BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

Thesis

EFFICIENTLY APPROXIMATE THE ATTENTION COMPUTATION PROBLEM THROUGH THE SOFTMAX REGRESSION

by

JUNZE YIN

Submitted in partial fulfillment of the

requirements for the degree of

Master of Arts

2024

Approved by

First Reader

Mark Kon, PhD
Professor of Mathematics and Statistics

Second Reader

Julio Castrillon, PhD
Professor of Mathematics and Statistics

Third Reader

Zhao Song, PhD
Researcher

Acknowledgments

I would like to express my deepest gratitude to Professor Kon, Dr. Song, Professor Castrillon, Professor Fried, and Professor Previato. This thesis could not have been completed without their tremendous help.

First and foremost, I want to thank my thesis advisor, Professor Kon. I consider myself very fortunate to have been advised by Professor Kon. Professor Kon taught me real analysis and provided countless invaluable suggestions, not only for this thesis but also for my future plans. I extremely appreciate all the precious time and help Professor Kon has given me.

Second, I would like to thank Dr. Song. For almost two years, Dr. Song has been teaching me computer science and mathematics concepts and guiding my theoretical computer science research. I am extremely grateful for his time and guidance.

Third, I want to thank Professor Castrillon for providing me with insightful suggestions for the introduction of this thesis and for his time in participating in the oral defense.

Fourth, I appreciate Professor Fried for teaching me numerical analysis, which has helped me gain a deeper understanding of theoretical computer science concepts.

Last but not least, I would like to thank Professor Previato for teaching me how to conduct research and for providing numerous insightful suggestions.

I truly appreciate the help I received from all of them.

Junze Yin

B.A./M.A. Student

Department of Mathematics and Statistics

EFFICIENTLY APPROXIMATE THE ATTENTION COMPUTATION PROBLEM THROUGH THE SOFTMAX REGRESSION

JUNZE YIN

ABSTRACT

In this thesis, we analyze the softmax regression problem, which was originally proposed and studied by [DLS23]. We aim to provide a clear explanation of [DLS23] and make it more approachable to the Mathematics community. We first introduce the mathematical formulation of the attention mechanism and the softmax regression problem. We then provide a detailed overview of the techniques used to analyze the softmax regression problem, including Hessian decomposition, proving the positive definiteness of the Hessian, and establishing the Lipschitz continuity of the Hessian. Finally, we present the main result of [DLS23], namely a randomized algorithm that can solve the softmax regression problem in logarithmic time with respect to the desired accuracy while maintaining a high probability of returning a solution close to the optimal. The algorithm utilizes an approximate Newton's method, where the Hessian matrix is approximated efficiently. Theoretical guarantees are provided for the algorithm's convergence and accuracy.

Contents

Chapter 1

Introduction

In this thesis, our goal is to provide a detailed exposition of the results from a theoretical machine learning paper [DLS23] using Mathematical and Statistical language to better facilitate interdisciplinary discussion between these fields. Large language models (LLMs) have emerged as a powerful AI technique for understanding and generating human-like text. Examples of prominent LLMs include BERT [DCLT18], OPT [ZRG+22], GPT series [BMR+20, RNS+18, RWC+19, Ope23], Llama [TLI+23], and Llama 2 [TMS+23]. These models are capable of handling various language tasks, including language modeling [MMS+19], sentiment analysis [UAS+20], creative writing [Cha22, Ope23], and natural language translation [HWL21]. A key component of enabling the powerful performance of LLMs is the attention mechanism, which was first proposed in [VSP+17].

Let $\mathbb{R}$ and $\mathbb{Z}_+$ be sets of real numbers and positive integers respectively. For all $n, d \in \mathbb{Z}_+$, we define $\mathbb{R}^{n \times d}$ to be a set of all $n \times d$ matrices whose entries are real numbers. An attention matrix $A \in \mathbb{R}^{n \times n}$ captures relationships between words from input text. Specifically, each row $(A_{i,*})^\top \in \mathbb{R}^n$ of the attention matrix A has an index i which corresponds to an enumeration of our word universe. Thus, i represents the i-th word in this universe. Similarly, the same indexing is used for the columns $A_{*,j} \in \mathbb{R}^n$ of the attention matrix A, where $i, j \in [n]$, and $[n] := \{1, 2, 3, \dots n\}$. Attention mechanisms enable models to emphasize influential parts of the input when producing outputs, rather than weighting all words uniformly. The attention weights,

calculated through a softmax function over the inputs, determine a relative emphasis or concentration allocated to each word. By selectively focusing computational resources on the highly related inputs, attention layers allow models to handle longer sequences and extract meanings more efficiently. Rather than requiring equal influence from all words without considering about relevance, attention mechanisms apply non-uniform significance estimates combined to each output.

Mathematically, the attention matrix is defined as follows:

Definition 1.0.1 (Attention matrix)**.** *Let $Q, K \in \mathbb{R}^{n \times d}$. The (Q, K)-attention matrix $A \in \mathbb{R}^{n \times n}$ is defined as*

$$A := \exp(QK^\top / d)$$

Computing the attention matrix is computationally expensive, requiring $O(n^2)$ time where n is the number of words. This quadratic scaling becomes prohibitively slow for large n. The exact attention problem is defined as follows:

Definition 1.0.2 (The exact attention problem)**.** *Let $Q, K, V \in \mathbb{R}^{n \times d}$. Let $A \in \mathbb{R}^{n \times n}$ be defined as in Definition 1.0.1. For all $i, j \in [n]$, we let* $\mathrm{diag} : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ *be defined as*

$$\mathrm{diag}(x)_{i,j} := \begin{cases} x_i & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Let $D \in \mathbb{R}^{n \times n}$ be defined as $D := \mathrm{diag}(A\mathbf{1}_n)$, where $\mathbf{1}_n \in \mathbb{R}^n$ is defined as $(\mathbf{1}_n)_i := 1$ for all $i \in [n]$. Then, the exact attention problem $\mathrm{Att}(Q, K, V) \in \mathbb{R}^{n \times d}$ is defined as

$$\mathrm{Att}(Q, K, V) := D^{-1}AV.$$

The quadratic complexity $O(n^2)$ of the attention computation makes it challenging for LLMs to process very long input sequences efficiently. To the best of our knowledge, there are three main approaches to improve the efficiency of attention computation and enable LLMs to handle longer sequences. First, some works transform the

exact attention problem into simpler forms like sparse attention [ZGD+20, KKL20], linear attention [BPC20], and softmax attention [DLS23], which aim to reduce the computational complexity while still capturing long-range dependencies. Although varying the attention computation may result in an improvement in the efficiency of LLMs, it may also lead to a decrease in model performance [DLZ+23]. Second, some models, like [RPJL19], incorporate recurrent architectures or external memory components to store and retrieve long-term context, allowing them to process longer sequences incrementally. Third, preprocessing techniques like text summarization [ZWZ19] can be used to condense long input sequences into more manageable lengths while preserving the essential information, which avoids LLMs processing very long input sequences.

