NLU course projects

Nicola Maestri (239920)

University of Trento

nicola.maestri@studenti.unitn.it

1. Introduction

In this assignment, we develop a *language model* (LM) to represent the probability distribution over a sequence of tokens. Starting with a *vanilla recurrent neural network* (RNN) followed by a linear and softmax layer trained using *stochastic gradient descent*, we explore various enhancements. Experiments incorporating *Long-Short Term Memory* (LSTM) networks, *dropout layers*, and the *AdamW* optimizer result in a significant reduction in test set perplexity. Applying *Weight Tying*, *Variational Dropout*, and *Non-monotonically Triggered Averaged SGD* (NT-ASGD) further reduce the perplexity to 104, surpassing the baseline by more than 50 points.

2. Implementation details

In the first experiment, we use a *vanilla RNN* followed by a linear and softmax layer. Enhancements include replacing the *RNN* with an *LSTM*, applying *dropout* to both embedding and hidden layers, and utilizing the *AdamW* optimizer. Model hyperparameters are provided in Table 1.

In the second experiment, we implement *Weight Tying*, *Variational Dropout* and *Non-monotonically Triggered Averaged SGD*, three regularization techniques outlined in [1].

Weight Tying shares weights for projecting the vocabulary into the embedding space and projecting the hidden state into the vocabulary. An additional linear layer aligns the hidden state with the embedding space dimensions.

Variational Dropout applies a unique binary mask for all sequences in the same batch. The mask is sampled according to a *dropout probability* p and scaled by $\frac{1}{1-p}$ to maintain the expected value of the activations.

Non-monotonically Triggered Averaged SGD is implemented following the pseudocode in Figure 1. During each epoch, we perform steps with *SGD* while maintaining an updated average of the weights starting from a dynamically determined iteration T. At the end of each epoch, averaged weights are used to update the model, and hyperparameters are reinitialized before starting the next epoch. Momentum is set to 0.9 in *SGD*, improving optimization by smoothing updates.

For both experiments, training runs for a maximum of 50 epochs, with early stopping triggered after 5 consecutive validation stagnations. Hyperparameter tuning is conducted to optimize the learning rates for both *SGD* and *AdamW* optimizers. To ensure stability, gradients are clipped to a norm of 5. Table 2 provides the full training configuration.

The model is trained using the *Cross Entropy* loss function, excluding <*pad*> tokens, while *Perplexity* is considered for evaluation.

We use the *PennTreeBank* dataset for training and evaluation. The vocabulary is derived from the corpus with two additional tokens: <*eos*> to mark the end of a sequence and <*pad*> to ensure uniform sequence lengths. Parameters for the data loaders are listed in Table 3.

3. Results

In all experiments, we report the average score over 5 runs along with the relative standard deviation. Results of both experiments are listed in Table 4, and the best scores are visualized in Figure 2.

The *Vanilla RNN*-based model trained with *SGD* achieves a test perplexity of 158.42 when the *learning rate* is set to 1. This score serves as a reference point for subsequent experiments.

Replacing the *Vanilla RNN* with an *LSTM* reduces the test perplexity to 142.11. Incorporating *dropout layers* significantly reduces test perplexity to 123.76. These results are obtained using *SGD* with a learning rate of 2. Replacing *SGD* with *AdamW* and setting the learning rate to 5e-4 further improves the final perplexity, achieving a score of 121.43. Figures 3 and 4 illustrate the progression of training and validation losses for the models mentioned above.

Substituting *Dropout* with *Variational Dropout* does not improve the performance, with a score of 121.40, while the addition of *Weight Tying* results in a test perplexity of 107.76. Using the *NT-ASGD* optimizer to train an *LSTM* model with *dropout layers* leads to a test perplexity of 130.04. Nevertheless, combining an *LSTM* model with *variational dropout* and the *NT-ASGD* optimizer achieves a perplexity of 104.27 on the test set, which is the best score obtained. Figures 5 and 6 show the progression of training and validation losses for the aforementioned models. It is interesting to observe that models trained with the *NT-ASGD* optimizer exhibit a validation loss after just one epoch that is half of that achieved by models trained with *AdamW*. Furthermore, training with *NT-ASGD* converges in fewer than 15 epochs, demonstrating that this optimizer enables faster learning in this scenario.

4. References

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

A. Tables and Figures

LSTM hyperparameters	
Embedding size	300
Hidden size	200
Output size	10001
n layers	1
bidirectional	*False*
Output dropout	0.1
Embedding dropout	0.1

Table 1: *LSTM hyperparameters*

Training hyperparameters	
n epochs	50
patience	5
clip	5
optimizer	*SGD, AdamW*
training loss	*Cross Entropy*
evealuation metric	*Perplexity*

Table 2: *Training hyperparameters*

Training loader	
Length training dataset	42068
Batchsize	64
Number of batches	658
Validation loader	
Length validation dataset	3370
Batchsize	128
Number of batches	27
Test loader	
Length test dataset	3761
Batchsize	128
Number of batches	30

