mathcal{X}^{t-1}} P(x_t, x_{<t}) \log P(x_t|x_{<t}), \tag{13}$$

where $P(x_t, x_{<t})$ denotes the joint probability of observing the sequence $(x_{<t}, x_t)$, and $P(x_t|x_{<t})$ represents the conditional probability of x_t given $x_{<t}$.

Definition 13 (Perplexity [31]). *Perplexity is a measure of the average uncertainty in predicting the next token in a sequence given the context tokens. The perplexity of a probability distribution $P(x_t|x_{<t}; \Theta)$, given the context tokens $x_{<t}$, is defined as the exponential of the conditional entropy:*

$$\text{Perplexity}(P(x_t|x_{<t}; \Theta)) = \exp(H(X_t|X_{<t})), \tag{14}$$

where $H(X_t|X_{<t})$ denotes the conditional entropy as defined in Definition 12.

Remark 4. *Lower perplexity values indicate a better model fit, as the model assigns higher probabilities to the observed sequences. Perplexity is often used as a performance metric for autoregressive self-supervised learning models.*

Given Definition 8, Assumption 1, and Definition 10, the objective function for training an autoregressive self-supervised learning model can be formulated as follows:

$$\mathcal{L}(\Theta; \mathcal{D}) = -\sum_{\mathbf{x} \in \mathcal{D}} \sum_{t=1}^{T} \log P(x_t|C_t^w(\mathbf{x}); \Theta), \tag{15}$$

where $\mathcal{D}$ is the dataset of input sequences $\mathbf{x}$ and Θ denotes the model's parameters.

Proposition 1. *Under the Markov assumption in Assumption 1 and given a token context window of size w as in Definition 10, the autoregressive self-supervised learning objective in Equation (15) converges to the true conditional probability distribution of the underlying data-generating process as the size of the dataset $\mathcal{D}$ goes to infinity, provided that the model has sufficient capacity and appropriate optimization techniques are employed.*

Proposition 2. *The conditional entropy $H(X_t|X_{<t})$ of the true data-generating process is upper-bounded by the logarithm of the size of the vocabulary $\mathcal{X}$, i.e., $H(X_t|X_{<t}) \leq \log |\mathcal{X}|$.*

Proof. The conditional entropy $H(X_t|X_{<t})$ is a function of the joint probability distribution $P(x_t, x_{<t})$ and the conditional probability distribution $P(x_t|x_{<t})$. By definition, $0 \leq P(x_t|x_{<t}) \leq 1$, and the maximum value of the conditional entropy occurs when $P(x_t|x_{<t}) = \frac{1}{|\mathcal{X}|}$ for all $x_t \in \mathcal{X}$. In this case, the conditional entropy becomes $H(X_t|X_{<t}) = \log |\mathcal{X}|$. Therefore, $H(X_t|X_{<t}) \leq \log |\mathcal{X}|$. □

It is worth noting that the bound on the conditional entropy in Proposition 2 has important implications for the optimization of autoregressive self-supervised learning models. Since the conditional entropy is upper-bounded by the logarithm of the size of the vocabulary $\mathcal{X}$, it follows that the perplexity, as defined in Definition 13, is also upper-bounded by $|\mathcal{X}|$. This provides an absolute reference point for comparing the performance of different models as well as a theoretical limit on the achievable perplexity. In practice, however, the true conditional entropy of the data-generating process may be much lower than the bound, and the choice of an appropriate model and optimization technique can lead to significant improvements in perplexity over a naïve uniform distribution over the vocabulary.

Definition 14 (GPT Autoregressive Self-Supervised Learning [4]). *Let $\mathcal{G}$ be a GPT model as defined in Definition 2. The autoregressive self-supervised learning of $\mathcal{G}$ is the process of training*

the model by optimizing the objective function in Equation (15), where Θ represents the parameters of $\mathcal{G}$ and $\mathcal{D}$ is the dataset of input sequences.

Assumption 3 (Stationary Data-generating Process)**.** *We assume that the dataset $\mathcal{D}$ is generated by a stationary data-generating process; i.e., the joint probability distribution of the tokens in the sequences does not change over time.*

Remark 5. *Assumption 3 simplifies the analysis of the convergence properties of autoregressive self-supervised learning. In practice, the data-generating process may be non-stationary, and the model may need to adapt to the changing distribution over time.*

Proposition 3. *Under the Markov assumption in Assumption 1 and given a token context window of size w as in Definition 10, the GPT autoregressive self-supervised learning process in Definition 14 converges to the true conditional probability distribution of the underlying data-generating process as the size of the dataset $\mathcal{D}$ goes to infinity, provided that the model has sufficient capacity and appropriate optimization techniques are employed.*

Proof. Given Definition 8 and Definition 14, the objective function for training a GPT model using autoregressive self-supervised learning can be formulated as in Equation (15). The proof of Proposition 3 follows directly from Proposition 1. □

