Language as a Mathematical Object: A Function Approximation Perspective for NLP and Generative AI

Matthew Long
Magneton Labs

February 4, 2025

Abstract

Language, often viewed as a system of symbols and rules, can also be characterized as a *mathematical function* from an input domain (tokens, phonemes, or characters) to an output domain (syntactic or semantic representations). In this paper, we present a clear conceptual framework for understanding language as a mathematical object that is naturally amenable to *function approximation* techniques. We argue that this point of view can greatly simplify modeling and improve the interpretability of methods in natural language processing (NLP) and generative AI. We discuss how mathematical foundations from areas such as the Kolmogorov-Arnold theorem and general function approximation theory can foster new architectures and approaches for language tasks, including generative modeling, machine translation, and language understanding.

1 Introduction

Language is a rich, dynamic medium for communication, characterized by a multitude of rules, patterns, and nuances. Although linguists often analyze language in terms of syntax, semantics, and pragmatics, it is also productive to view language as a *function* acting on a domain of discrete symbols (e.g., words, tokens, characters) and mapping them to structured outputs (e.g., linguistic meanings, next-token probabilities, sentiment scores).

Machine learning approaches to NLP and generative AI have taken multiple forms over the past decades, from rule-based systems to neural networks that often act as "black boxes." A unifying thread across these approaches is the **function approximation perspective**: many language tasks reduce to learning a mapping from one set of representations to another. By framing language as an approachable mathematical function, we create new opportunities for:

- **Interpretable Models:** Illuminating how words and phrases map to semantic or syntactic structures.

- **Generalization:** Leveraging the vast theory of function approximation to ensure robust, data-efficient learning.
- **Novel Architectures:** Applying insights from classical approximation theorems to design neural and hybrid models specifically tailored to language tasks.

The remainder of this paper is organized as follows. Section 2 formalizes the notion of language as a mathematical function. Section 3 introduces the fundamental concepts of function approximation, highlighting how they naturally tie into NLP. Section 4 explores how this perspective directly informs model building in modern generative AI. Finally, Section 6 summarizes key takeaways and outlines future research.

2 Language as a Mathematical Function

2.1 Language Mappings

At its core, language can be interpreted as a mapping from an *input set* of symbols or tokens to an *output set* of meaningful constructs. In many NLP tasks, we capture the mapping:

$$f : \mathcal{X} \to \mathcal{Y},$$

where:

- $\mathcal{X}$ is a domain of linguistic inputs (e.g., sequences of tokens, phonemes, or word embeddings).
- $\mathcal{Y}$ is a range or codomain of outputs (e.g., part-of-speech tags, next-word predictions, semantic vectors, or entire text sequences).

Depending on the task, f could be a classification function (for sentiment analysis), a translation function (for machine translation), or a generative function (for text generation). Despite differences in task formulation, each instance can be viewed as *learning a function* that captures the structure of language.

2.2 Why Treat Language as a Function?

Clarity and Simplicity. It offers a unifying framework: any NLP task can be re-framed in terms of function learning or function composition.

Mathematical Rigor. Well-established results in approximation theory can be applied to guarantee that certain classes of functions can be learned given sufficient data and representational capacity.

Interpretability. By decomposing or factorizing a learned model into sub-functions, one can often pinpoint linguistic aspects or features the function leverages.

3 Fundamentals of Function Approximation

3.1 Approximation of High-Dimensional Functions

Classical results in approximation theory, such as the Universal Approximation Theorem [1] or the Kolmogorov-Arnold representation theorem [2, 3], show that continuous functions from high-dimensional spaces to the real numbers can be approximated arbitrarily well using suitable building blocks. These building blocks can be polynomials, splines, or even simple neural networks.

In the context of language, we must handle *discrete* inputs (tokens, characters) and often *sequence-to-sequence* mappings rather than mapping $\mathbb{R}^n$ to $\mathbb{R}$. However, the spirit remains the same: if language can be modeled as a suitably continuous (or smooth) function in a latent space (e.g., word embeddings), then a function approximator with adequate structure can model the relationships among tokens.

3.2 Neural Networks as Function Approximators

Modern neural networks, from basic `MLPs` to `Transformers`, can be viewed as complex function approximators. A Transformer-based language model, for example, is trained to approximate the probability distribution:

$$P(\text{next token} \mid \text{previous tokens}) \approx f(\text{previous tokens}),$$

where f is a high-dimensional function parameterized by attention weights, feed-forward layers, and other components.