In this thesis, we give a detailed analysis and summary of the results and techniques from [DLS23]. We analyze an efficient attention variants approach: softmax attention. Mathematically, the softmax regression is defined as follows:

Definition 1.0.3 (Softmax Regression [DLS23])**.** *Let $\|\cdot\|_2$ be the ℓ_2 norm of a vector. For all $x \in \mathbb{R}^d$, we have $\|x\|_2 := \sqrt{\sum_{i=1}^d x_i^2}$. Given $A \in \mathbb{R}^{n \times d}$ and $b \in \mathbb{R}^n$, we define the softmax regression problem as*

$$\min_{x \in \mathbb{R}^d} \| \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \exp(Ax) - b \|_2^2,$$

where $\exp(\cdot)$ is applied entrywisely.

In practice, it is common to take into account regularization [LLR23]. To preserve the convexity of the softmax regression, [DLS23] examine a regularized form of softmax regression.

Definition 1.0.4 (Regularized Softmax Regression [DLS23])**.** *Given $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$, we define the regularized softmax regression problem as finding the minimum value $\min L(x)$, where*

$$L(x) = 0.5 \cdot \| \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \exp(Ax) - b \|_2^2 + 0.5 \cdot \| \operatorname{diag}(w) Ax \|_2^2.$$

Remark 1.0.5. *Note that $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$ are fixed throughout this thesis.*

Chapter 2

Softmax Regression

In this chapter, we present a detailed explanation of the approach and main result of [DLS23] for analyzing the softmax regression problem (Definition 1.0.4). Specifically, in Section 2.1, we introduce notation and present some mathematical background required for the methods used in [DLS23]. In Section 2.2, we give an overview of the techniques used to analyze the softmax regression problem. In Section 2.3, we present the main result of [DLS23], which includes the pseudocode for solving the softmax regression problem and the main theorem providing the theoretical guarantee of the algorithm.

2.1 Preliminaries

In Section 2.1.1, we introduce the basic notation used in this thesis. In Section 2.1.2, we derive the definitions of derivatives related to matrices and vectors from the traditional scalar derivative. In Section 2.1.3, we present the fundamental definitions of numerical analysis and linear algebra. In Section 2.1.4, we introduce the approximate Newton's method.

2.1.1 Notation

Let $\mathbb{N} = \{0, 1, 2, \dots\}$ be the set of natural numbers. We use $\circ$ to denote a binary operation called the Hadamard product which will be defined on the set $\mathbb{R}^d$ or on $\mathbb{R}^{n \times d}$, for arbitrary positive integers n and d. For all $x, y \in \mathbb{R}^d$, we define $x \circ y \in \mathbb{R}^d$

as $(x \circ y)_i := x_i \cdot y_i$, for all $i \in [d]$ and use $\langle x, y \rangle$ to denote the inner product of x and y, that is, $\langle x, y \rangle = \sum_{i \in [n]} x_i \cdot y_i$. For all $A, B \in \mathbb{R}^{n \times d}$, we define $A \circ B \in \mathbb{R}^{n \times d}$ as $(A \circ B)_{i,j} := A_{i,j} \cdot B_{i,j}$, for all $i \in [n]$ and $j \in [d]$.

Let $\|A\|$ be the spectral norm of the matrix $A \in \mathbb{R}^{n \times d}$, which is defined as $\|A\| := \sup_{x \in \mathbb{R}^d} \|Ax\|_2 / \|x\|_2$. For a differentiable loss function $L : \mathbb{R}^d \to \mathbb{R}$, we let $g = \nabla L : \mathbb{R}^d \to \mathbb{R}^d$ denote its gradient and $H : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ denote its Hessian, which is defined as $H(x)_{i,j} := \frac{\mathrm{d}^2 L(x)}{\mathrm{d}x_i \mathrm{d}x_j}$[1].

2.1.2 Matrix Calculus

In this section, we present definitions related to matrix calculus. We present a common definition of scalar-by-scalar derivative below:

Definition 2.1.1 (Scalar-by-scalar derivative)**.** *Let $f : \mathbb{R} \to \mathbb{R}$. The derivative of f with respect to a scalar variable x, $\frac{\mathrm{d}f}{\mathrm{d}x} : \mathbb{R} \to \mathbb{R}$, is defined as*

$$\frac{\mathrm{d}f}{\mathrm{d}x}(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}.$$

Then, based on this definition, we define a vector-by-scalar derivative and scalar-by-vector derivative. These are analogous to the concept of gradient.

Definition 2.1.2 (Vector-by-scalar derivative)**.** *Let $y \in \mathbb{R}^n$. Let x be a scalar variable. Then, we define vector-by-scalar derivative as*

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \begin{bmatrix} \frac{\mathrm{d}y_1}{\mathrm{d}x} \\ \frac{\mathrm{d}y_2}{\mathrm{d}x} \\ \vdots \\ \frac{\mathrm{d}y_n}{\mathrm{d}x} \end{bmatrix}.$$

Definition 2.1.3 (Scalar-by-vector derivative)**.** *Let $y \in \mathbb{R}^n$. Let $x \in \mathbb{R}$. Then, we define scalar-by-vector derivative as*

$$\frac{\mathrm{d}x}{\mathrm{d}y} = \begin{bmatrix} \frac{\mathrm{d}x}{\mathrm{d}y_1} & \frac{\mathrm{d}x}{\mathrm{d}y_2} & \cdots & \frac{\mathrm{d}x}{\mathrm{d}y_n} \end{bmatrix}.$$

[1] In this thesis, all lose functions we consider are in C^2.

After that, we define derivatives of matrices and a vector-by-vector derivative.

