

Chetan Sharan

Data Scientist|Engineer|Private Pilot

IIT KanPur

An Artificial Intelligence Model for Natural Language Processing |A Case Study

# Table Of Content

[1.0 Understanding Problem Statement 2](#_Toc173254264)

[2.0 Literature Survey 3](#_Toc173254265)

[3.0 Methodology 3](#_Toc173254266)

[3.1 Preparing Data 3](#_Toc173254268)

[3.2 Setting up the Environment 4](#_Toc173254269)

[3.3 Model Selection 5](#_Toc173254270)

[3.4 Fine-Tuning the Model 5](#_Toc173254271)

[3.5 Evaluation and Testing 5](#_Toc173254272)

[4.0 Conclusion 5](#_Toc173254273)

# Understanding Problem Statement

***Problem Statement:*** *Develop a state-of-the-art question-answering model leveraging the Quora Question Answer Dataset. The objective is to create an AI system capable of understanding and generating accurate responses to a variety of user queries, mimicking a human-like interaction.*

The Date set is from Quora, it contains around 56 thousand questions answers arranged in two columns. One row for each question & answer. This is mostly text data.

# Literature Survey

Before setting up the methodology of the project, it was necessary to do some literature survey. I started with ***Bag of words*** through ***RNN*** finally to ***Transformers.*** Followed by popular example on NLP that are available.

# Methodology

## 

Figure 1: The Methodology.

## 3.1 Preparing Data

**Accessing Data**: The data is hosted at hugging face & its easily accessible using Pandas or Datasets (hugging face) libraries.

from datasets import load\_dataset

ds = load\_dataset("toughdata/quora-question-answer-dataset")

Since this is a decently sized text data, a methodology is following step by step to preprocess the data for NLP models. The process is shown in the following figure.

Figure 2: Data Preparation.

**Data Cleaning**: Remove any unwanted text, handle inconsistencies, and ensure the data is in a clean, usable format. Regular Expressions Library is used to make the data clean.

**Tokenization**: Convert the text data into tokens that the model can understand.

**Stop Word Removal:** This step is left on purpose as I have selected high processing GPU & CPU from AWS server.

**Lemmatization:** For this project, lemmatization has been selected over stemming as we are dealing with text data of mostly English language. Also, I have procured significant GPU capacity from AWS.

## 3.2 Setting up the Environment

**Hardware:** Since I have a Dell G series laptop with basic GPU configuration, I could not use this machine for NLP process. I have selected AWS as the cloud service because of its good customer support & familiarity with the system. One Instance of ***p3.2xlarge*** with 16 GiB VRAM was used as shown in fig. 3.

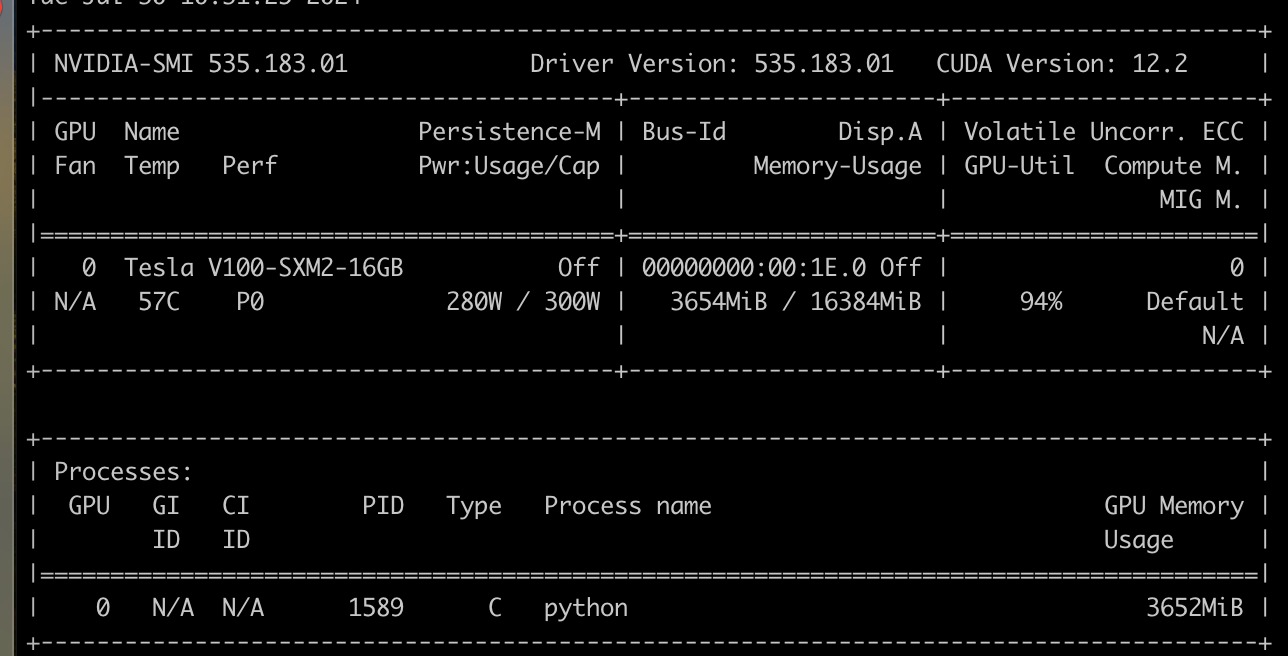


Figure 3: AWS Instances

**Libraries:** Installed necessary libraries like Hugging Face’s transformers, nltk, .

## 3.3 Model Selection

**Choosing a Base Model**: I started finetuning GPT Start with a pre-trained GPT model (like GPT-2 or GPT-3) from Hugging Face or OpenAI. Fine-tuning a pre-trained model is much more efficient than training from scratch.

**Customizing the Model**: Depending on your needs, you might want to customize the architecture or parameters.

## 3.4 Fine-Tuning the Model

**Loading the Pre-trained Model**: Use libraries like Hugging Face’s transformers to load a pre-trained model.

**Preparing the Data Loader**: Create data loaders that feed your tokenized data into the model.

**Training Loop**: Implement the training loop. This involves defining the loss function, optimizer, and training the model on your data for several epochs.

**Monitoring**: Monitor the training process, check for overfitting, and adjust hyperparameters as needed.

## 3.5 Evaluation and Testing

**Evaluation Metrics**: Define metrics to evaluate the model’s performance on your specific tasks.

**Testing**: Test the model on a separate validation set to ensure it generalizes well.

# Conclusion