Regularizing and Optimizing LSTM Language Models

Stephen Merity[1] Nitish Shirish Keskar[1] Richard Socher[1]

20160413 Soyoung Yoon
20150824 Jaeyoung Hwang
20150390 Dongmin Seo

Regularizing and Optimizing LSTM Language Models

1. Introduction
2. Previous Approaches
3. Optimization Techniques
4. Regularization Techniques
5. Evaluation
6. Conclusion & Contributions

Introduction

Language modeling is **useful** for pre-training decoders in Seq2Seq architectures, and **custom architectures** often proposed.

- Apply **Generalization / Regularization** Techniques
- Propose new **Optimization** Techniques (NT-AvSGD)
- Apply **pointer model & QRNN**

Attained **State of The Art performance** for many tasks and become popular baseline model for LM papers

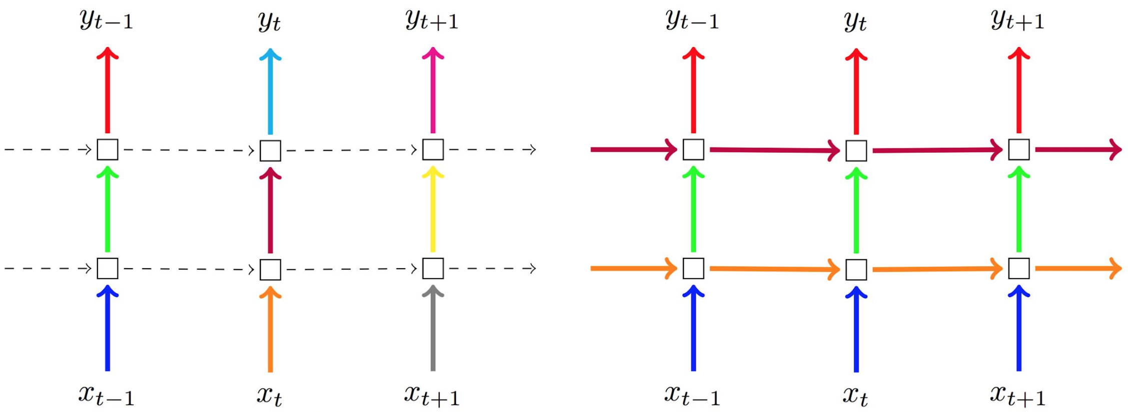
Regularizing and Optimizing LSTM Language Models

1. Introduction
2. Previous Approaches ✓
3. Optimization Techniques
4. Regularization Techniques
5. Evaluation
6. Conclusion & Contributions

Neural networks suffer from over-parameterization (**overfitting**) - **Regularization** is important for performance

Dropout & Batch normalization: good for feed-forward and CNNs
BUT Naive dropout disrupt RNN’s ability to retain long term dependencies

-> 1. Retain **same** dropout mask over multiple time steps (variational dropout)

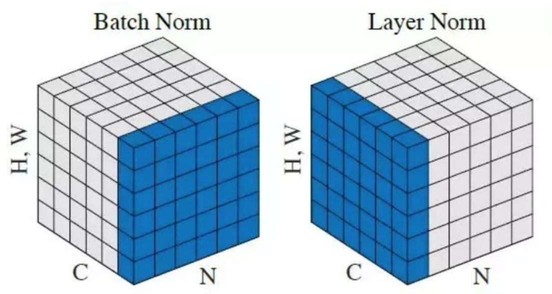


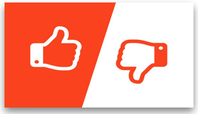
(a) Naive dropout RNN (b) Variational RNN

2. Limit updates to RNN's hidden state (≈ Zone-out)

3. Limit updates to RNN's recurrent state
   - 3-1. Restrict capacity of matrix
   - 3-2. Through element-wise interactions

4. Normalization Techniques
   - 4-1. Batch Normalization
   - 4-2. Recurrent Batch Normalization
   - 4-3. Layer Normalization





Need additional training parameters -> increase sensitivity of model

Regularizing and Optimizing LSTM Language Models

1. Introduction
2. Previous Approaches
3. Optimization Techniques
4. Regularization Techniques
5. Evaluation
6. Conclusion & Contributions

Optimization - NT-AvSGD

- SGD
  - Use **mini-batch** rather than entire data when calculate loss function

$$\boxed{\min_{w} \quad \frac{1}{N}\sum_{i=1}^{N} f_i(w),}$$

f_i is the loss function for i'th data point

- ASGD (Averaged-SGD)
  - **K**: total number of iterations
  - **T**: user-specified averaging (T < K)
  - But, **unclear guidelines** for the learning-rate and T

$$\underbrace{\frac{1}{(K-T+1)}}_{\text{Averaging Term}} \sum_{i=T}^{K} \boxed{w_i}$$