Table 3: *Loader hyperparameters*

Model	**Optimizer**	**lr**	**Test PPL**	
			Avg	Std
Vanilla RNN	SGD	0.1	188.75	1.18
		1	**158.42**	0.95
		2	166.75	2.92
LSTM	SGD	0.5	147.55	0.40
		1	145.57	0.79
		2	**142.11**	0.23
LSTM + Dropout	SGD	0.5	140.82	0.55
		1	126.82	0.60
		2	**123.76**	0.26
LSTM + Dropout	AdamW	1 e-4	151.54	0.96
		5 e-4	**121.43**	0.62
		1 e-3	123.17	0.63
LSTM + Var. Dropout	AdamW	1 e-4	151.77	1.52
		5 e-4	**121.40**	0.38
		1 e-3	122.84	0.07
LSTM + Dropout + Weight Tying	AdamW	1 e-4	144.02	0.90
		5 e-4	115.35	0.34
		1 e-3	**107.76**	0.43
LSTM + Dropout	NT-ASGD	1	130.04	0.43
LSTM + Var. Dropout + Weight Tying	NT-ASGD	0.1	107.00	0.47
		0.5	**104.27**	0.55
		1	104.80	0.54

Table 4: *Test perplexity for each training setting*

Algorithm 1 Non-monotonically Triggered AvSGD (NT-AvSGD)

Inputs: Initial point w_0, learning rate γ, logging interval L, non-monotone interval n.

Initialize $k \leftarrow 0$, $t \leftarrow 0$, $T \leftarrow 0$, `logs` $\leftarrow$ `[]`
while stopping criterion not met **do**
  Compute stochastic gradient $\nabla f(w_k)$ and take SGD step (1).
  if $\text{mod}(k, L) = 0$ and $T = 0$ **then**
    Compute validation perplexity v.
    if $t > n$ and $v > \min_{l \in \{0, \cdots, t-n-1\}}$ `logs[l]` **then**
      Set $T \leftarrow k$
    end if
    Append v to `logs`
    $t \leftarrow t + 1$
  end if
  $k \leftarrow k + 1$
end while
return $\frac{\sum_{i=T}^{k} w_i}{(k-T+1)}$

Figure 1: *Non-monotonically Triggered ASGD algorithm Image courtesy [1]*

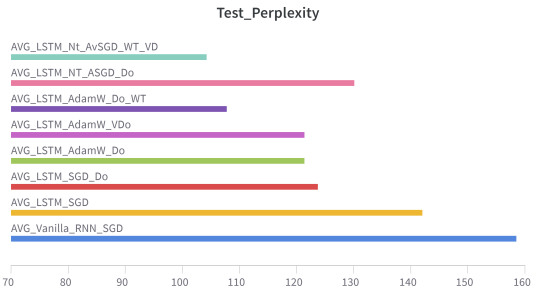

Figure 2: *Best test perplexity for each model*

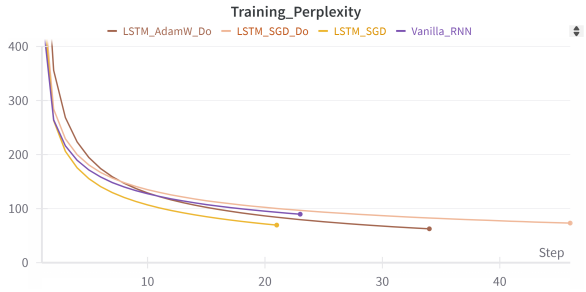

Figure 3: *Training Perplexity of Vanilla RNN model, LSTM model, LSTM with dropout, LSTM with dropout and AdamW optimizer*

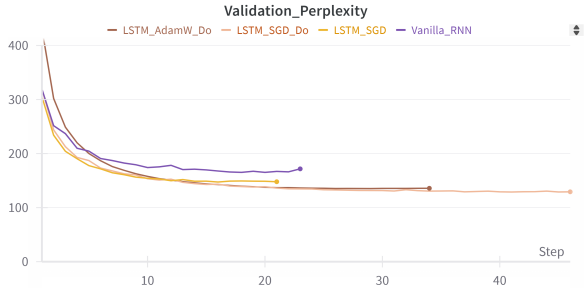

Figure 4: *Validation Perplexity of Vanilla RNN model, LSTM model, LSTM with dropout, LSTM with dropout and AdamW optimizer*

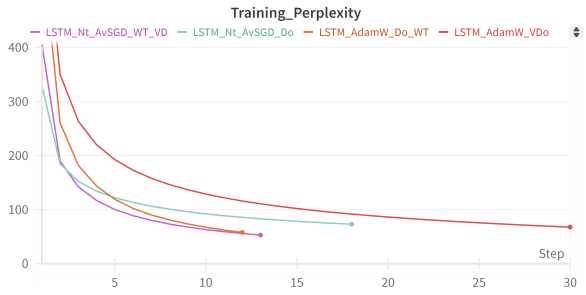

Figure 5: *Training Perplexity of LSTM model with AdamW optimizer and dropout, LSTM model with AdamW optimizer and variational dropout, LSTM model with AdamW optimizer, dropout and weight tying, LSTM model with NT-ASGD optimizer, variational dropout, weight tying.*

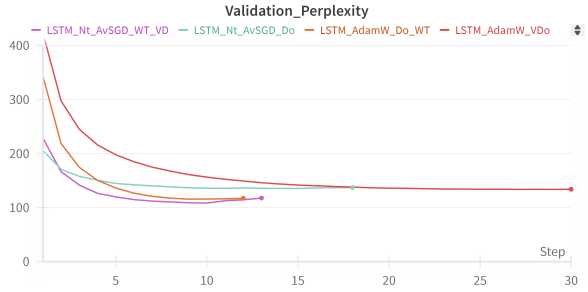

Figure 6: *Validation Perplexity of LSTM model with AdamW optimizer and dropout, LSTM model with AdamW optimizer and variational dropout, LSTM model with AdamW optimizer, dropout and weight tying, LSTM model with NT-ASGD optimizer, variational dropout, weight tying.*