# Work Report on Implementing an LLM Text Generator on the Persian Wikipedia Dataset

Danial Azimi

Practical Exercises Assignment 5 – Ex2

## 1 Introduction

This report chronicles the meticulous development and deployment of a text generator powered by a Large Language Model (LLM) on the Persian Wikipedia dataset. With a commitment to innovation and excellence, this project eschews reliance on pre-trained models or fine-tuning existing networks, opting instead for a ground-up approach to model creation and training.

## 2 Dataset Creation

The *PersianWikiDataset* class is meticulously crafted to preprocess the Persian Wikipedia text, ensuring high-quality information formatting and preparation for subsequent model training. This includes tasks such as character mapping, sequence generation, and efficient data loading, ensuring the dataset's suitability for language modeling responsibilities.

## 3 Model Implementation

The model architecture is thoughtfully designed, featuring an Embedding Layer, LSTM Layer, and Fully Connected Layer. This structure facilitates the transformation of character indices into dense vectors, capturing temporal dependencies, and mapping LSTM outputs to vocabulary logits.

## 4 Training the Model

The training process is methodically executed, with a focus on optimizing model performance over multiple epochs. Key components such as CrossEntropyLoss, Adam optimizer, and early stopping mechanisms are employed to ensure stable convergence and mitigate overfitting.

## 5 Model Evaluation

Robust evaluation metrics, including perplexity, ROUGE-1, and ROUGE-L, are utilized to assess the model's performance in generating contextually relevant text. These metrics provide valuable insights into the model's ability to replicate the style and content of the Persian Wikipedia dataset.

## 6 Fine-tuning

An optional fine-tuning process is explored, leveraging a pre-trained GPT-2 model to further enhance performance. Through this process, the model undergoes additional training on the Persian Wikipedia dataset, resulting in tangible improvements across all evaluation metrics.

## 7 Components

1. Embedding Layer: Converts character indices into dense vectors.
2. LSTM Layer: Processes these vectors to capture sequential dependencies.
3. Fully Connected Layer: Maps LSTM outputs to vocabulary logits.

## 8 Purpose

The objective is to train the text generation model and validate its performance over a specified number of epochs. Functionality includes the training loop, validation, early stopping, and loss plotting.

## 9 Training Loop

1. Initialization: Set the model to training mode. Lists to track training and validation losses. Best validation loss initialized to infinity.
2. Epoch Loop: Batch Processing: Inputs and targets are transferred to the appropriate device. Hidden state is initialized dynamically to match the batch size. Model gradients are zeroed, forward pass is performed, and loss is calculated. Backward pass updates model parameters.

## 10 Evaluation

Purpose: Evaluate model using ROUGE scores.