Example 1 (GPT and Autoregressive Self-Supervised Learning)**.** *Let $\mathcal{G}$ be a GPT model and $\mathcal{D}$ be a dataset of text sequences. We can train $\mathcal{G}$ using autoregressive self-supervised learning, as described in Definition 14. By optimizing the objective function in Equation (15), we can learn a conditional probability distribution over the vocabulary $\mathcal{X}$ that captures the statistical dependencies between tokens in the input sequences. Consider a GPT model, $\mathcal{G}$, trained on the dataset $\mathcal{D}$ that contains English sentences. When the model encounters the sentence "The quick brown fox jumps over the lazy dog", it predicts the next word in the sentence given the previous words. For example, after processing "The quick brown fox jumps", the model predicts "over" as the most likely next word. This demonstrates the use of autoregressive self-supervised learning, where the model learns the conditional probability distribution over the vocabulary that captures the statistical dependencies between words in a sentence.*

3. Natural Language Space

Definition 15 (Natural Language Space)**.** *The natural language space L is a high-dimensional space that contains all possible human language sentences or expressions, where each point in the space corresponds to a unique sentence or expression. Each point can be represented as a vector in a high-dimensional space with d_{L} being the dimensionality of the natural language space.*

Definition 16 (Vector Representation of Sentences)**.** *A vector representation function $f : \mathrm{L} \to \mathbb{R}^{d_{\mathrm{L}}}$ maps sentences or expressions in the natural language space L to points in a high-dimensional space, typically $\mathbb{R}^{d_{\mathrm{L}}}$, where each point is represented as a unique vector. The function f ensures that each sentence or expression $s \in \mathrm{L}$ has a corresponding vector $v_s \in \mathbb{R}^{d_{\mathrm{L}}}$.*

Definition 17 (Knowledge Space)**.** *The knowledge space $\mathcal{K}$ is a lower-dimensional space that contains abstract representations of the information or meaning conveyed by sentences or expressions in the natural language space L. Each point in the knowledge space can be represented as a vector in a relatively lower-dimensional space with $\mathcal{D}_{\mathcal{K}} < \mathcal{D}_{\mathrm{L}}$ being the dimensionality of the knowledge space.*

Assumption 4 (Smoothness of Natural Language Space)**.** *We assume that the natural language space L is smooth, meaning that small changes in the coordinates of a point in the space correspond to small changes in the meaning or information content of the corresponding sentence or expression.*

Assumption 5 (Smoothness of Knowledge Space)**.** *We assume that the knowledge space $\mathcal{K}$ is smooth, meaning that small changes in the coordinates of a point in the space correspond to small changes in the underlying information or meaning represented by the point.*

Assumption 6 (Locality of Projection Function)**.** *We assume that the projection function $p : \mathrm{L} \to \mathcal{K}$ is local, meaning that if two sentences or expressions $s_1, s_2 \in \mathrm{L}$ are similar in meaning or information content, then their projections $p(s_1), p(s_2) \in \mathcal{K}$ are also similar in their abstract representations of information or meaning.*

Definition 18 (Projection Function)**.** *A projection function $p : \mathrm{L} \to \mathcal{K}$ maps points in the natural language space* L *to points in the knowledge space $\mathcal{K}$, such that the information or meaning conveyed by a sentence or expression in* L *is preserved as an abstract representation in $\mathcal{K}$.*

Lemma 1 (Existence of a Projection Function)**.** *There exists a projection function $p : \mathrm{L} \to \mathcal{K}$ that maps points in the natural language space* L *to points in the knowledge space $\mathcal{K}$, preserving the information or meaning conveyed by the corresponding sentence or expression.*

Proof. Under Assumptions 4 and 5, both the natural language space L and the knowledge space $\mathcal{K}$ are smooth. Thus, it is possible to define a continuous mapping between these spaces. Furthermore, since the dimensionality of the knowledge space is lower than that of the natural language space, there exists a projection function $p : \mathrm{L} \to \mathcal{K}$ that maps points in L to points in $\mathcal{K}$ while preserving the information or meaning conveyed by the corresponding sentence or expression. □

This lemma establishes the foundational existence of the projection function, facilitating Lemma 2, which discusses the composition of this projection function and its inverse.

Example 2 (Projection of Natural Language Space to Knowledge Space)**.** *Consider a sentence in the natural language space* L*. Let $s \in \mathrm{L}$ represent a specific sentence or expression, with vector representation $v_s \in \mathbb{R}^{d_\mathrm{L}}$. We can define a projection function $p : \mathrm{L} \to \mathcal{K}$ that maps the sentence s to a point $k \in \mathcal{K}$, with vector representation $v_k \in \mathbb{R}^{d_\mathcal{K}}$, such that $v_k = p(v_s)$. Consider the sentence "London is the capital of England" in the natural language space* L*. This sentence or expression $s \in \mathrm{L}$ can be represented as a vector $v_s \in \mathbb{R}^{d_\mathrm{L}}$. Using a projection function $p : \mathrm{L} \to \mathcal{K}$, we map this sentence to a point $k \in \mathcal{K}$ in the knowledge space, which is, indeed, challenging to illustrate, but it might be a set of relationships of ("London", "is capital of", "England"). This illustrates how a projection function can convert a sentence from the natural language space to a more formal representation in the knowledge space.*

