ANLP-FALL2025-HW1 Report

Andrew_id: peterwa2

Introduction

The baseline code used right-padded sequences without any attention mask. This meant that during inference, the model could attend to padded tokens, which introduced noise and degraded performance.

To address this, I implemented an attention mask mechanism that prevents the model from attending to padded tokens.

Implementation

I made the following major changes to enable attention mask:

run_llama.py

- **Change**: Added a new command-line argument `--use_attn_mask`

代码块

```
parser.add_argument("--use_attn_mask", action="store_true",
    help="Enable attention mask to ignore padding tokens")
```

- **Change**: Updated `test_with_prompting` to automatically rename output files when attention mask is enabled

代码块

```
if args.use_attn_mask:
    dataset = os.path.basename(args.dev).split("-")[0]
    args.dev_out = f"{dataset}-dev-advanced-output.txt"
    args.test_out = f"{dataset}-test-advanced-output.txt"
```

In training/evaluation loops, `attention_mask` from the dataloader is passed into the model:

代码块

```
logits = model(b_ids, attention_mask=batch['attention_mask'].to(device))
```

LlamaDataset class

Mask construction: Built a binary attention mask where `1` marks real tokens and `0` marks padding

代码块

```
attention_mask = [[1] * len(sentence) + [0] * (max_length_in_batch - len(sentence))
                      for sentence in encoding]
attention_mask = torch.LongTensor(attention_mask)
```

Returned inside `collate_fn`, so every batch includes:

代码块

```
batched_data = {
    'token_ids': token_ids,
    'labels': labels,
    'sents': sents,
    'attention_mask': attention_mask,
}
```

llama.py

- **Attention**: Added `attn_mask` argument, used to mask out padded positions in score computation.
- **LlamaLayer**: Forward now calls attention with `attn_mask`.
- **Llama.forward**: Passes the same `attention_mask` down through all layers.

代码块

```
# Attention
if attn_mask is not None:
    attn_scores = attn_scores.masked_fill(attn_mask == 0, float("-inf"))

# LlamaLayer
attn_out = self.attention(norm, attn_mask=attn_mask)

# Llama.forward
for layer in self.layers:
    h = layer(h, attn_mask=attention_mask)
```

Experimental Setup

I evaluated on the CFIMDB dataset for binary sentiment classification using zero-shot prompting with the same prompt template as the baseline. The model is a 42M-parameter pretrained Llama backbone, run with right-padded batching. We added a padded attention mask to exclude the pad tokens.

Result

Setting	Dev Accuracy	Test Accuracy
Baseline	0.490	0.109
Attention Mask	**0.498**	**0.756**

The attention mask leads to a +0.647 improvement in test accuracy, showing that masking padded tokens significantly improves inference quality in the zero-shot prompting setting.

Usage

Added a `--use_attn_mask` flag to the command line. When enabled, results are saved as `{dataset}-dev-advanced-output.txt` and `{dataset}-test-advanced-output.txt`

代码块

```
# Example: run zero-shot prompting on the CFIMDB dataset with attention mask enabled
python run_llama.py \
  --option prompt \
  --batch_size 10 \
  --train data/cfimdb-train.txt \
  --dev data/cfimdb-dev.txt \
  --test data/cfimdb-test.txt \
  --label-names data/cfimdb-label-mapping.json \
  --dev_out cfimdb-dev-prompting-output.txt \
  --test_out cfimdb-test-prompting-output.txt \
  --use_attn_mask
```