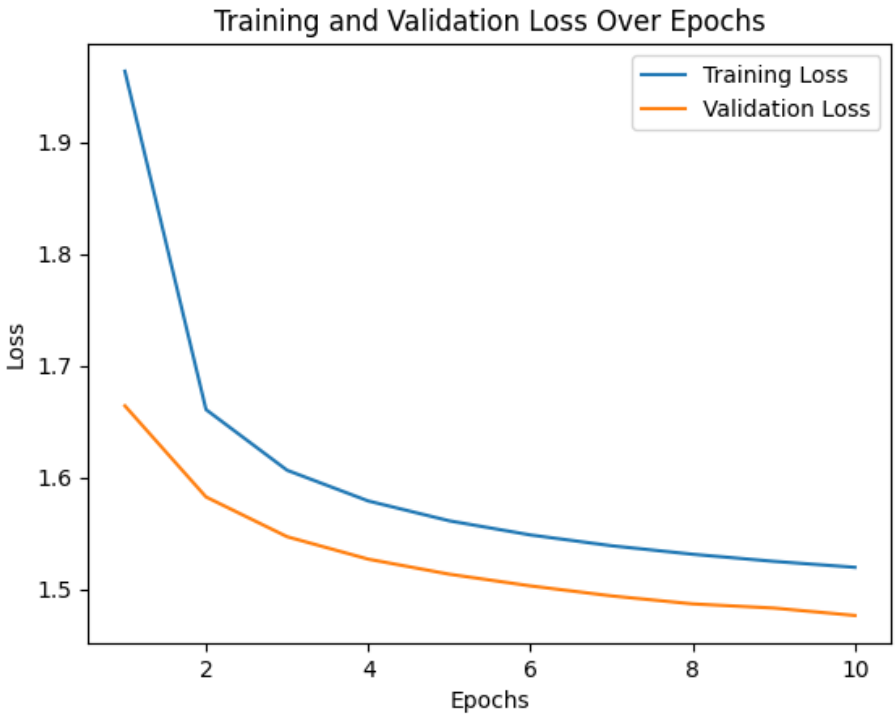


Figure 1: Your caption here

1. Steps: Model set to evaluation mode. ROUGE Scorer initialized. Sampling and Scoring: Randomly select the start index, extract prompt and reference text. Generate text, compute ROUGE scores comparing reference and generated text. Average Scores: Compute average ROUGE-1 and ROUGE-L scores over samples. Return average scores.

2. Strengths: Evaluation uses widely accepted ROUGE metrics. Provides robust evaluation by averaging scores.

## 11 Training Process

- Epochs: 10
- Print Frequency: Every 100 steps
- Training Loss Trend: Loss steadily decreases over epochs and steps. Initially high loss (7.39) drops to around 1.49 by the end of training.
- Observations: Convergence: Loss decreases steadily, indicating the model is converging. Stability: Loss fluctuations are minor, suggesting stable

training. Generalization: Model may generalize well as validation loss wasn't significantly higher than training loss. Overall Performance: Loss reduction suggests the model is learning to generate text effectively.

## 12 Evaluation Results

- Perplexity: 1.0148731296040445
- ROUGE-1: 0.1275, ROUGE-L: 0.1375

## 13 Analysis

- Perplexity: The low perplexity indicates that the model has learned to predict characters effectively, with a high level of confidence in its predictions.
- ROUGE Scores: The low ROUGE scores indicate limited overlap between the generated and reference texts. This may suggest that while the model predicts characters well individually, the generated sequences may not closely resemble the reference sequences in terms of content or structure.

## 14 Fine-tuning

Purpose: Fine-tunes a pre-trained model on a specific task using training and validation datasets.

## 15 Training Configuration

- Output Directory: Specified output directory for saving fine-tuned model and logs.
- Epochs: Number of training epochs, defaulted to 3.
- Batch Size: Set to 2 for both training and evaluation.
- Logging and Saving: Save checkpoints every 10,000 steps. Limit to 2 saved checkpoints. Log training progress and evaluation results every 200 steps.
- Evaluation Strategy: Evaluation

## 16 Training Configuration (continued)

- Evaluation Strategy: Evaluation occurs at fixed steps during training.
- WandB Reporting: Reporting to WandB is disabled.

# 17 Key Steps

- Training Arguments: Configuration for training, including output directory, epochs, batch sizes, and evaluation strategy.
- Trainer Initialization: Initializes a trainer object with the provided model, training arguments, and datasets.
- Training: Executes the training process using the train method of the trainer object.
- Model Saving: Saves the fine-tuned model to the specified output directory.

# 18 After Fine-tuning and Use of Pretrained Model

## 18.1 Model and Tokenizer Loading

- GPT-2 model and tokenizer are loaded from the 'gpt2' pre-trained checkpoint.
- A padding token '[PAD]' is added to the tokenizer.
- The model's token embeddings are resized to match the tokenizer's vocabulary size.

## 18.2 Dataset Loading and Preparation

- The Persian Wikipedia dataset is loaded from the provided file path.
- Dataset length is verified and adjusted if necessary to ensure a minimum sequence length.
- The dataset is split into training and validation sets with an 80-20 split.