Definition 19 (Similarity Metric [32])**.** *A similarity metric $d : \mathcal{K} \times \mathcal{K} \to \mathbb{R}_{\geq 0}$ is a function that measures the similarity between two points in the knowledge space $\mathcal{K}$. It satisfies the following properties for all $k_1, k_2, k_3 \in \mathcal{K}$:*

1. *$d(k_1, k_2) \geq 0$ (non-negativity);*
2. *$d(k_1, k_2) = 0$ if and only if $k_1 = k_2$ (identity of indiscernibles);*
3. *$d(k_1, k_2) = d(k_2, k_1)$ (symmetry); and*
4. *$d(k_1, k_3) \leq d(k_1, k_2) + d(k_2, k_3)$ (triangle inequality).*

Remark 6. *Under the smoothness Assumptions 4 and 5, the projection function p preserves the similarity between sentences or expressions in* L *as measured by the similarity metric d in $\mathcal{K}$. In other words, if $s_1, s_2 \in \mathrm{L}$ are similar in meaning or information content, then $d(p(s_1), p(s_2))$ will be small.*

Definition 20 (Inverse Projection Function)**.** *An inverse projection function $p^{-1} : \mathcal{K} \to \mathrm{L}$ maps points in the knowledge space $\mathcal{K}$ back to points in the natural language space* L*, such that the information or meaning represented by a point in $\mathcal{K}$ is transformed into a corresponding sentence or expression in* L*.*

Assumption 7 (Existence of an Inverse Projection Function)**.** *We assume that there exists an inverse projection function* $p^{-1} : \mathcal{K} \to \mathrm{L}$ *that maps points in the knowledge space* $\mathcal{K}$ *back to points in the natural language space* L*, transforming the information or meaning represented by a point in* $\mathcal{K}$ *into a corresponding sentence or expression in* L*.*

Remark 7. *A human can be considered as an example of an inverse projection function* $p^{-1} : \mathcal{K} \to \mathrm{L}$*. When a person is given an abstract representation of information or meaning from the knowledge space* $\mathcal{K}$*, they can generate a corresponding sentence or expression in the natural language space* L*. This process involves the cognitive ability to understand the meaning or information represented by a point in* $\mathcal{K}$ *and to transform it into a coherent and comprehensible sentence or expression in* L*. Thus, the human ability to communicate and express information can be viewed as an instantiation of the inverse projection function* p^{-1}*.*

Lemma 2 (Projection-Inverse Projection Composition)**.** *Given a projection function* $p : \mathrm{L} \to \mathcal{K}$ *and an inverse projection function* $p^{-1} : \mathcal{K} \to \mathrm{L}$*, the composition of these functions, denoted by* $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$*, is an approximate identity mapping on the natural language space* L*.*

Proof. Let $s \in \mathrm{L}$ be an arbitrary sentence or expression, and let $k = p(s) \in \mathcal{K}$ be the projection of s into the knowledge space. Since $p^{-1} : \mathcal{K} \to \mathrm{L}$ is an inverse projection function, it maps k back to a sentence or expression $s' = p^{-1}(k) \in \mathrm{L}$.

Now, consider the composition of the projection and inverse projection functions: $p^{-1} \circ p(s) = p^{-1}(k) = s'$. By Assumption 7, the information or meaning represented by k is transformed into the corresponding sentence or expression s'. Under Assumptions 4 and 5, we can deduce that s' is similar in meaning or information content to the original sentence s.

While s' might not be identical to s, their similarity in meaning or information content implies that the composition $p^{-1} \circ p$ is an approximate identity mapping on the natural language space L. This holds for any arbitrary $s \in \mathrm{L}$. □

This lemma builds upon Lemma 1 and forms the underpinning of Theorem 2, which uses this approximate identity mapping for dimensionality reduction.

Theorem 2 (Dimensionality Reduction of Natural Language Space)**.** *Given a natural language space* L *with dimensionality* $\mathcal{D}_{\mathrm{L}}$ *and a knowledge space* $\mathcal{K}$ *with dimensionality* $\mathcal{D}_{\mathcal{K}}$*, where* $\mathcal{D}_{\mathcal{K}} < \mathcal{D}_{\mathrm{L}}$*, there exists a projection function* $p : \mathrm{L} \to \mathcal{K}$ *and an inverse projection function* $p^{-1} : \mathcal{K} \to \mathrm{L}$ *that allows for dimensionality reduction while preserving the information or meaning conveyed by sentences or expressions in* L*.*

Proof. By Lemma 1, we have the existence of a projection function $p : \mathrm{L} \to \mathcal{K}$ that maps points in the natural language space L to points in the knowledge space $\mathcal{K}$, preserving the information or meaning conveyed by the corresponding sentence or expression.

By Assumption 7, there exists an inverse projection function $p^{-1} : \mathcal{K} \to \mathrm{L}$ that maps points in the knowledge space $\mathcal{K}$ back to points in the natural language space L, transforming the information or meaning represented by a point in $\mathcal{K}$ into a corresponding sentence or expression in L.

