Analysis and Implementation of a Transformer Model for Translation

---

Your Name Here

Supervisor(s):

A research proposal submitted in partial fulfillment of the requirements for the degree of Degree Name Here

in the

Name of School or Department Here
University of the Witwatersrand, Johannesburg

25 May 2024

Your Name Here

25 May 2024

Contents

List of Figures

0.1 Introduction

This section provides a detailed explanation of the Python code used for training a Transformer model for language translation. The code utilizes PyTorch and other auxiliary libraries for handling datasets, preprocessing, and training procedures.

0.2 Code Overview

The script is structured to perform dataset loading, preprocessing, model training, validation, and saving checkpoints. Key functionalities are encapsulated in functions and executed conditionally.

0.3 Detailed Code Explanation

0.3.1 Configuration and Setup

```
from torch.utils.data import DataLoader
from dataset import BilingualDataset
from model import build_transformer
from config import get_config
import torch
from pathlib import Path
from datasets import load_dataset
```

Libraries and modules necessary for dataset handling, model operations, and configurations are imported. The script checks for GPU availability and sets the device accordingly for training.

0.3.2 Data Preparation

Load and Preprocess Function

```
def load_and_preprocess(config):
    ds_raw = load_dataset(f"{config['datasource']}", f"{config['
    lang_src']}-{config['lang_tgt']}", split='train')
    tokenizer_src = get_or_build_tokenizer(config, ds_raw, config['
    lang_src'])
```

```
    ...
    return train_dataloader, val_dataloader, tokenizer_src,
tokenizer_tgt
```

This function handles the loading of the dataset and its preprocessing. It initializes tokenizers for both the source and target languages, splits the dataset, and prepares data loaders.

0.3.3 Model Training

Training Function