Improving minBERT with Conditional Layer Normalization

Abstract

The aim of the project is to enhance the performance of baseline BERT model using Conditional Layer Normalization (CLN), adopted from Conditional batch normalization, which adapts the normalization process based on the task at hand.

Introduction

Even though the BERT model has significantly advanced the state of the art in various NLP tasks , there is scope for improvement especially when applying BERT to a variety of tasks simultaneously as done in this project. Standard Layer Normalization uses fixed normalization parameters for all tasks limiting the model's flexibility and performance on task-specific nuances.

To address this limitation, we propose Conditional Layer Normalization(CLN) as an extension of the baseline BERT model. CLN uses task specific normalization parameters allowing the model to tailor its internal representations according to the specific tasks at hand, improving overall performance across tasks.

In this project, we implement CLN into the baseline BERT model and evaluate its effectiveness on the three tasks of sentiment classification, paraphrase detection, and semantic textual similarity. We hypothesize that the CLN-enhanced model will outperform the baseline BERT model due to its ability to use specific normalization parameters for each task separately.

Approach

The implementation has 2 steps : Additional pre-training on datasets specific to all 3 downstream tasks and the integration of CLN.

Additional Training

Additional training on the Quora and STS datasets was done to ensure task-specific weights could be learned.
This was implemented with the following steps:

- Load and preprocess the SST, Quora, and STS datasets.
- Create DataLoader objects for each dataset to facilitate batch processing.
- Initialize the MultitaskBERT model with the specified configuration.
- Set the learning rate and initialize the optimizer.

- Iterate over the specified number of epochs:
    - For each epoch, iterate over each dataset (SST, Quora, and STS).
    - For each batch in a dataset:
        - Compute the logits using the model's prediction method for the task
        - Compute the loss (cross-entropy for sentiment classification, binary cross-entropy for paraphrase detection, and MSE for semantic similarity).
        - Perform backpropagation and update the model parameters.

CLN Implementation

Conditional Layer Normalization (CLN) was implemented as follows:

- Define the ConditionalLayerNorm class inside bert.py, inheriting from nn.Module.
- Initialize parameters for default layer normalization: self.weight and self.bias.
- Introduce task-specific parameters as nn.Parameter with size (num_tasks, hidden_size): self.task_weight, self.task_bias
- Initialize self.task_weight and self.task_bias as 0 tensors.
- In the forward method:
    - Compute the mean and standard deviation of the input tensor.
    - Adjust the normalization parameters based on the task-specific weights and biases.
    - Add the task-specific weights and biases to the default weights and biases.
- Integrate ConditionalLayerNorm into the BertLayer class:
    - Replace the default nn.LayerNorm layers with ConditionalLayerNorm.
    - Modify the add_norm method to pass the task_id to the ConditionalLayerNorm layers.
    - Modify the forward method of BertLayer to correctly handle the task_id.
- Modify the BertModel class to support CLN:
    - Replace the default nn.LayerNorm for the embedding layer with ConditionalLayerNorm.
    - Update the embed method to include the task_id in the initial embedding layer normalization.
    - Update the encode method to pass the task_id to each BertLayer.
    - Update the forward method to accept and handle the task_id, passing it through to the embed and encode methods.

- Update the MultitaskBERT class to use the new BertModel with CLN integrated:
    - Modify the forward method of MultitaskBERT to pass the task_id to the BERT model.

Standard Layer Normalization

The standard Layer Normalization formula is defined as:

$$y = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot \gamma + \beta$$

[4]

Where:

- $x \in \mathbb{R}^d$ is the input tensor to the layer, and d is the dimensionality of the input.
- μ is the mean of x, computed as:

$$\mu = \frac{1}{d} \sum_{i=1}^{d} x_i$$

- σ^2 is the variance of x, computed as:

$$\sigma^2 = \frac{1}{d} \sum_{i=1}^{d} (x_i - \mu)^2$$

- ϵ is a small constant added to the variance to avoid division by zero.
- $\gamma \in \mathbb{R}^d$ and $\beta \in \mathbb{R}^d$ are the learnable weight and bias parameters, respectively.

Conditional Layer Normalization

CLN adapts the normalization parameters by introducing additional task-specific weight and bias parameters. This allows the model to adjust its normalization process based on the task at hand. The CLN formula is defined as:

$$y = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \cdot (\gamma + \gamma_t(t)) + (\beta + \beta_t(t)) \tag{4}$$

Where:

- x, μ, σ^2, and ϵ are defined as in standard Layer Normalization.
- $\gamma(t) \in \mathbb{R}^d$ and $\beta(t) \in \mathbb{R}^d$ are the task-specific weight and bias parameters.
- t represents the task identifier, indicating which task the current data belongs to.
- $\gamma + \gamma_t(t)$ and $\beta + \beta_t(t)$ are the combined weight and bias parameters that incorporate both the standard and task-specific parameters.

Datasets

- Stanford Sentiment treebank (SST) for sentiment classification
- Quora for paraphrase detection
- SemEval STS for similarity detection

The evaluation metrics are accuracy for sentiment classification and paraphrase detection, and Pearson correlation for the semantic textual similarity task.

Results

Training Metrics

Task	**Without CLN**	**With CLN integrated**
SST(Accuracy)	96.39	97.89
Quora(Accuracy)	91.60	95.73
STS(Pearson Correlation)	0.3734	0.9248

Testing Metrics

Task	**With CLN integrated**
SST(Accuracy)	62.43
STS(Pearson Correlation)	0.5478