As shown in the proof of Lemma 2, the composition $p^{-1} \circ p$ is an approximate identity mapping on the natural language space L. Therefore, the information or meaning conveyed by sentences or expressions in L is approximately preserved through the projection to $\mathcal{K}$ and the inverse projection back to L. □

We now introduce a new assumption concerning the preservation of information or meaning when points are mapped between the natural language space and the knowledge space.

Assumption 8 (Preservation of Information)**.** *We assume that for any sentence or expression $s \in$ L and its projection $k \in \mathcal{K}$, the inverse projection function p^{-1} preserves the information or meaning conveyed by s such that $p^{-1}(k) \approx s$.*

This assumption implies that the projection function p and the inverse projection function p^{-1} can be used to perform dimensionality reduction on the natural language space L while preserving the information or meaning conveyed by sentences or expressions in L.

We proceed to show that the composition of the projection function p and the inverse projection function p^{-1} can be used to approximately recover the original sentence or expression in L.

Proposition 4 (Approximate Recovery)**.** *Given a sentence or expression $s \in$ L, its projection $k \in \mathcal{K}$, and the inverse projection $p^{-1}(k)$, under Assumption 8, the composition $p^{-1} \circ p(s) \approx s$.*

Proof. By Assumption 8, the inverse projection function p^{-1} preserves the information or meaning conveyed by s such that $p^{-1}(k) \approx s$. Therefore, the composition $p^{-1} \circ p(s) = p^{-1}(p(s)) = p^{-1}(k) \approx s$. □

The above proposition demonstrates that the dimensionality reduction can be achieved while approximately preserving the information or meaning conveyed by sentences or expressions in the natural language space L. This result justifies the use of the projection function p and the inverse projection function p^{-1} for dimensionality reduction in the natural language space, allowing for a compact representation of human language in a lower-dimensional knowledge space $\mathcal{K}$.

4. Representation Space of GPT

Definition 21 (Representation Space of GPT)**.** *The representation space of a GPT model, denoted by $\mathcal{R}$, is a high-dimensional space that contains vector representations of tokens or sequences typically obtained from an intermediate layer of the GPT model, such as the decoder output. Each point in the representation space can be represented as a vector in a high-dimensional space, typically $\mathbb{R}^{d_\mathcal{R}}$, with $d_\mathcal{R}$ being the dimensionality of the representation space.*

In the representation space of a GPT model, tokens and sequences from the natural language space L are mapped to high-dimensional vectors. This mapping allows for the manipulation and processing of language data in a continuous and differentiable space, which is particularly useful for training deep learning models. Importantly, the structure of the representation space should ideally capture the semantic and syntactic properties of the language so that similar meanings or structures are represented by nearby points in the space.

Definition 22 (GPT Vector Representation Function)**.** *A GPT vector representation function $h : \text{L} \to \mathcal{R}$ maps sentences or expressions in the natural language space* L *to points in the representation space $\mathcal{R}$, where each point is represented as a unique vector. The function h ensures that each sentence or expression $s \in$ L has a corresponding vector $v_s \in \mathbb{R}^{d_\mathcal{R}}$ in the representation space $\mathcal{R}$.*

The GPT vector representation function h should be designed to preserve the semantic and syntactic properties of the language in the representation space. This often involves the use of continuous embeddings, which can be learned during the training process. As the GPT model is trained to optimize the autoregressive self-supervised learning objective, the model implicitly learns a mapping that captures the structure and relations between tokens and sequences in the natural language space. The resulting representation space should thus enable the GPT model to reason about and generate natural language text based on the learned representations.

Assumption 9 (Smoothness of Representation Space)**.** *We assume that the representation space $\mathcal{R}$ is smooth, meaning that small changes in the coordinates of a point in the space correspond to small changes in the information or meaning conveyed by the corresponding token or sequence in the GPT model.*

The smoothness assumption on the representation space implies that the GPT model can generalize well to unseen examples, as the learned representations of similar tokens and sequences are expected to be close in the representation space. This smoothness property is crucial for the GPT model to perform well on a wide range of natural language understanding and generation tasks, as it allows the model to exploit the structure of the representation space and make meaningful predictions even for previously unseen combinations of tokens and sequences.

Definition 23 (GPT Inverse Projection Function)**.** *A GPT inverse projection function $g : \mathcal{K} \to \mathcal{R}$ maps points in the knowledge space $\mathcal{K}$ to points in the representation space $\mathcal{R}$, such that the information or meaning represented by a point in $\mathcal{K}$ is transformed into a corresponding vector representation in $\mathcal{R}$, which can be decoded into a sentence or expression in the natural language space* L *by the GPT model.*

The GPT inverse projection function g serves as a bridge between the knowledge space $\mathcal{K}$ and the representation space $\mathcal{R}$. By mapping points in the knowledge space to corresponding vector representations in the representation space, the GPT model can leverage its learned representations to reason about and manipulate information in the knowledge space. The existence of such an inverse projection function is essential for establishing a connection between the GPT model's representation space and the underlying knowledge space, allowing the model to effectively access and generate knowledge in the form of natural language text.

