Assignment Nº 3: RNN and GRU with PyTorch

Daniel Jader Pellattiero
고려대학교
Seoul, Republic of Korea
danieljaderpellattiero@outlook.com

Abstract

In this assignment, we employ vanilla recurrent neural networks (RNNs) and gated recurrent units (GRUs) for a text classification task on a five-category text documentation dataset. The classification dataset contains 2225 text data samples and 5 classes (politics, sports, technology, entertainment and business) in a CSV file. The sequences are initially preprocessed with a Keras tokenizer (vocabulary size 5,000), padded to a length of 100 and then split into training, validation and test sets. (70%-15%-15%).

1. Models

1.1. RNN

The RNN-based classifier implements a single layer unidirectional recurrent neural network over an embedding space (vocabulary) of size $|\mathcal{V}| = 5000$ and embedding dimension $d_e = 64$. At each time step t, the recurrent update and output layer are computed as follows:

$$h_t = \tanh\big(W_x\, x_t + W_h\, h_{t-1} + b_h\big), h_t \in \mathbb{R}^{d_h}, d_h = 128$$

$$y = \tanh\big(W_y\, h_t + b_y\big),$$

After processing the whole sentence the hidden state h_t is passed to a fully connected (FC) layer to produce the final output y. (class prediction).

The model has been trained with Adam ($\mathrm{lr} = 1 \times 10^{-3}$) and cross entropy loss for 10 epochs.

1.2. GRU

The GRU based classifier is characterised by the same overall parametrisation as the former RNN model. The gates and the hidden state are implemented as affine linear transformations, having as input/output features the same embedding and hidden state dimensions as the RNN model. (i.e. $d_h = 128$, $d_e = 64$).

At each time step t, the GRU unit computes the following equations:

$$z_t = \sigma\big(W_z x_t + U_z h_{t-1} + b_z\big),$$

$$r_t = \sigma\big(W_r x_t + U_r h_{t-1} + b_r\big),$$

$$\tilde{h}_t = \tanh\big(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h\big),$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

The model has been trained with Adam ($\mathrm{lr} = 1 \times 10^{-3}$) and cross entropy loss for 20 epochs.

2. Experimental results

Training the RNN for 10 epochs produced modest gains: validation accuracy increased from 26.95% at epoch 1 to a peak of 45.81% at epoch 6, before fluctuating and settling at 45.21% at epoch 10. The corresponding test accuracy of 45.51% suggests that the model's ability to capture sequential dependencies without gating was limited, leading to underfitting on longer patterns. In contrast, the GRU model, trained for 20 epochs under identical hyperparameters, showed a sudden and sustained improvement: validation accuracy exceeded 50% at epoch 4, reached over 70% at epoch 11, and peaked at 76.05% at epoch 18 before slight overfitting reduced it to 71.86% at epoch 20. Its final test accuracy of 73.65% confirms far superior generalization compared to the RNN. The GRU's update and reset gates clearly mitigated the vanishing gradient problem, allowing effective long range dependencies modelling.

Overall, the gated architecture not only converged faster but also turned out to be more robust and accurate.