2025 Fall - Artificial Neural Networks Final Project

Chinese-to-English Machine Translation (ZH→EN)

This project is the final term assignment for the "Artificial Neural Networks" course .

You are required to implement **two** different model architectures (RNN and Transformer) to complete a Chinese-to-English machine translation task. The models will be evaluated based on BLEU scores and code implementation details.

---

1. Project Overview

Task

Build a Machine Translation system to translate Chinese sentences into English using Deep Learning frameworks.

Required Architectures

1. **RNN-based Seq2Seq:**
   - Must use **GRU or LSTM** (2 layers, unidirectional) for Encoder and Decoder.
   - **Attention Mechanism** must be implemented independently (explore dot product, multiplicative, or additive alignment functions).
2. **Transformer:**
   - Complete the missing modules in the provided framework: `MultiHeadAttention`, `Scaled Dot-Product Attention`, `PositionalEncoding`, `EncoderLayer`, and `DecoderLayer`.
   - **Optional Challenge:** Explore Multi-Query Attention (MQA), Grouped-Query Attention (GQA), or Sparse Attention for extra marks.

Environment

- **Language:** Python
- **Framework:** PyTorch

---

2. Dataset & Preparation

Data Source

The dataset includes Small Training (100k), Large Training (10k), Validation (500), and Test (200) sets in `.jsonl` format.

- **Download Link:** Baidu Netdisk.
- **Note:** If resources are limited, you may use 10k samples from the Small Training Set, though using the Large Training Set is encouraged.

Data Preprocessing

Configure paths in `config.yaml`. The preprocessing pipeline should handle:

1. **Cleaning:** Filter invalid characters and truncate excessively long sentences.
2. **Tokenization:**
   - **Chinese:** Use Jieba or HanLP.
   - **English:** Use NLTK, BPE, or WordPiece.
3. **Vocabulary:** Build statistical vocabulary and filter low-frequency words.

Run the preprocessing script:

```
python preprocess.py -c config.yaml
```

---

3. Implementation Details

Directory Structure

```
.
├── check_translations.py     # Translation format validation script
├── config.yaml               # Global configuration (Data/Model/Training)
├── evaluate.py               # Inference and BLEU evaluation
├── preprocess.py             # Data cleaning and tokenization
├── train.py                  # Training entry point
├── utils.py                  # Utility functions
└── model
    └── transformer.py        # ★ Core file to be completed (TODO)
    └── rnn.py                # Core file that you need to create
    # Note: You may need to add files to support the RNN architecture requirement.
```

Core Tasks

Edit `model/transformer.py` to complete the following `# TODO` items:

Module	Requirement
`PositionalEncoding`	Implement positional encoding matrix calculation.
`MultiHeadAttention`	Implement Q/K/V projection, Scaled-Dot Product, and masking.
`EncoderLayer`	Implement Self-Attention + FFN + Residual Connection + LayerNorm.
`DecoderLayer`	Implement Masked Self-Attn, Cross-Attn, and FFN.

Add `model/rnn.py` (referring to the function signatures in transformer.py)

4. Training & Inference

4.1 Training

Train your model. You can adjust parameters in `config.yaml`.

```
python train.py -c config.yaml
```

Checkpoints will be saved to the `runs/` directory by default.

4.2 Evaluation

Evaluate the model using **Greedy decoding** (or implement alternative strategies). Performance is measured using **BLEU-4**.

```
python evaluate.py                 \
    -c config.yaml                 \
    --ckpt runs/best_model.pt      \
    --save_path translations.json
```

Output Format (`translations.json`):
The output must be a list of JSON objects containing Source, Reference, Hypothesis, and a SHA hash.

```
[
  {
    "src": "今天天气很好",
    "ref": "It is a fine day today",
    "hyp": "The weather is great today",
    "sha": "..."
  }
]
```

---

5. Submission Guidelines

Self-Check

Before submitting, verify your output format:

```
python check_translations.py translations.json
```

Project Report

A PDF report is required, covering:

1. Description of model architectures and implementation.
2. Experimental results analysis (including BLEU scores).
3. Visualization of attention weights (analyze representative cases).

File Packaging

Compress the following into `2025ANN-final-term-project-StudentID-Name.zip/rar` :

- **Source Code** (including completed `transformer.py` and RNN implementation).
- **Final Checkpoint** (`runs/best_model.pt`).
- **Output File** (`translations.json`).
- **Project Report** (PDF).