Definition 2.1.4 (Matrix-by-scalar derivative)**.** *Let $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}$. Then we define matrix-by-scalar derivative as*

$$\frac{\mathrm{d}A}{\mathrm{d}x} = \begin{bmatrix} \frac{\mathrm{d}A_{1,1}}{\mathrm{d}x} & \frac{\mathrm{d}A_{1,2}}{\mathrm{d}x} & \cdots & \frac{\mathrm{d}A_{1,d}}{\mathrm{d}x} \\ \frac{\mathrm{d}A_{2,1}}{\mathrm{d}x} & \frac{\mathrm{d}A_{2,2}}{\mathrm{d}x} & \cdots & \frac{\mathrm{d}A_{2,d}}{\mathrm{d}x} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}A_{n,1}}{\mathrm{d}x} & \frac{\mathrm{d}A_{n,2}}{\mathrm{d}x} & \cdots & \frac{\mathrm{d}A_{n,d}}{\mathrm{d}x} \end{bmatrix} \in \mathbb{R}^{n\times d}$$

Definition 2.1.5 (Vector-by-vector derivative)**.** *Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^d$. Then, we define vector-by-vector derivative as*

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \begin{bmatrix} \frac{\mathrm{d}y_1}{\mathrm{d}x_1} & \frac{\mathrm{d}y_1}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}y_1}{\mathrm{d}x_n} \\ \frac{\mathrm{d}y_2}{\mathrm{d}x_1} & \frac{\mathrm{d}y_2}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}y_2}{\mathrm{d}x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}y_d}{\mathrm{d}x_1} & \frac{\mathrm{d}y_d}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}y_d}{\mathrm{d}x_n} \end{bmatrix} \in \mathbb{R}^{d\times n}$$

Definition 2.1.6 (Scalar-by-matrix derivative)**.** *Let $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}$. Then we define scalar-by-matrix derivative as*

$$\frac{\mathrm{d}x}{\mathrm{d}A} = \begin{bmatrix} \frac{\mathrm{d}x}{\mathrm{d}A_{1,1}} & \frac{\mathrm{d}x}{\mathrm{d}A_{2,1}} & \cdots & \frac{\mathrm{d}x}{\mathrm{d}A_{n,1}} \\ \frac{\mathrm{d}x}{\mathrm{d}A_{1,2}} & \frac{\mathrm{d}x}{\mathrm{d}A_{2,2}} & \cdots & \frac{\mathrm{d}x}{\mathrm{d}A_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}x}{\mathrm{d}A_{1,d}} & \frac{\mathrm{d}x}{\mathrm{d}A_{2,d}} & \cdots & \frac{\mathrm{d}x}{\mathrm{d}A_{n,d}} \end{bmatrix} \in \mathbb{R}^{d\times n}$$

2.1.3 Numerical Analysis and Linear Algebra

In this section, we introduce key concepts in numerical analysis and linear algebra.

We start with defining positive definite and Lipschitz continuous functions.

Definition 2.1.7 (Positive definite)**.** *Let $A, B \in \mathbb{R}^{n\times n}$. The matrix A is positive definite (PD), denoted as $A \succ 0$, if for all $x \in \mathbb{R}^n \backslash \{\mathbf{0}_n\}$, $x^\top Ax > 0$. Moreover, we say $A \succ B$ if for all $x \in \mathbb{R}^n \backslash \{\mathbf{0}_n\}$, we have $x^\top Ax > x^\top Bx$.*

Definition 2.1.8 (L_h-Lipschitz)**.** *Let $h : \mathbb{R}^m \to \mathbb{R}^m$ be a function. The function h is L_h-Lipschitz if there exists a real number $L_h \geq 0$ such that for all $x, y \in \mathbb{R}^m$,*

$$\|h(x) - h(y)\|_2 \leq L_h \cdot \|x - y\|_2.$$

Now, we present a fact, listing some basic properties of vectors.

Fact 2.1.9. *Let $x, y, z \in \mathbb{R}^d$. Then, we have*

- $\|x\|_2^2 = x^\top x$
- $\langle x, y \rangle = x^\top y = y^\top x$
- $x^\top (y \circ z) = (z^\top y) \circ x$
- $x \circ y = \mathrm{diag}(x) y$

2.1.4 Approximate Newton's Method

In this section, we present definitions and properties of Newton's method and the approximate Newton's method. Newton's method uses a recurrence relation, so we start by presenting the formal definition of a recurrence relation:

Definition 2.1.10 (Recurrence relation)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}^d$. A recurrence relation defined by f is a mapping that produces new elements P_{n+1} of $\mathbb{R}^d$ from a previous element $P_n \in \mathbb{R}^d$. Specifically, the mapping is*

$$P_{n+1} = f(P_n)$$

where $n \in \mathbb{N}$.

Therefore, we have

$$\begin{aligned}
P_1 &= f(P_0) \\
P_2 &= f(P_1) = f(f(P_0)) \\
P_3 &= f(P_2) = f(f(P_1)) = f(f(f(P_0))) = f^{(3)}(P_0) \\
&\vdots \\
P_{n+1} &= f(P_n) = f(f(P_{n-1})) = \ldots = f^{(n+1)}(P_0) \\
&\vdots
\end{aligned}$$

An initial value P_0 is called a seed; a sequence from an iteration, P_0, P_1, P_2, ... is called an orbit.

Definition 2.1.11 ((l, M)-good Loss function)**.** *Let $l, M > 0$ be arbitrary real numbers. Let $L : \mathbb{R}^d \to \mathbb{R}$ be an arbitrary loss function. L is (l, M)-good if*

- *L* **has a unique *l*-local minimum:** *this means that there exists a unique vector $x^* \in \mathbb{R}^d$ such that*

  (1). $\nabla L(x^*) = \mathbf{0}_d$ and

  (2). $H(x^*) \succeq l \cdot I_d$.

- **The Hessian of *L* is *M*-Lipschitz:** *according to Definition 2.1.8, that is, for all $x, y \in \mathbb{R}^d$,*

$$\|H(y) - H(x)\| \leq M \cdot \|y - x\|_2$$

Definition 2.1.12 (Good initialization point)**.** *Given a recurrence relation $x_{n+1} = f(x_n)$ and a (l, M)-good function L, this recurrence relation has a good seed (or good initialization point) x_0 relative to L if*

$$\|x_0 - x^*\|_2 M \leq 0.1l,$$

where x^* denotes the optimal solution of L.

