Software Requirements Specification

for

Self-Attention Mechanism of ChatGPT

Version 1.0.2 approved

Prepared by Self-Attention Mechanism of ChatGPT Team at CSULA

Dr. Russell Abbott

Created March 24, 2023

**Table of Contents**

Table of Contents................................................................................................................. pg 2

Revision History................................................................................................................... pg 3

 1. Introduction................................................................................................................ pg 4

 1.1. Purpose........................................................................................................... pg 4

 1.2. Intended Audience and Reading Suggestions................................................ pg 4

 1.3. Project Scope................................................................................................ pg 4

 1.4. Definitions, Acronyms, and Abbreviations .................................................. pg 5

 1.5. References...................................................................................................... pg 5

 2. Overall Description.................................................................................................... pg 6

 2.1. System Analysis…......................................................................................... pg 6

 2.2. Operating Environment.................................................................................. pg 6

 2.3. Design and Implementation Constraints........................................................ pg 6

 2.4. User Documentation...................................................................................... pg 7

 2.5. Assumptions and Dependencies.................................................................... pg 7

 3. External Interface Requirements............................................................................... pg 8

 3.1. User Interfaces............................................................................................... pg 8

 3.2. Hardware Interfaces....................................................................................... pg 8

 3.3. Software Interfaces........................................................................................ pg 8

 4. Requirements Specification....................................................................................... pg 9

 4.1. Functional Requirements............................................................................... pg 9

 4.2. Design Constraints......................................................................................... pg 9

 5. Other Nonfunctional Requirements........................................................................... pg 11

 5.1. Performance Requirements............................................................................ pg 11

 5.2. Safety Requirements ................................................................................... pg 11

 5.3. Security Requirements ................................................................................... pg 11

Appendix A: Glossary........................................................................................................ pg 12

Appendix B: Analysis Models........................................................................................... pg 13

Revision History

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Initial | 03/24/23 | creation of document | 1.0.0 |
| Modification for premise | 04/02/23 | layout modification for table of contents and relevant pages | 1.0.1 |
| Midway update | 04/10/23 | updating purpose related sections | 1.0.2 |
|  |  |  |  |

**1. Introduction**

This Software Requirements Specification (SRS) document describes the requirements for the miniGPT tutorial, which is a practical guide for building a simple language model using the transformer architecture with PyTorch. This document outlines the purpose of the tutorial, its intended audience, product scope, definitions, acronyms, and references.

**1.1 Purpose**

The purpose of this SRS is to provide a clear and comprehensive understanding of the requirements for the miniGPT tutorial. This document outlines the functional and non-functional requirements, constraints, and assumptions of the tutorial to ensure that the product meets the expectations of the stakeholders.

**1.2 Intended Audience and Reading Suggestions**

This tutorial is intended for anyone interested in learning about natural language processing and building language models. The tutorial assumes some prior knowledge of programming in Python and familiarity with basic machine learning concepts. The intended audience includes students, researchers, developers, and anyone who wants to gain practical experience in building language models using the transformer architecture.

Reading suggestions for this SRS include the miniGPT tutorial itself, as well as relevant literature on natural language processing, PyTorch, and the transformer architecture.

**1.3 Product Scope**

The miniGPT tutorial provides a step-by-step guide for building a simple language model using the transformer architecture with PyTorch. The tutorial covers the following aspects of building a language model:

* Preprocessing the input text data
* Tokenizing the input text using the Hugging Face library
* Building the transformer-based language model architecture
* Training the language model on a large corpus of text data
* Evaluating the performance of the language model on text generation tasks

The tutorial does not cover advanced topics in natural language processing, such as semantic understanding or sentiment analysis.

**1.4 Definitions, Acronyms, and Abbreviations**

NLP: Natural Language Processing

SRS: Software Requirements Specification

PyTorch: An open-source machine learning library based on the Torch library

Hugging Face: A natural language processing library that provides state-of-the-art pre-trained models and tools for building custom models

**1.5 References**

The following references were consulted in the development of this SRS:

* "Attention is All You Need" by Vaswani et al. (2017)
* "The Illustrated Transformer" by Jay Alammar (2018)
* "PyTorch" documentation (2021)
* "Hugging Face" documentation (2021)

**2. Overall Description**

Based on the information presented in the Karpathy video, ChatGPT is a language model that uses deep learning techniques to generate human-like responses to text-based input. The model is trained on a vast corpus of text data, enabling it to generate coherent and contextually-relevant responses to a wide range of inputs.

To use ChatGPT, users simply need to input text prompts, and the model generates a response based on the input. The model can be fine-tuned for specific use cases, such as customer service chatbots or language translation services, by adjusting the training data and hyperparameters. Overall, ChatGPT represents a powerful tool for natural language processing that has the potential to revolutionize the way we communicate with machines.

**2.1 System Analysis**

Based on the information presented in the Karpathy video, ChatGPT is a language model that uses deep learning techniques to generate human-like responses to text-based input. Although we did not build the model ourselves, our research involved analyzing the core features of ChatGPT and its potential applications.

One of the major technical hurdles associated with developing a language model like ChatGPT is the need for vast amounts of high-quality training data to ensure the model's accuracy and effectiveness. Additionally, optimizing the model's architecture and hyperparameters is crucial for achieving desired performance levels. As the model is designed to generate responses in natural language, it must also be able to handle nuances such as slang and regional variations in language.

While we did not directly address these technical hurdles ourselves, our research has provided valuable insights into the potential applications and challenges associated with natural language processing. By analyzing the core features of ChatGPT and the technical requirements for developing such models, we hope to contribute to the ongoing development of this exciting field.