3.3 Benefits of a Functional View

- **Modularity:** Functions can be composed, allowing for hierarchical or layered approaches (e.g., lexical analysis $\rightarrow$ syntactic analysis $\rightarrow$ semantic representation).
- **Theoretical Guarantees:** If f belongs to a class of approximable functions, then theoretical bounds (e.g., on error or sample complexity) can be derived.
- **Interpretation and Analysis:** Developers can analyze each sub-function or layer to understand how language features are transformed.

4 Applications in NLP and Generative AI

The concept of "language as a function" naturally extends to a variety of NLP and Generative AI tasks:

4.1 Language Modeling

Language modeling is often defined as predicting the probability of the next token, given a sequence of preceding tokens:

$$\hat{y} = f(x_1, x_2, \ldots, x_n),$$

where $\hat{y}$ could be a probability distribution over the next token. Treating f as a learnable function clarifies that the goal is not just *pattern recognition* but approximating a systematic mapping from past tokens to future tokens.

4.2 Machine Translation

Machine translation transforms text from one language to another:

$$\text{English} \xrightarrow{f} \text{German}.$$

A function-based perspective emphasizes how we learn a continuous (or piecewise continuous) mapping in an embedding space that preserves semantic relationships. By framing translation as approximating this mapping, we open the door to:

- Hybrid symbolic-connectionist methods, where sub-functions handle morphological rules or named entities.
- New ways to regularize or constrain f to reflect linguistic knowledge (like word-order constraints).

4.3 Generative Text Models

Generative models (e.g., GPT variants) implement a function:

$$f : \mathcal{X} \to \mathcal{X},$$

mapping a partial sequence to an extension. From a functional standpoint, there is nothing mysterious about text generation; it is simply the iterative evaluation of a learned function on its own outputs. For interpretability and control:

- One can explicitly modify certain "sub-functions" if the model is factorized appropriately.
- One can harness existing approximation theories to determine how many parameters are sufficient to achieve a certain fidelity in text generation.

4.4 Dialogue Systems and QA

In dialogue systems and question-answering (QA), a function f maps user queries and context to responses. The function-based view can encourage improved designs, e.g.,

$$f(\text{User Query}, \text{History}) = \text{System Response}.$$

Because the function must incorporate multi-turn context, attention mechanisms or memory networks can be seen as *function decomposition strategies*, distributing different parts of the function across time steps.

5 Challenges and Future Directions

5.1 Discrete vs. Continuous Spaces

Language involves discrete tokens, but many function approximation theorems assume continuity. Embedding discrete tokens into continuous vector spaces (word embeddings) partially addresses this issue, but future research could develop continuous analogs or advanced discrete approximation theorems explicitly tailored to language.

5.2 Long-Range Dependencies

Natural language frequently exhibits long-range dependencies (e.g., long sentences, paragraphs, documents). Classical function approximation results usually consider vector inputs of fixed dimension, but in language, the number of tokens can vary widely. Attention-based models partially solve this by focusing on relevant context, yet more explicit approximations for variable-length sequences require new theory and architectures.

5.3 Interpretability and Transparency

Although the function viewpoint can encourage more interpretable models, current large-scale neural networks remain challenging to interpret in practice. Research is needed to design function approximators that are *both* powerful enough for large-scale tasks and sufficiently transparent to be understood by end users (e.g., legal or medical professionals).

6 Conclusion

Presenting language as a mathematical object grounded in function approximation theory simplifies and unifies many aspects of NLP and generative AI. This perspective illuminates how neural networks and related models are ultimately *function approximators*, bridging the discrete and continuous realms to capture complex linguistic phenomena.

By harnessing tools and theorems from approximation theory, we can:

- Develop more interpretable and robust architectures.
- Create bridges between symbolic rules and learned representations.
- Apply theoretical guarantees to model capacity and generalization behavior.

The future of NLP and generative AI will likely see a continued shift toward even more powerful and efficient function approximators, with increasing emphasis on *interpretability, modularity, and*

theoretical grounding. This paper stands as an invitation to explore the vast possibilities that emerge when language is treated *unquestionably* as a mathematical function — one that is *elegant*, *universal*, and *open to deeper analytic inquiry.*

References

[1] Kurt Hornik. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[2] A. N. Kolmogorov. On the representation of continuous functions of several variables by superposition of continuous functions of a smaller number of variables. *Dokl. Akad. Nauk SSSR*, 114:953–956, 1957.

[3] V. I. Arnold. On functions of three variables. *Soviet Math. Dokl.*, 5:599–601, 1963.