Assumption 10 (Existence of a GPT Inverse Projection Function)**.** *We assume that there exists a GPT inverse projection function $g : \mathcal{K} \to \mathcal{R}$ that maps points in the knowledge space $\mathcal{K}$ to points in the representation space $\mathcal{R}$, transforming the information or meaning represented by a point in $\mathcal{K}$ into a corresponding vector representation in $\mathcal{R}$, which can be decoded into a sentence or expression in the natural language space* L *by the GPT model.*

Lemma 3 (GPT as an Approximation of $p^{-1} \circ p$)**.** *Under Assumptions 9 and 10, a GPT model trained using autoregressive self-supervised learning, as described in Definition 14, can be considered as an approximation of the function $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$.*

Proof. Let $x \in \mathrm{L}$ be a sentence or expression in the natural language space. By Definition 18, we have $p(x) \in \mathcal{K}$, which is the corresponding point in the knowledge space.

Now, consider the GPT inverse projection function g as defined in Definition 23. We have $g(p(x)) \in \mathcal{R}$, which is a point in the representation space. Since GPT is trained using autoregressive self-supervised learning as described in Definition 14, it learns to approximate the function $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$ by minimizing the difference between its own output and the original input.

Let $\hat{x} \in \mathrm{L}$ be the output of the GPT model for the input x. By the assumption of the existence of the GPT inverse projection function (Assumption 10), there exists a function $g : \mathcal{K} \to \mathcal{R}$ such that $g(p(x)) \in \mathcal{R}$.

Under the assumption of smoothness of the representation space (Assumption 9), small changes in the coordinates of a point in the space correspond to small changes in the information or meaning conveyed by the corresponding token or sequence in the GPT model. Since the GPT model is trained to minimize the difference between its output and the input, we have $\hat{x} \approx x$.

Therefore, the GPT model can be considered as an approximation of the function $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$. □

Lemma 3 suggests that a well-trained GPT model can approximate the composition of the projection function p and its inverse p^{-1}. This approximation enables the GPT model to effectively learn the structure and relations in the natural language space by transforming input sentences and expressions into a suitable representation space, processing them, and then transforming the resulting representations back into the natural language space. This capability allows the GPT model to perform a wide range of natural language understanding and generation tasks, as it can manipulate and reason about language data in a continuous and differentiable space.

Theorem 3 (GPT Representation Space as an Approximation of Knowledge Space)**.** *Under Assumptions 9 and 10, the representation space $\mathcal{R}$ of a GPT model trained using autoregressive self-supervised learning can be considered as an approximation of the knowledge space $\mathcal{K}$, where the GPT model serves as an approximation of the function $p^{-1} \circ p$.*

Proof. Let $x \in \mathrm{L}$ be a sentence or expression in the natural language space. By Lemma 3, the GPT model serves as an approximation of the function $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$, meaning that the GPT model learns to approximate the transformation from the natural language space to the knowledge space and back to the natural language space.

By Definition 18, we have $p(x) \in \mathcal{K}$, which is the corresponding point in the knowledge space. Now, consider the GPT inverse projection function g as defined in Definition 23. We have $g(p(x)) \in \mathcal{R}$, which is a point in the representation space.

As the GPT model is an approximation of $p^{-1} \circ p : \mathrm{L} \to \mathrm{L}$, it should also learn to approximate the inverse projection function g that maps points from the knowledge space to the representation space. Thus, the GPT model learns to approximate the transformation from the knowledge space to the representation space. Since the GPT model is trained to minimize the difference between its output and the input, we have $g(p(x)) \approx p(x)$.

Therefore, under Assumptions 9 and 10, the representation space $\mathcal{R}$ of a GPT model trained using autoregressive self-supervised learning can be considered as an approximation of the knowledge space $\mathcal{K}$. □

Theorem 3 serves as a crucial mathematical underpinning in our exploration of GPT-based models. It enables us to unravel the internal mathematical workings of these models by positing the GPT representation space $\mathcal{R}$ as an approximation of the knowledge space $\mathcal{K}$. This formalized definition brings a mathematical rigor to the understanding of the transformation process that the GPT model performs in mapping the natural language space L to the knowledge space $\mathcal{K}$ and vice versa through its layers of non-linear transformations and attention mechanisms. The proof of the theorem further buttresses this interpretation.

This assertion can be viewed as an encapsulation of the intuition prevalent among experts studying GPT-based models. The GPT model is essentially learning to transform natural language inputs into a dense representation space and vice versa. This dense representation is what we are referring to as the GPT representation space $\mathcal{R}$. Thus, the $\mathcal{R}$ serves as an approximation of $\mathcal{K}$, capturing the semantic essence of the sentences or expressions in the natural language space L.

5. Challenges and Limitations of GPT and Autoregressive Self-Supervised Learning

Definition 24 (Incomplete Inverse Projection Function)**.** *An incomplete inverse projection function is an inverse projection function $p^{-1} : \mathcal{K} \to \mathrm{L}$ that may not perfectly map all points in the knowledge space $\mathcal{K}$ to their corresponding points in the natural language space* L*. This implies that certain types of information or meaning may not be adequately represented in* L *using the inverse projection function p^{-1}.*

The projection function limitation refers to the inability of the projection function $p : \mathrm{L} \to \mathcal{K}$ and its inverse $p^{-1} : \mathcal{K} \to \mathrm{L}$ to accurately capture and represent specific knowledge or meaning within the natural language space L and knowledge space $\mathcal{K}$. This

limitation could be attributed to the complexity of the relationship between these spaces or the insufficiency of the model's architecture in representing certain types of knowledge.