Here, we present a definition of an exact update of Newton's method.

Definition 2.1.13 (Exact update of Newton's method)**.** *Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function. Suppose it has a gradient function $g : \mathbb{R}^d \to \mathbb{R}^d$ and a Hessian function $H : \mathbb{R}^d \to \mathbb{R}^{d \times d}$. The exact update of Newton's method for finding a zero of the function L is a recurrence relation defined on L:*

$$x_{t+1} = x_t - H(x_t)^{-1} \cdot g(x_t).$$

Finding the Hessian matrix is very expensive in many real-world tasks. Therefore, in the algorithm of [DLS23], an approximated computation of the Hessian is utilized. We present related definitions below.

Definition 2.1.14 (ϵ_0-approximate Hessian)**.** *Let $x \in \mathbb{R}^d$ and $H(x) \in \mathbb{R}^{d \times d}$ be a Hessian matrix. For all $\epsilon_0 \in (0, 0.1)$, we define an ϵ_0-approximate Hessian*[2] *$\widetilde{H}(x) \in$*

[2]This approximate Hessian does not need to be a Hessian matrix. It is used to approximate the Hessian $H(x) \in \mathbb{R}^{d \times d}$.

$\mathbb{R}^{d\times d}$ *to be a matrix that satisfies:*

$$(1-\epsilon_0)\cdot H(x) \preceq \widetilde{H}(x) \preceq (1+\epsilon_0)\cdot H(x).$$

Remark 2.1.15. *Note that, ω denotes the currently known best exponent of matrix multiplication, currently $\omega \approx 2.373$ [Wil12, LG14, AW21]. It quantifies the rate at which the computational cost increases with the size of the matrices being multiplied. The higher ω is, the higher the computational cost it takes of multiplying two matrices.*

Lemma 4.5 in [DSW22] states the existence of an algorithm for providing an ϵ_0-approximate Hessian $\widetilde{H}(x_t)$ efficiently. An approximate Hessian is much easier to compute than the Hessian.

Lemma 2.1.16 ([DSW22, SYYZ22])**.** *Let $\epsilon_0, \delta \in (0, 0.1)$. Let $A \in \mathbb{R}^{n\times d}$. Let $\mathrm{nnz}(A) \in \mathbb{N}$ denote the number of nonzero entries of A.*

Then, for all $i \in [n]$, for all $D \in \mathbb{R}^{n\times n}$ satisfying $D_{i,i} > 0$, there exists an algorithm which runs in time

$$O((\mathrm{nnz}(A) + d^\omega)\,\mathrm{poly}(\log(n/\delta)))$$

and outputs an $O(d\log(n/\delta))$ sparse diagonal matrix $\widetilde{D} \in \mathbb{R}^{n\times n}$, i.e. a diagonal matrix where most of the entries are zeros, and the number of non-zero entries is less than or equal to a constant times $d\log(n/\delta)$, such that

$$(1-\epsilon_0)A^\top DA \preceq A^\top \widetilde{D}A \preceq (1+\epsilon_0)A^\top DA.$$

Following from [Ans00, JKL+20, SZZ21, HJS+22, LSZ23], we consider an approximate update of Newton's method.

Definition 2.1.17 (ϵ_0-approximate update Newton's method)**.** *Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function. Suppose it has the gradient function $g : \mathbb{R}^d \to \mathbb{R}^d$ and the Hessian matrix $H : \mathbb{R}^d \to \mathbb{R}^{d\times d}$. Let $\widetilde{H} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ be an ϵ_0-approximate Hessian defined in Definition 2.1.14 and obtained through Lemma 2.1.16, for any $\epsilon_0 \in (0, 0.1)$. An ϵ_0-approximate update of Newton's method is a recurrence relation defined on L:*

$$x_{t+1} = x_t - \widetilde{H}(x_t)^{-1}\cdot g(x_t).$$

Now, we show some mathematical properties of the derivation of a convergent and stable approximate Newton's method through the definitions of the positive definiteness and Lipschitz properties.

Lemma 2.1.18 (Iterative shrinking, Lemma 6.9 on page 32 of [LSZ23]). *For a positive integer t, we define $x_t \in \mathbb{R}^d$ to be the t-th iteration of a recurrence relation $x_{t+1} = f(x_t)$. We let $x^* \in \mathbb{R}^d$ be the unique exact solution of the softmax regression (see Definition 1.0.4), for fixed $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$. Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function which is (l, M)-good (see Definition 2.1.11). Let $r_t := \|x_t - x^*\|_2$. Let $\overline{r}_t := M \cdot r_t$.*

Then, for all $\epsilon_0 \in (0, 0.1)$, we have

$$r_{t+1} \leq 2 \cdot (\epsilon_0 + \overline{r}_t/(l - \overline{r}_t)) \cdot r_t.$$

We define T as the total iterations required by an algorithm. To utilize Lemma 2.1.18, the following induction hypothesis lemma is necessary. This approach is used in [LSZ23].

Lemma 2.1.19 (Induction on the recurrence relation with variable t, Lemma 6.10 on page 34 of [LSZ23]). *For a positive integer t, for each $i \in [t]$, we define $x_i \in \mathbb{R}^d$ to be the i-th iteration of a recurrence relation $x_{t+1} = f(x_t)$. We let $x^* \in \mathbb{R}^d$ be the exact solution of the softmax regression for our choice of $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$. For each $i \in [t]$, we define $r_i := \|x_i - x^*\|_2$. Let $\epsilon_0 \in (0, 0.1)$. Suppose $r_i \leq 0.4 \cdot r_{i-1}$, for all $i \in [t]$. For M and l to be defined for Definition 2.1.11, we will assume $M \cdot r_i \leq 0.1 l$, for all $i \in [t]$.*

Then we have

- $r_{t+1} \leq 0.4 r_t$.
- $M \cdot r_{t+1} \leq 0.1 l$.

Proof. See [LSZ23], Lemma 6.10 on page 34. □

Now, we present a lower bound on β.

