NLU course projects

Ettore Saggiorato (247178)

University of Trento

ettore.saggiorato@studenti.unitn.it

1. Introduction

This report presents the implementation and evaluation of various techniques aimed at improving model performance in a next-word prediction task, as the first assignment for the Natural Language Understanding course at the University of Trento. The assignment's objective is to incrementally modify a baseline Recurrent Neural Network (RNN) model, exploring how these modifications affect performance. Modifications may either improve or degrade model performance.

The performance of the models is evaluated using Perplexity (PPL), a standard metric in language modeling tasks. The Penn Treebank[1] dataset is used for training and evaluation. Experiment logs are provided in the assignment's `logs` directory, formatted for TensorBoard.

2. Implementation Details

The project is implemented entirely in Python, utilizing PyTorch as the deep learning framework. Early stopping is employed during training to prevent overfitting. The assignment is divided into two parts, each focusing on different architectural choices, regularization techniques, and optimizers.

2.1. Part 1

The first part of the assignment explores fundamental changes to the model architecture, regularization techniques, and the choice of optimizer.

2.1.1. Architecture

Replacing RNN with LSTM: Long Short-Term Memory (LSTM) networks are expected to perform better than standard RNNs, particularly with long sequential data, due to their ability to mitigate the vanishing gradient problem[2].

2.1.2. Regularization

Dropout: Dropout layers are introduced at two points in the network: after the embedding layer and after the final linear layer. The expectation is that dropout will increase the model's robustness.

2.1.3. Optimizer

Replacing SGD with AdamW: The AdamW optimizer is known for its faster convergence. However, this comes with a trade-off: an increased risk of converging to a local minimum and higher memory consumption due to additional data storage requirements. AdamW typically requires a smaller learning rate compared to SGD.

2.2. Part 2

The second part focuses on more advanced regularization techniques and a specialized optimizer.

2.2.1. Regularization

Weight Tying: This technique significantly reduces the number of parameters in the network by tying the weights of the input embedding and output projection layers, which can lead to improved performance in language modeling tasks[3] and faster training time.

Variational Dropout: Variational dropout[4] applies the same dropout mask across the entire sequence within a batch, particularly for recurrent layers. This approach is expected to enforce a stronger regularization effect.

2.2.2. Optimizer

Non-monotonically Triggered Averaged SGD (NT-AvSGD): NT-AvSGD[5] aims to mitigate the instability of SGD by averaging the weights during training when the validation performance worsens. The implementation involves extending SGD to include methods that determine when to start averaging and how to compute the averages. This optimizer is expected to improve the model performance when the averaging is triggered.

3. Results

3.1. Architecture

The architecture modification (replacing the RNN with an LSTM), lead to a significant improvement in model performance as measured by PPL. Further, the LSTM required fewer training steps to reach a similar level of performance, suggesting faster convergence.

3.2. Regularization

Regularization techniques had a marked impact on model performance. Dropout applied to the output layer alone yielded the best results, possibly as the output layers is more prone to overfitting, while applying dropout to both the embedding and output layers resulted in the worst performance.

Weight tying substantially improved model performance, reducing PPL by approximately 10 points in the best run compared to identical settings without weight tying. On the other hand, variational dropout offered minimal improvement, indicating that its regularization effect may be less effective in this context.

3.3. Optimizers

The results indicate that the AdamW optimizer outperforms SGD in terms of convergence speed, achieving comparable or superior results with fewer training epochs. Interestingly, NT-AvSGD did not provide any significant benefit to the model.

4. References

[1] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," in *Proceed-*

ings of the 26th annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 1993, pp. 313–330.

[2] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[3] O. Press and L. Wolf, "Using the output embedding to improve language models," 2017. [Online]. Available: https://arxiv.org/abs/1608.05859

[4] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," 2016. [Online]. Available: https://arxiv.org/abs/1512.05287

[5] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and optimizing lstm language models," 2017. [Online]. Available: https://arxiv.org/abs/1708.02182

Architecture	# Layers	Emb./Hid.	Drop.	WT	Var. Drop.	Optim.	Lr.	Steps	PPL
RNN	1	300/200				SGD	1	19	197.76
RNN	1	300/200				SGD	2	9	224.96
RNN	1	300/300				SGD	3	10	276.26
RNN	2	300/300				SGD	1	16	192.89
LSTM	1	300/300				SGD	2	13	189.20
LSTM	1	300/300	0.2			SGD	2	20	189.23
LSTM	**1**	**300/300**	**0.5**			**SGD**	2	**27**	**179.25**
LSTM	1	150/150				AdamW	0.001	15	188.42
LSTM	1	300/300	0.5			AdamW	0.001	15	181.85
LSTM	1	300/300				AdamW	0.001	9	185.12
LSTM	1	300/300		False	0.1	SGD	2	21	218.60
LSTM	1	300/300		True	0.1	SGD	2	22	203.50
LSTM	1	300/300		True	0.2	SGD	2	23	211.16
LSTM	1	300/300		True	0.3	SGD	2	20	213.98
LSTM	**1**	**300/300**		**True**	-	**SGD**	**2**	**25**	**195.22**
LSTM	1	300/300		True	-	NTvSGD	1	26	230.10
LSTM	1	300/300		True	0.1	NTvSGD	2	18	218.24
LSTM	1	300/300		True	-	NTvSGD	2	24	201.24

Table 1: *Experiment Results* [1]