Assumption 11 (Incomplete Inverse Projection Function)**.** *We assume that the inverse projection function* $p^{-1} : \mathcal{K} \to \mathrm{L}$*, as described in Assumption 7, may be incomplete in the sense that it may not adequately map certain types of knowledge or meaning from* $\mathcal{K}$ *to* L*, as defined in Definition 24.*

Assumption 12 (Projection Function Limitation Impact)**.** *We assume that the projection function limitation could adversely affect the performance of GPT-based models, such as ChatGPT, in tasks that require precise mapping between the knowledge space* $\mathcal{K}$ *and the natural language space* L*. This limitation could manifest as inaccuracies, ambiguities, or inconsistencies in the generated responses, particularly when dealing with complex or nuanced information.*

The impact of the projection function limitation on GPT-based models, as described in Assumption 12, not only influences the quality of the generated responses but can also affect the model's ability to make reliable inferences. In particular, when the projection function and its inverse fail to accurately map between the knowledge space and the natural language space, the model may encounter difficulties in synthesizing relevant information, comprehending contextual clues, and adapting to new or evolving concepts. This limitation may be exacerbated when handling specialized domains or interdisciplinary subjects where a precise understanding of terminology, relationships, and dependencies is paramount for generating coherent and contextually accurate responses.

Remark 8 (Calculation Limitation)**.** *Assumption 11 implies that certain types of knowledge, such as calculation, may be difficult to represent in the natural language space* L *using the inverse projection function* p^{-1}*. Consequently, GPT-based models may face challenges in performing accurate calculations due to the limitations of their inverse projection function.*

Another potential challenge arising from the limitations of the inverse projection function in GPT-based models is the difficulty in processing and reasoning about abstract concepts, particularly when they involve logical, mathematical, or scientific principles. Due to the inherent complexity and often non-linear nature of these concepts, it can be difficult for the inverse projection function to effectively represent the associated knowledge within the natural language space. This may lead to suboptimal performance when attempting to generate responses that require reasoning about abstract or complex ideas, as the model may struggle to accurately represent and manipulate the relevant information within its internal representations.

Assumption 12 suggests that GPT-based models might face challenges in accurately understanding and interpreting certain types of information due to the limitations of the projection function and its inverse. This could result in a lack of understanding of the underlying meaning or context of the input data, leading to inappropriate or irrelevant responses.

Definition 25 (Model Complexity)**.** *Model complexity refers to the number of parameters or the depth of a deep learning model. A model with a higher complexity generally has more parameters or deeper layers, enabling it to capture more intricate patterns and relationships within the input data.*

Corollary 1 (Complexity and Approximation)**.** *Given Theorem 1, a deep learning model with a larger complexity, as defined in Definition 25, is more likely to satisfy the conditions of the Universal Approximation Theorem, allowing it to approximate a target function with higher accuracy.*

Corollary 2 (Complexity and Efficiency Trade-off)**.** *Given Definition 25, there exists a trade-off between model complexity and model efficiency in deep learning models. Increasing the complexity of a model may improve its ability to approximate a target function, as stated in Corollary 1, but*

it may also lead to increased computational costs and resource requirements, thereby reducing the model's efficiency.

Moreover, the trade-off between model complexity and efficiency, as described in Corollary 2, has implications for the practicality and accessibility of GPT-based models. As these models grow in size and complexity to better approximate target functions, their computational requirements can become increasingly demanding, making them more difficult to deploy and maintain in real-world applications. This can be particularly challenging for smaller organizations or individuals with limited resources who may struggle to harness the full potential of these large-scale models. Additionally, the increased computational demands can lead to higher energy consumption and environmental concerns, further highlighting the need to balance complexity and efficiency in the development of GPT-based models.

Corollary 3 (Model Complexity and Generalization)**.** *While increasing model complexity, as defined in Definition 25, may improve the ability of a deep learning model to approximate a target function, as stated in Corollary 1, it may also lead to overfitting, resulting in reduced generalization capabilities. This trade-off between complexity and generalization is an important consideration when designing and training GPT-based models.*

Example 3 (Large Language Models and Universal Approximation)**.** *Large language models, such as GPT-based models, have gained popularity due to their ability to satisfy the conditions of the Universal Approximation Theorem. The large number of parameters in these models enables them to capture intricate patterns and relationships within the input data, allowing them to perform tasks such as natural language understanding and generation with high accuracy. Consider a large language model such as GPT-3 being utilized for translation tasks. Given a complex sentence in English, the model, due to its extensive parameter space, is able to understand the intricate relationships between words and their context, translating it accurately to another language such as French. This capability demonstrates how the large number of parameters enables GPT-3 to approximate the intricacies of language translation.*

The advantages of large language models, such as GPT-based models, in terms of their ability to satisfy the conditions of the Universal Approximation Theorem, as described in Example 3, should not overshadow the potential pitfalls associated with their scale. As these models grow in size, they may become increasingly susceptible to overfitting, noise, or biases present in the training data. This can manifest as a heightened sensitivity to specific patterns, phrases, or concepts within the input data, potentially leading to the generation of responses that are less adaptable, less diverse, or less contextually appropriate. Consequently, it is important to carefully consider the impact of model size and complexity on both the benefits and potential drawbacks associated with GPT-based models.