Lemma 2.1.20. *Let R be a positive real number. For $A \in \mathbb{R}^{n \times d}$ satisfying $\|A\| \leq R$, $x \in \mathbb{R}^d$ satisfying $\|x\|_2 \leq R$, and β a lower bound of $\langle \exp(Ax), \mathbf{1}_n \rangle$, we have*

$$\beta \geq \exp(-R^2)$$

Note that the Hessian of the loss function L (defined in Definition 1.0.4) is shown to be Lipschitz, namely

$$\|H(x) - H(y)\| \leq \beta^{-2} n^{1.5} \exp(20R^2) \cdot \|x - y\|_2.$$

Therefore, for M being a positive real number, as the Hessian is M-Lipschitz (see Definition 2.1.8), we provide the upper bound on M as:

Lemma 2.1.21. *Let R be a positive real number. For $A \in \mathbb{R}^{n \times d}$ satisfying $\|A\| \leq R$, $x \in \mathbb{R}^d$ satisfying $\|x\|_2 \leq R$, H the hessian of loss function L, we have*

$$M \leq n^{1.5} \exp(30R^2).$$

Proof. It follows from Lemma 2.1.20. □

2.2 Technique Overview

To get the main result, namely Theorem 2.3.1 and Algorithm 1, [DLS23] uses the following techniques: [DLS23] decomposes the Hessian for the Softmax regression, shows that the Hessian is positive definite and Lipschitz, and uses the approximate Newton's method to construct the algorithm for solving the Softmax regression.

2.2.1 Hessian Decomposition

[DLS23] computes the gradient and Hessian of the loss function defined in the softmax regression problem (Definition 1.0.4), for fixed $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$. While these computations are essential, the techniques involved are relatively straightforward compared to other aspects of the paper. Therefore, due to space constraints in this

thesis, we omit certain computational details and prioritize discussing higher-level techniques.

Let's first recall the loss function of the softmax regression problem as defined in Definition 1.0.4.

Definition 2.2.1. *Given $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$, we define the loss function $L_{\mathrm{soft}} : \mathbb{R}^d \to \mathbb{R}$ of the regularized softmax regression (see Definition 1.0.4) as:*

$$L_{\mathrm{soft}}(x) = \min_{x \in \mathbb{R}^d} 0.5 \cdot \| \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \exp(Ax) - b \|_2^2 + 0.5 \cdot \| \operatorname{diag}(w) Ax \|_2^2,$$

where, as defined in Definition 1.0.4, $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$.

Remark 2.2.2. *We refer the gradient of the loss function $L_{\mathrm{soft}} : \mathbb{R}^d \to \mathbb{R}$ as $g_{\mathrm{soft}} : \mathbb{R}^d \to \mathbb{R}^d$ and refer the Hessian of the loss function $L_{\mathrm{soft}} : \mathbb{R}^d \to \mathbb{R}$ as $H_{\mathrm{soft}} : \mathbb{R}^d \to \mathbb{R}^{d \times d}$.*

Let $L_{\mathrm{soft}}(x) := L_{\exp}(x) + L_{\mathrm{reg}}(x)$, where

$$L_{\exp}(x) := 0.5 \cdot \| \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \cdot \exp(Ax) - b \|_2^2 \tag{2.1}$$

$$L_{\mathrm{reg}}(x) := 0.5 \cdot \| WAx \|_2^2 \tag{2.2}$$

To simplify the expression of the loss function (see Definition 2.2.1), we define $f : \mathbb{R}^d \to \mathbb{R}^n$ as

Definition 2.2.3 (Function f, Definition 5.1 of [DLS23])**.** *Let $A \in \mathbb{R}^{n \times d}$ and $x \in \mathbb{R}^d$. We define function $f : \mathbb{R}^d \to \mathbb{R}^n$ as*

$$f(x) := \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \cdot \exp(Ax).$$

Therefore, we have

$$\begin{aligned}
\frac{\mathrm{d}L_{\exp}(x)}{\mathrm{d}x_i} &= \frac{\mathrm{d}}{\mathrm{d}x_i}(0.5 \cdot \|f(x) - b\|_2^2) \\
&= (f(x) - b)^\top \frac{\mathrm{d}}{\mathrm{d}x_i}(f(x) - b) \\
&= (f(x) - b)^\top (-\langle f(x), A_{*,i} \rangle \cdot f(x) + f(x) \circ A_{*,i}) \\
&= A_{*,i}^\top f(x)(f(x) - b)^\top f(x) + (f(x) - b)^\top f(x) \circ A_{*,i} \\
&= A_{*,i}^\top f(x)(f(x) - b)^\top f(x) + A_{*,i}^\top f(x) \circ (f(x) - b) \\
&= A_{*,i}^\top (f(x)(f(x) - b)^\top f(x) + \mathrm{diag}(f(x))(f(x) - b)),
\end{aligned}$$

where the first step follows from the combination of Eq. (2.1) and Definition 2.2.3, the second step follows from the product rule, the third step follows from their computation of $\frac{\mathrm{d}}{\mathrm{d}x_i}(f(x) - b)$ in [DLS23], the fourth step follows from the distributive law and Fact 2.1.9, the fifth step follows from Fact 2.1.9, and the last step follows from Fact 2.1.9 and the distributive law.

We then get

$$\frac{\mathrm{d}^2 L_{\exp}}{\mathrm{d}x_i^2} = A^\top B(x) A \qquad \frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} = A_1^\top W^2 A_1,$$

where

$$\begin{aligned}
B(x) = & \underbrace{\langle 3f(x) - 2b, f(x) \rangle}_{\text{scalar}} \cdot \underbrace{f(x)}_{n \times 1} \underbrace{f(x)^\top}_{1 \times n} + \underbrace{\langle f(x) - b, f(x) \rangle}_{\text{scalar}} \cdot \underbrace{\mathrm{diag}(f(x))}_{n \times n \text{ diagonal matrix}} \\
& + \underbrace{\mathrm{diag}((2f(x) - b) \circ f(x))}_{n \times n \text{ diagonal matrix}} + \underbrace{(b \circ f(x))}_{n \times 1} \cdot \underbrace{f(x)^\top}_{1 \times n} + \underbrace{f(x)}_{n \times 1} \cdot \underbrace{(b \circ f(x))^\top}_{1 \times n}
\end{aligned}$$

2.2.2 Hessian is Positive Definite

The goal is to show that $\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2}$ is positive definite, namely $\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2} \succ 0$ (see Definition 2.1.7).