**2.2 Operating Environment**

The Learning Mechanism of ChatGPT code was tested and trained on Google Collaboratory (Google Collabs). Google Collabs allowed the team to leverage Google’s computational resources available to the general public without the need to set up local environments in every team member’s machines.

**2.3 Design and Implementation Constraints**

The Learning Mechanism of ChatGPT design and implementation constraints were set by Andrej Karpathy, the author of the code this team analyzed. The code this project researches on is the Self-Attention mechanism portion of Karpathy’s version of ChatGPT. Due to this being a portion of the code, altering it may have resulted in changes that would affect the entirety of the project. Additionally, the code for this section is generally as-is, since it follows the self-attention architecture.

**2.4 User Documentation**

The user documentation provided with Karpathy's miniGPT tutorial includes a detailed step-by-step explanation of the code implementation, as well as explanations of the key design decisions and trade-offs that were made during the development process. The tutorial also provides guidance on how to modify and extend the code to experiment with different hyperparameters, neural network architectures, and training techniques. In addition, the tutorial includes links to relevant research papers, PyTorch documentation, and other online resources that may be helpful for users who want to dive deeper into the topic.

**2.5 Assumptions and Dependencies**

* It is assumed that the user has a basic understanding of Python programming language and machine learning concepts.
* The tutorial requires the use of Python and PyTorch libraries, which must be installed on the user's system prior to following the tutorial.
* The tutorial assumes that the user has access to a machine with sufficient computational resources, including a CPU and GPU, for training the language model.
* The performance of the language model depends on the quality and size of the training corpus used.
* The tutorial assumes that the user has access to a large corpus of text data for training the language model or has the ability to acquire one.
* The tutorial assumes that the user has access to a text editor or IDE for editing and running Python code.
* The tutorial assumes that the user has a basic understanding of natural language processing and neural network architecture concepts, although these are briefly explained in the tutorial.

**3. External Interface Requirements**

There are no requirements to run our project as it is stored in Google Collab. Meaning al the computational power and storage is handled by Google. It should be noted that if one wants to run this program locally they can but would require significant local computing power as training the dataset would be longer when using a local machine.

**3.1 User Interfaces**

 As of right now we don't have a GUI or user input The input for our mini GPT is in a text file that will be used for training. The output is displayed in a console with no GUI.

**3.2 Hardware Interfaces**

No external hardware or interfaces will be needed nor documented.

**3.3 Software Interfaces**

No external software products or interfaces will be needed nor documented.

**4. Requirements Specification**

4.0.1 The system shall take Shakespearean text from https://raw.githubusercontent.com/karpathy/char-rnn/master/data/tinyshakespeare/input.txt as input.

4.0.2 The system may take input text of another genre or time period for generating within a different topic.

4.0.3 The system shall build a neural network model, training and optimizing it with the input, and output generated text based on the style of Shakespeare.

**4.1 Functional Requirements**

4.1.1 The system shall check whether the machine makes use of a cuda through a torch function and an if statement.

4.1.2 The system shall allow users to verify the input has been garnered correctly through outputting its length and first portion.

4.1.3 The system shall convert the characters in input text to corresponding numbers, many later outputs (through print statements) being based on this version of the text.

4.1.4 The system shall employ parameters relating to the neural network that affect its speed and accuracy and can be changed for different outcomes.

4.1.5 The system shall continue implementing print statements throughout the code, making it easier for users to check steps are running smoothly.

4.1.6 The system shall convert numbers generated by the neural network back to characters to output the final result of the program.

**4.2 Design Constraints**

The following design constraints are applicable to this tutorial:

1. The tutorial assumes that the reader has a basic understanding of programming concepts and is comfortable with the Python programming language.
2. The tutorial is designed to be platform-independent and should be compatible with most modern operating systems.
3. The tutorial requires access to a computer with a modern CPU and sufficient memory to run the code examples.
4. The tutorial relies on external libraries such as PyTorch and Hugging Face Transformers. The versions of these libraries used in the tutorial are specified in the requirements.txt file.
5. The tutorial assumes that the user has access to a stable internet connection for downloading the required libraries and datasets.
6. The tutorial is designed to be a high-level overview of the transformer-based language model architecture and is not intended to be a comprehensive guide to the field of natural language processing or machine learning.
7. The tutorial is subject to change and may be updated in the future as new technologies and techniques are developed.

**5. Other Nonfunctional Requirements**

**5.1 Performance Requirements**

5.1.1 A text file with some of Shakespeare’s work is provided in Andrej Karpathy’s version of ChatGPT.

5.1.2. Another text file could be used in this product such as a text file with some of Dr. Suess’s work in order to output text like in Dr. Suess’s books.

**5.2 Safety Requirements**

5.2.1 While using Andrej Karpathy’s version of ChatGPT, just take into account the text files to download to use in the mechanism he designed. Make sure to download a text file that is secured or from a secured website.

**5.3 Security Requirements**

5.3.1 For this product, there are no user identification requirements as anyone can look to his resources on his GitHub or Youtube page.

**Appendix A: Glossary**

1. **K**: Key Matrix
2. **V**: Value Matrix
3. **Q**: Query Matrix
4. **wei**: Weight Matrix
5. **Normalization**: The process in which we make take values that have scale and uniform them so that the data is readable
6. **Softmax**: Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector
7. **ChatGPT**: ChatGPT is an artificial intelligence language model developed by OpenAI. It's based on the transformer architecture, specifically the GPT (Generative Pretrained Transformer) series, which as of my knowledge cutoff in September 2021, was up to GPT-4.

**Appendix B: Analysis Models**

**Self Attention Example**





**Matrix Multiplication Example**