Example 4 (Large Language Models and Generalization Challenges)**.** *While large language models, such as GPT-based models, excel at capturing intricate patterns and relationships within input data, they may also face challenges related to generalization. As discussed in Corollary 3, the trade-off between complexity and generalization can result in overfitting, limiting the model's ability to generalize to new, unseen data or contexts. Despite the impressive capabilities of large language models, generalization remains a challenge. For instance, a GPT-3 model trained predominantly on English literature might struggle to accurately generate text in the style of an obscure, regional dialect or an emerging online slang despite its vast parameter space. This example illustrates the trade-off between complexity and generalization and the resultant risk of overfitting to familiar data at the expense of novel contexts.*

6. Conclusions

In this paper, we have explored the mathematical foundations of GPT-based models such as ChatGPT, delving into the intricate relationships between the natural language space L, knowledge space $\mathcal{K}$, and the representation space $\mathcal{R}$. We have formalized key

concepts, definitions, assumptions, and theorems to provide a rigorous understanding of these models' underlying mechanisms.

Our investigation has revealed that GPT-based models, when trained using autoregressive self-supervised learning, can be considered as approximations of the composition of the projection function $p : \mathrm{L} \to \mathcal{K}$ and the inverse projection function $p^{-1} : \mathcal{K} \to \mathrm{L}$ (refer to the second research aspect in the introduction). Consequently, the GPT representation space $\mathcal{R}$ serves as an approximation of the knowledge space $\mathcal{K}$, capturing and preserving the information or meaning conveyed by sentences or expressions in L (pertaining to the first and third research aspects in the introduction).

Notwithstanding their remarkable capabilities, we have identified certain limitations of GPT-based models stemming from incomplete inverse projection functions, which may not adequately map all points in the knowledge space $\mathcal{K}$ to their corresponding points in the natural language space L (as outlined in the fourth research aspect in the introduction). This shortcoming results in challenges, such as difficulties in performing accurate calculations, which are inherently problematic for models such as ChatGPT (as illustrated in Examples 3 and 4).

In the pursuit of elucidating the complex mathematical foundations underpinning the GPT model, particularly its functional aspects denoted as $p^{-1} \circ p(s)$, we concede the inherent challenge of directly validating these findings through empirical experimentation due to their abstract nature. Nonetheless, our endeavor was not solely confined to the construction of mathematical arguments; instead, we endeavored to make these abstractions more comprehensible to the readers. To this end, we integrated illustrative examples within our discourse wherever feasible. These examples, coalesced with our theoretical discourse, are aimed at providing the readers with a more tangible grasp of the mathematical constructs that govern the GPT model. The objective of our study was twofold: firstly, to maintain the requisite mathematical rigor, and secondly, to enhance the accessibility of our exposition to a broad spectrum of readers. We trust that our efforts have been successful in striking a harmonious balance between these two critical aspects of academic writing.

This study has provided a mathematical characterization of GPT-based LLMs, establishing a framework that can serve as a launchpad for future investigations in this domain. Our findings, particularly the mathematical underpinnings of how GPT models capture and represent the semantic essence of natural language, present avenues for the optimization and expansion of these models. By unveiling the limitations of GPT-based models, we open doors for future research to focus on addressing these challenges, potentially advancing the development of LLMs that offer more accurate and comprehensive mappings between natural language and knowledge spaces. Our exploration of GPT as an approximation of the projection function and its inverse has potential implications for the development of more efficient, effective, and robust LLMs, thereby driving advancements in the field of language understanding and generation.

In the broader context, this work has several potential implications for future advancements in the field of GPT-based LLMs. Firstly, our formal definition of the natural language space L and the knowledge space $\mathcal{K}$, along with the associated projection functions, provides a strong mathematical foundation for understanding the complex mechanisms underlying these models. This understanding can guide future efforts to develop more efficient dimensionality reduction techniques, which could significantly improve the computational efficiency of these models.

Moreover, our theoretical analysis of the projection function p and its inverse p^{-1} may provide valuable insights for enhancing the language generation capabilities of GPT-based LLMs. Specifically, our observations regarding the limitations of the inverse projection function could spur research toward methods for improving these models' ability to generate accurate and contextually appropriate responses.