We have

$$\begin{aligned}\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2} &= \frac{\mathrm{d}^2 L_{\exp}}{\mathrm{d}x^2} + \frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} \\ &= A^\top B(x) A + A^\top W^2 A \\ &= A^\top (B(x) + W^2) A, \end{aligned} \tag{2.3}$$

where the first step follows from the definition of L_{soft}, the second step follows from Eq. (2.1) and Eq. (2.2), and the last step follows from the distributive law.

[DLS23] claims $\|f(x)\|_1 = 1$ without giving a specific proof. This property is important for showing that Hessian is positive definite. To fill this gap, we provide the proof of this property below.

Fact 2.2.4. *Let $f : \mathbb{R}^d \to \mathbb{R}^n$ be defined as in Definition 2.2.3. Then, $\|f(x)\|_1 = 1$.*

Proof. We have

$$\begin{aligned}\|f(x)\|_1 &= \|\langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \cdot \exp(Ax)\|_1 \\ &= \langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \cdot \|\exp(Ax)\|_1 \\ &= (\sum_{i \in [n]} (\exp(Ax))_i \cdot 1)^{-1} \cdot \|\exp(Ax)\|_1 \\ &= (\sum_{i \in [n]} (\exp(Ax))_i)^{-1} \cdot \|\exp(Ax)\|_1 \\ &= (\|\exp(Ax)\|_1)^{-1} \cdot \|\exp(Ax)\|_1 \\ &= 1, \end{aligned}$$

where the first step follows from the definition of $f(x)$ (see Definition 2.2.3), the second step follows from $\langle \exp(Ax), \mathbf{1}_n \rangle^{-1} \in \mathbb{R}$, the third step follows from the definition of the inner product, the fourth step follows from simple algebra, the fifth step follows from the definition of the ℓ_1 norm, and the last step follows from simple algebra. □

After filling this gap, we can use another lemma from [DLS23].

Lemma 2.2.5. *Let $\|f(x)\|_1 = 1$ (see Fact 2.2.4). Let $B(x) \in \mathbb{R}^{n \times n}$ be defined as in*

Section 2.2.1. Then, we have

$$-4I_n \preceq B(x) \preceq 8I_n$$

Recall that from Eq. (2.3), we have

$$\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2} = A^\top (B(x) + W^2)A,$$

and from Lemma 2.2.5, we have the bound for $B(x)$. Therefore, to study the positive definiteness of $\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2}$, it suffices to find the bound of W^2.

Note that the diagonal matrix $W = \mathrm{diag}(w)$ comes from the regularized term from Definition 1.0.4, so when we construct this matrix, we assume that all of its diagonal entries w_i^2 satisfy

$$w_i^2 \geq 4 + l/\sigma_{\min}(A)^2, \tag{2.4}$$

for all $i \in [n]$ and $l > 0$.

Therefore, we have

$$\begin{aligned}
\frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2} &= A^\top (B(x) + W^2)A \\
&\succeq A^\top (-4I_n + (4 + l/\sigma_{\min}(A)^2)^2 I_n)A \\
&= A^\top ((-4 + (4 + l/\sigma_{\min}(A)^2)^2) I_n)A \\
&\succeq A^\top (\frac{l}{\sigma_{\min}(A)^2} I_n)A,
\end{aligned} \tag{2.5}$$

where the first step follows from Eq. (2.3), the second step follows from Lemma 2.2.5 and Eq. (2.4), the third step follows from the distributive law, and the last step follows from $(a+b)^2 = a^2 + b^2 + 2ab \geq a^2 + b^2$, for all $a, b > 0$.

Because $\frac{l}{\sigma_{\min}(A)^2} > 0$, we can get that $\frac{l}{\sigma_{\min}(A)^2} I_n$ is positive definite. Therefore, combining this with Eq. (2.5), we can get that the Hessian matrix $H(x) = \frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2}$ is

positive definite.

2.2.3 Hessian is Lipschitz

After showing that the Hessian matrix is positive definite, now we present the techniques of showing the Hessian function is Lipschitz.

Recall that

$$H(x) := \frac{\mathrm{d}^2 L_{\mathrm{soft}}}{\mathrm{d}x^2} = \frac{\mathrm{d}^2 L_{\exp}}{\mathrm{d}x^2} + \frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} = A^\top (B(x) + W^2) A, \tag{2.6}$$

where

$$\begin{aligned} B(x) = & \underbrace{\langle 3f(x) - 2b, f(x) \rangle}_{\text{scalar}} \cdot \underbrace{f(x)}_{n \times 1} \underbrace{f(x)^\top}_{1 \times n} + \underbrace{\langle f(x) - b, f(x) \rangle}_{\text{scalar}} \cdot \underbrace{\operatorname{diag}(f(x))}_{n \times n \text{ diagonal matrix}} \\ & + \underbrace{\operatorname{diag}((2f(x) - b) \circ f(x))}_{n \times n \text{ diagonal matrix}} + \underbrace{(b \circ f(x))}_{n \times 1} \cdot \underbrace{f(x)^\top}_{1 \times n} + \underbrace{f(x)}_{n \times 1} \cdot \underbrace{(b \circ f(x))^\top}_{1 \times n}. \end{aligned}$$

The goal is to bound $\|H(x) - H(y)\|$ in terms of $\|x - y\|_2$. First, we define the following equations in order to express $\|H(x) - H(y)\|$.

- $G_1 := \|f(x)\|_2^2 f(x) f(x)^\top - \|f(y)\|_2^2 f(y) f(y)^\top$
- $G_2 := \langle f(x), b \rangle f(x) f(x)^\top - \langle f(y), b \rangle f(y) f(y)^\top$
- $G_3 := \langle f(x), f(x) \rangle \operatorname{diag}(f(x)) - \langle f(y), f(y) \rangle \operatorname{diag}(f(y))$
- $G_4 := \langle f(x), b \rangle \operatorname{diag}(f(x)) - \langle f(y), b \rangle \operatorname{diag}(f(y))$
- $G_5 := \operatorname{diag}(f(x) \circ (f(x) - b)) - \operatorname{diag}(f(y) \circ (f(y) - b))$
- $G_6 := \operatorname{diag}(f(x) \circ f(x)) - \operatorname{diag}(f(y) \circ f(y))$
- $G_7 := f(x)(f(x) \circ b)^\top - f(y)(f(y) \circ b)^\top$
- $G_8 := (f(x) \circ b) f(x)^\top - (f(y) \circ b) f(y)^\top$