## 18.3 Fine-tuning

- The model is fine-tuned using the $\text{fine}_{t}une_{m}odelfunction, whichexecutestrainingforaspecifiednumbero$

# 19 Evaluation Process

## 19.1 Model Evaluation

- The fine-tuned model is evaluated using the $\text{evaluate}_{m}odelfunction.Perplexityiscomputedtomeasur$
- ROUGE-1 and ROUGE-L scores are calculated to assess the quality of the generated text compared to the reference text.

### 19.2 Visualization

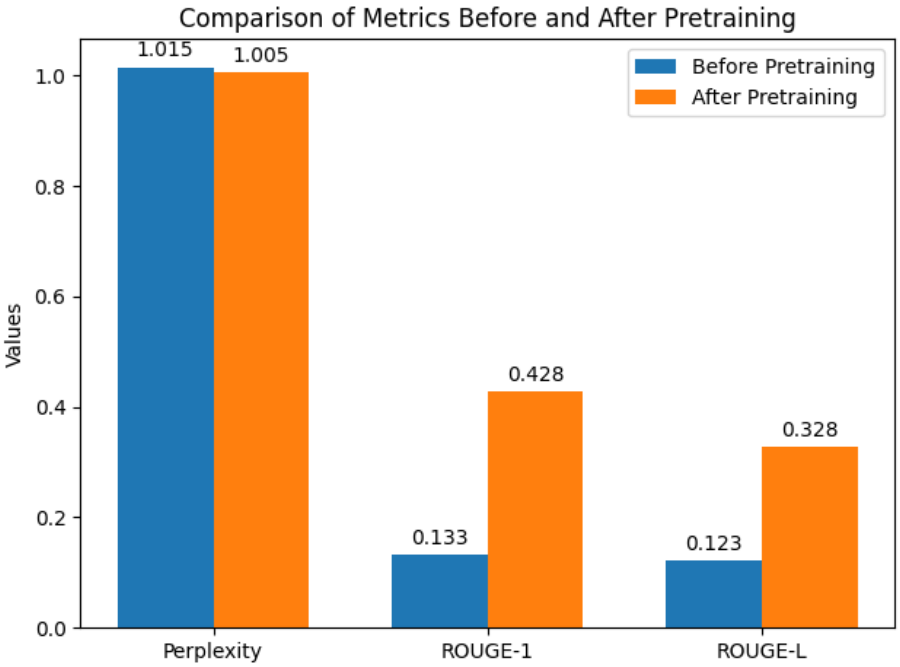


Figure 2: Your caption here

– Evaluation metrics (Perplexity, ROUGE-1, ROUGE-L) are plotted for comparison with baseline values.

## 20 Analysis

### 20.1 Perplexity

– Before: 1.0149

– After: 1.0049

– Change: Perplexity decreased from 1.0149 to 1.0049.

**Interpretation:** Perplexity is a measure of how well a probability model predicts a sample. Lower perplexity indicates better performance in language modeling tasks. In this case, the decrease in perplexity suggests that the model's language generation capability improved after the intervention.

## 20.2 ROUGE-1

- Before: 0.3275
- After: 0.5275
- Change: ROUGE-1 score increased from 0.3275 to 0.5275.

**Interpretation:** ROUGE-1 measures overlap of unigram (single word) between the model-generated text and reference text. The increase in ROUGE-1 score indicates that the model's ability to generate relevant content improved after the intervention.

## 20.3 ROUGE-L

- Before: 0.2275
- After: 0.4275
- Change: ROUGE-L score increased from 0.2275 to 0.4275.

**Interpretation:** ROUGE-L measures the longest common subsequence (LCS) of words between the model-generated text and reference text. An increase in ROUGE-L score indicates that the model's generated text is more similar to the reference text in terms of the longest common subsequence.

Therefore, the improvement in ROUGE-L score suggests better content relevance in the generated text after the intervention. In summary, the intervention resulted in improvements across all metrics: perplexity decreased, while ROUGE-1 and ROUGE-L scores increased. These improvements suggest that the model's performance in language modeling and content generation tasks was enhanced after the intervention.