Funding: This work was supported by a research grant funded by Generative Artificial Intelligence System Inc. (GAIS).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. In *OpenAI Technical Report*; OpenAI Inc.: San Francisco, CA, USA, 2019.
2. OpenAI. GPT-4 Technical Report. In *OpenAI Technical Report*; OpenAI Inc.: San Francisco, CA, USA, 2023.
3. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
4. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. In *OpenAI Technical Report*; OpenAI Inc.: San Francisco, CA, USA, 2018.
5. Tirumala, K.; Markosyan, A.; Zettlemoyer, L.; Aghajanyan, A. Memorization without overfitting: Analyzing the training dynamics of large language models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 38274–38290.
6. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.H.; Le, Q.V.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 29 November 2022.
7. Kung, T.H.; Cheatham, M.; Medenilla, A.; Sillos, C.; De Leon, L.; Elepaño, C.; Madriaga, M.; Aggabao, R.; Diaz-Candido, G.; Maningo, J.; et al. Performance of ChatGPT on USMLE: Potential for AI-assisted medical education using large language models. *PLoS Digit. Health* **2023**, *2*, e0000198. [CrossRef] [PubMed]
8. Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv* **2019**, arXiv:1909.08053.
9. Lee, M. A Mathematical Investigation of Hallucination and Creativity in GPT Models. *Mathematics* **2023**, *11*, 2320. [CrossRef]
10. Carlini, N.; Tramer, F.; Wallace, E.; Jagielski, M.; Herbert-Voss, A.; Lee, K.; Roberts, A.; Brown, T.B.; Song, D.; Erlingsson, U.; et al. Extracting Training Data from Large Language Models. In Proceedings of the USENIX Security Symposium, Virtual, 11–13 August 2021; Volume 6.
11. Hu, E.J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W. Lora: Low-rank adaptation of large language models. *arXiv* **2021**, arXiv:2106.09685
12. Ko, K.; Yeom, T.; Lee, M. Superstargan: Generative adversarial networks for image-to-image translation in large-scale domains. *Neural Netws.* **2023**, *162*, 330–339. [CrossRef]
13. Ku, H.; Lee, M. TextControlGAN: Text-to-Image Synthesis with Controllable Generative Adversarial Networks. *Appl. Sci.* **2023**, *13*, 5098. [CrossRef]
14. Kim, J.; Lee, M. Class-Continuous Conditional Generative Neural Radiance Field. *arXiv* **2023**, arXiv:2301.00950.
15. Kim, I.; Lee, M.; Seok, J. ICEGAN: Inverse covariance estimating generative adversarial network. *Mach. Learn. Sci. Technol.* **2023**, *4*, 025008. [CrossRef]
16. Luo, R.; Sun, L.; Xia, Y.; Qin, T.; Zhang, S.; Poon, H.; Liu, T.Y. BioGPT: Generative pre-trained transformer for biomedical text generation and mining. *Briefings Bioinform.* **2022**, *23*, bbac409. [CrossRef] [PubMed]
17. Zhu, Q.; Zhang, X.; Luo, J. Biologically Inspired Design Concept Generation Using Generative Pre-Trained Transformers. *J. Mech. Des.* **2023**, *145*, 041409. [CrossRef]
18. Albelwi, S. Survey on self-supervised learning: Auxiliary pretext tasks and contrastive learning methods in imaging. *Entropy* **2022**, *24*, 551. [CrossRef]
19. Liu, X.; Zhang, F.; Hou, Z.; Mian, L.; Wang, Z.; Zhang, J.; Tang, J. Self-supervised learning: Generative or contrastive. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 857–876. [CrossRef]
20. Jaiswal, A.; Babu, A.R.; Zadeh, M.Z.; Banerjee, D.; Makedon, F. A survey on contrastive self-supervised learning. *Technologies* **2020**, *9*, 2. [CrossRef]
21. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
22. Lu, Y.; Lu, J. A universal approximation theorem of deep neural networks for expressing probability distributions. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 3094–3105.
23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6000–6010.
24. Xu, R.; Wang, X.; Chen, K.; Zhou, B.; Loy, C.C. Positional encoding as spatial inductive bias in gans. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13569–13578.
25. Zheng, J.; Ramasinghe, S.; Lucey, S. Rethinking positional encoding. *arXiv* **2021**, arXiv:2107.02561
26. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450
27. Li, J.; Wang, X.; Tu, Z.; Lyu, M.R. On the diversity of multi-head attention. *Neurocomputing* **2021**, *454*, 14–24. [CrossRef]

28. Voita, E.; Talbot, D.; Moiseev, F.; Sennrich, R.; Titov, I. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv* **2019**, arXiv:1905.09418.
29. Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T.B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; Amodei, D. Scaling laws for neural language models. *arXiv* **2020**, arXiv:2001.08361.
30. Orlitsky, A. Information Theory. In *Encyclopedia of Physical Science and Technology*, 3rd ed.; Meyers, R.A., Ed.; Academic Press: New York, NY, USA, 2003; pp. 751–769. [CrossRef]
31. Brown, P.F.; Della Pietra, S.A.; Della Pietra, V.J.; Lai, J.C.; Mercer, R.L. An estimate of an upper bound for the entropy of English. *Comput. Linguist.* **1992**, *18*, 31–40.
32. Santini, S.; Jain, R. Similarity measures. *IEEE Trans. Pattern Anal. Mach. Intell.* **1999**, *21*, 871–883. [CrossRef]