The sum of $G_1, G_2, \ldots, G_8$ is equal to $B(x) - B(y)$ which is a crucial component of $H(x) - H(y)$ (see Eq. (2.6)). Therefore, we can bound these $G_1, G_2, \ldots, G_8$ under the spectral norm $\|\cdot\|$. Then, using the triangle inequality, we can show that

$$\|B(x) - B(y)\| \leq 2\|G_1\| + \|G_2\| + \cdots + \|G_8\|.$$

Specifically, we can show that

$$\begin{aligned}
\|H(x) - H(y)\| &= \|A^\top (B(x) + W^2)A - A^\top (B(y) + W^2)A\| \\
&= \|A^\top (B(x) - B(y))A\| \\
&\leq \|A\| \cdot (2\|G_1\| + \|G_2\| + \cdots + \|G_8\|)\|A\| \\
&\leq R^2 \cdot (2\|G_1\| + \|G_2\| + \cdots + \|G_8\|) \\
&\leq R^2 \cdot 100R \cdot \|f(x) - f(y)\|_2 \\
&\leq R^2 \cdot 100R \cdot \beta^{-2} n^{1.5} \exp(3R^2)\|x - y\|_2 \\
&\leq \beta^{-2} n^{1.5} \exp(20R^2)\|x - y\|_2,
\end{aligned}$$

where the first step follows from the definition of $H(x)$ (see Eq. (2.6)), the second step follows from the distributive law, the third step follows from the definitions of $G_1, \ldots, G_8$, the fourth step follows from the assumption that $\|A\| \leq R$, the fifth step follows from that [DLS23] bound each of the spectral norm of $G_1, \ldots, G_8$[3], the sixth step follows from the bound of $\|f(x) - f(y)\|_2$ in terms of $\|x - y\|_2$ as presented in [DLS23], and the last step follows from simple algebra.

Since bounding each of $G_1, \ldots, G_8$ requires a large amount of computation using similar techniques, because of the space limit, we present an example of bounding

[3]An example of bounding the spectral norm of G_7 is given. The techniques used for bounding other spectral norms are similar.

$\|G_7\|$. To bound $\|G_7\|$, we can express $\|G_7\|$ as $\|G_{7,1} + G_{7,2}\|$, where

$$G_{7,1} := f(x)(f(x) \circ b)^\top - f(x)(f(y) \circ b)^\top \tag{2.7}$$

$$G_{7,2} := f(x)(f(y) \circ b)^\top - f(y)(f(y) \circ b)^\top \tag{2.8}$$

We can bound $G_{7,1}$ by

$$\begin{aligned} \|G_{7,1}\| &= \|f(x)(f(x) \circ b)^\top - f(x)(f(y) \circ b)^\top\| \\ &= \|f(x)((f(x) - f(y)) \circ b)^\top\| \\ &\leq \|f(x)\|_2 \|(f(x) - f(y)) \circ b\|_2 \\ &\leq \|f(x) - f(y)\|_2 \|b\|_2, \end{aligned}$$

where the first step follows from Eq. (2.7), the second step follows from the distributive law, the third step follows from $\|y \cdot x^\top\| \leq \|y\|_2 \cdot \|x\|_2$, and the last step follows from $\|f(x)\|_2 \leq \|f(x)\|_1 \leq 1$ and Fact 2.1.9.

Similarly, $\|G_{7,1}\|$ can also be bounded by $\|f(x) - f(y)\|_2 \|b\|_2$. Combining everything together, we have

$$\begin{aligned} \|G_7\| &= \|G_{7,1} + G_{7,2}\| \\ &\leq \|G_{7,1}\| + \|G_{7,2}\| \\ &= 2\|f(x) - f(y)\|_2 \cdot \|b\|_2, \end{aligned}$$

where the first step follows from the definition of G_7, the second step follows from the triangle inequality.

2.3 Main result

In this section, we present the main algorithm and main theorem of [DLS23], where the main algorithm is used for solving the softmax regression problem (Definition 1.0.4)

and the main theorem provides the theoretical guarantee for the algorithm.

Algorithm 1 Main algorithm of [DLS23].

procedure IterativeSoftmaxRegression($A \in \mathbb{R}^{n \times d}, b \in \mathbb{R}^n, w \in \mathbb{R}^n, \epsilon, \delta$) ▷ Theorem 2.3.1
  We choose x_0 (suppose it satisfies Definition 2.1.11)
  We use $T \leftarrow \log(\|x_0 - x^*\|_2/\epsilon)$ to denote the number of iterations.
  for $t = 0 \to T$ **do**
    $D \leftarrow B_{\mathrm{diag}}(x_t) + \mathrm{diag}(w \circ w)$
    $\widetilde{D} \leftarrow \textsc{SubSample}(D, A, \epsilon_1 = \Theta(1), \delta_1 = \delta/T)$ ▷ Lemma 2.1.16
    $g \leftarrow A^\top (f(x_t)\langle c(x_t), f(x_t)\rangle + \mathrm{diag}(f(x_t))c(x_t))$
    $\widetilde{H} \leftarrow A^\top \widetilde{D} A$
    $x_{t+1} \leftarrow x_t + \widetilde{H}^{-1} g$
  end for
  $\widetilde{x} \leftarrow x_{T+1}$
  return $\widetilde{x}$
end procedure

Theorem 2.3.1 (Main Result, Theorem 1.5 in [DLS23])**.** *Let $\epsilon, \delta \in (0, 0.1)$. Let x_0 be an initial point satisfying Definition 2.1.11. Let ω denote the exponent of matrix multiplication, namely $\omega \approx 2.37$.*

Let x^* to denote the optimal solution of $L_{\mathrm{soft}}(x)$, namely

- $\nabla L_{\mathrm{soft}}(x^*) = \mathbf{0}_d$.
- $\|x^*\|_2 \leq R$,

for some $R > 0$. Suppose that $\|A\| \leq R$, $\|b\|_1 \leq 1$, $w_i^2 \geq 100 + l/\sigma_{\min}(A)^2$ for all $i \in [n]$, $M = n^{1.5}\exp(30R^2)$, and $l > 0$.

Then, there exists a randomized algorithm (Algorithm 1) that runs $\log(\|x_0 - x^*\|_2/\epsilon)$ iterations spend $O((\mathrm{nnz}(A) + d^\omega) \cdot \mathrm{poly}(\log(n/\delta))$ time per iteration, and finally outputs a vector $\widetilde{x} \in \mathbb{R}^d$ such that

$$\Pr[\|\widetilde{x} - x^*\|_2 \leq \epsilon] \geq 1 - \delta.$$

Chapter 3

Conclusions

In this thesis, we have presented a detailed analysis of the softmax regression problem from [DLS23], which is a key component in the attention mechanism used in LLMs. We first decomposed the Hessian of the softmax regression problem into constituent terms, providing a structured way to analyze its properties. Then, we show that the Hessian of the softmax regression problem is positive definite under certain conditions on the regularization term. This is an important property that enables the use of efficient optimization methods. We established that the Hessian function is Lipschitz continuous, which is a crucial ingredient for the theoretical guarantees of the proposed algorithm. Leveraging the positive definiteness and Lipschitz continuity of the Hessian, an efficient randomized algorithm was developed that can solve the softmax regression problem in logarithmic time with respect to the desired accuracy, while maintaining a high probability of returning a solution close to the optimal. The techniques and results presented in this thesis contribute to the broader understanding and efficient computation of attention mechanisms, which are critical for enabling LLMs to handle longer input sequences and extract meanings more effectively. The findings have important implications for improving the scalability and efficiency of large-scale language models, a rapidly growing area of AI research and development.

References

[Ans00] Kurt M Anstreicher. The volumetric barrier for semidefinite programming. *Mathematics of Operations Research*, 2000.

[AW21] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539. SIAM, 2021.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[BPC20] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[Cha22] ChatGPT. Optimizing language models for dialogue. *OpenAI Blog*, November 2022.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[DLS23] Yichuan Deng, Zhihang Li, and Zhao Song. Attention scheme inspired softmax regression. *arXiv preprint arXiv:2304.10411*, 2023.

[DLZ+23] Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Cheng. Puma: Secure inference of llama-7b in five minutes. *arXiv preprint arXiv:2307.12533*, 2023.

[DSW22] Yichuan Deng, Zhao Song, and Omri Weinstein. Discrepancy minimization in input-sparsity time. *arXiv preprint arXiv:2210.12468*, 2022.

[HJS+22] Baihe Huang, Shunhua Jiang, Zhao Song, Runzhou Tao, and Ruizhe Zhang. Solving sdp faster: A robust ipm framework and efficient implementation. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 233–244. IEEE, 2022.

[HWL21] Weihua He, Yongyun Wu, and Xiaohua Li. Attention mechanism for neural machine translation: A survey. In *2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 5, pages 1485–1489. IEEE, 2021.

[JKL+20] Haotian Jiang, Tarun Kathuria, Yin Tat Lee, Swati Padmanabhan, and Zhao Song. A faster interior point method for semidefinite programming. In *2020 IEEE 61st annual symposium on foundations of computer science (FOCS)*, pages 910–918. IEEE, 2020.

[KKL20] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[LG14] François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303, 2014.

[LLR23] Yuchen Li, Yuanzhi Li, and Andrej Risteski. How do transformers learn topic structure: Towards a mechanistic understanding. *arXiv preprint arXiv:2303.04245*, 2023.

[LSZ23] Zhihang Li, Zhao Song, and Tianyi Zhou. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint arXiv:2303.15725*, 2023.

[MMS+19] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suarez, Yoann Dupont, Laurent Romary, Eric Villemonte de La Clergerie, Djame Seddah, and Benoit Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

[Ope23] OpenAI. Gpt-4 technical report, 2023.

[RNS+18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. ., 2018.

[RPJL19] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

[RWC+19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[SYYZ22] Zhao Song, Xin Yang, Yuanyuan Yang, and Tianyi Zhou. Faster algorithm for structured john ellipsoid computation. *arXiv preprint arXiv:2211.14407*, 2022.

[SZZ21] Zhao Song, Lichen Zhang, and Ruizhe Zhang. Training multi-layer over-parametrized neural network in subquadratic time. *arXiv preprint arXiv:2112.07628*, 2021.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[TMS+23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[UAS+20] Mohd Usama, Belal Ahmad, Enmin Song, M Shamim Hossain, Mubarak Alrashoud, and Ghulam Muhammad. Attention-based sentiment analysis using convolutional and recurrent neural network. *Future Generation Computer Systems*, 113:571–578, 2020.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Wil12] Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 887–898, 2012.

[ZGD+20] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

[ZRG+22] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[ZWZ19] Xingxing Zhang, Furu Wei, and Ming Zhou. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv preprint arXiv:1905.06566*, 2019.

CURRICULUM VITAE

Junze Yin

Education

Boston University, MA
Master of Arts in Mathematics *Expected May 2024*
Bachelor of Arts in Mathematics, Philosophy and Religion *Expected May 2024*
Cumulative GPA: 3.75/4.00

Publications

1. Zhao Song, **Junze Yin**, Lichen Zhang, and Ruizhe Zhang. "Fast Dynamic Sampling for Determinantal Point Processes." In the 27th International Conference on Artificial Intelligence and Statistics (AISTATS 2024).

2. Zhao Song, **Junze Yin**, and Lichen Zhang. "Solving attention kernel regression problem via pre-conditioner." In the 27th International Conference on Artificial Intelligence and Statistics (AISTATS 2024).

3. Yuzhou Gu, Zhao Song, **Junze Yin**, and Lichen Zhang. "Low rank matrix completion via robust alternating minimization in nearly linear time." In the Twelfth International Conference on Learning Representations (ICLR 2024).

4. **Junze Yin**. "Dynamical fractal: Theory and case study." Chaos, Solitons & Fractals 176 (2023): 114190.

5. Zhao Song, Mingquan Ye, **Junze Yin**, and Lichen Zhang. "A Nearly-Optimal Bound for Fast Regression with ℓ_∞ Guarantee." In the Fortieth International Conference on Machine Learning (ICML 2023), pp. 32463-32482. PMLR, 2023.

6. **Junze Yin**, Jiale Zhang, and Ying Chen. "Analysis of Research and Trends in Online Review: A Perspective." In 2021 International Symposium on Artificial Intelligence and its Application on Media (ISAIAM), pp. 137-141. IEEE, 2021.

Work Experiences

1. *Peer Tutor, Boston University* *Spring 2021 - Present*

2. *Data Scientist Internship, IDG Capital* *Summer 2023*