- NT-ASGD (Non-monotonically Triggered ASGD)
  - **Well defined guidelines** for the learning-rate and T

Optimization - NT-AvSGD

$$\frac{1}{(K-T+1)}\sum_{i=T}^{K} w_i$$

$$\boxed{\min_{w} \quad \frac{1}{N}\sum_{i=1}^{N} f_i(w),}$$

f_i is the loss function for i'th data point

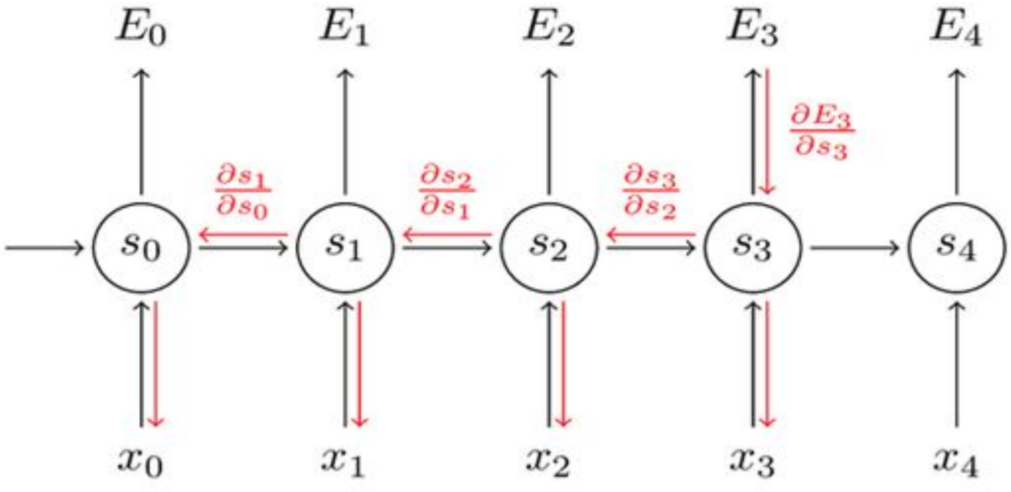
Averaging Term

Regularizing and Optimizing LSTM Language Models

1. Introduction
2. Previous Approaches
3. Optimization Techniques
4. Regularization Techniques ✓
5. Evaluation
6. Conclusion & Contributions

Variable length backpropagation sequences

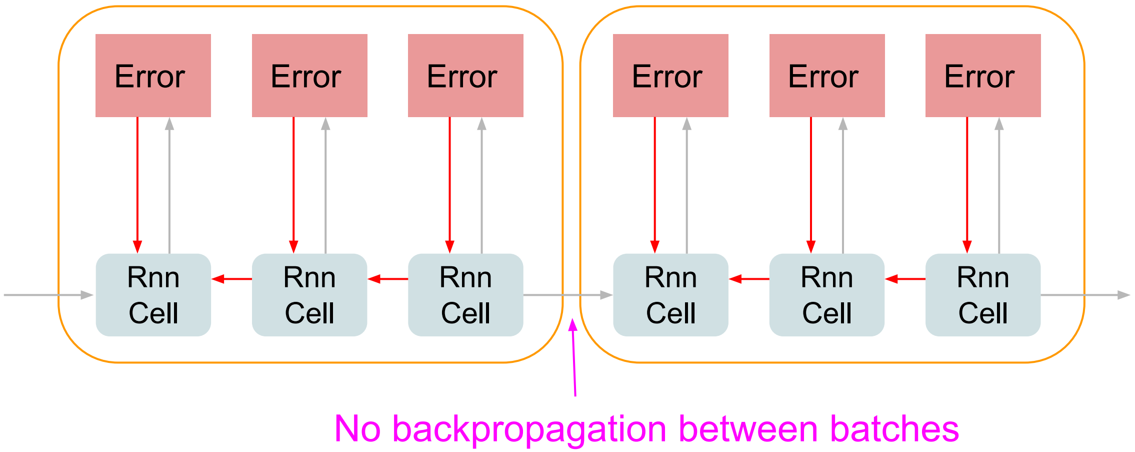
- recap: BPTT
  (Backpropagation Through Time)



Variable length backpropagation sequences

- truncated-BPTT
  - Limit backprop distance
  - Apply BPTT for each divided batch

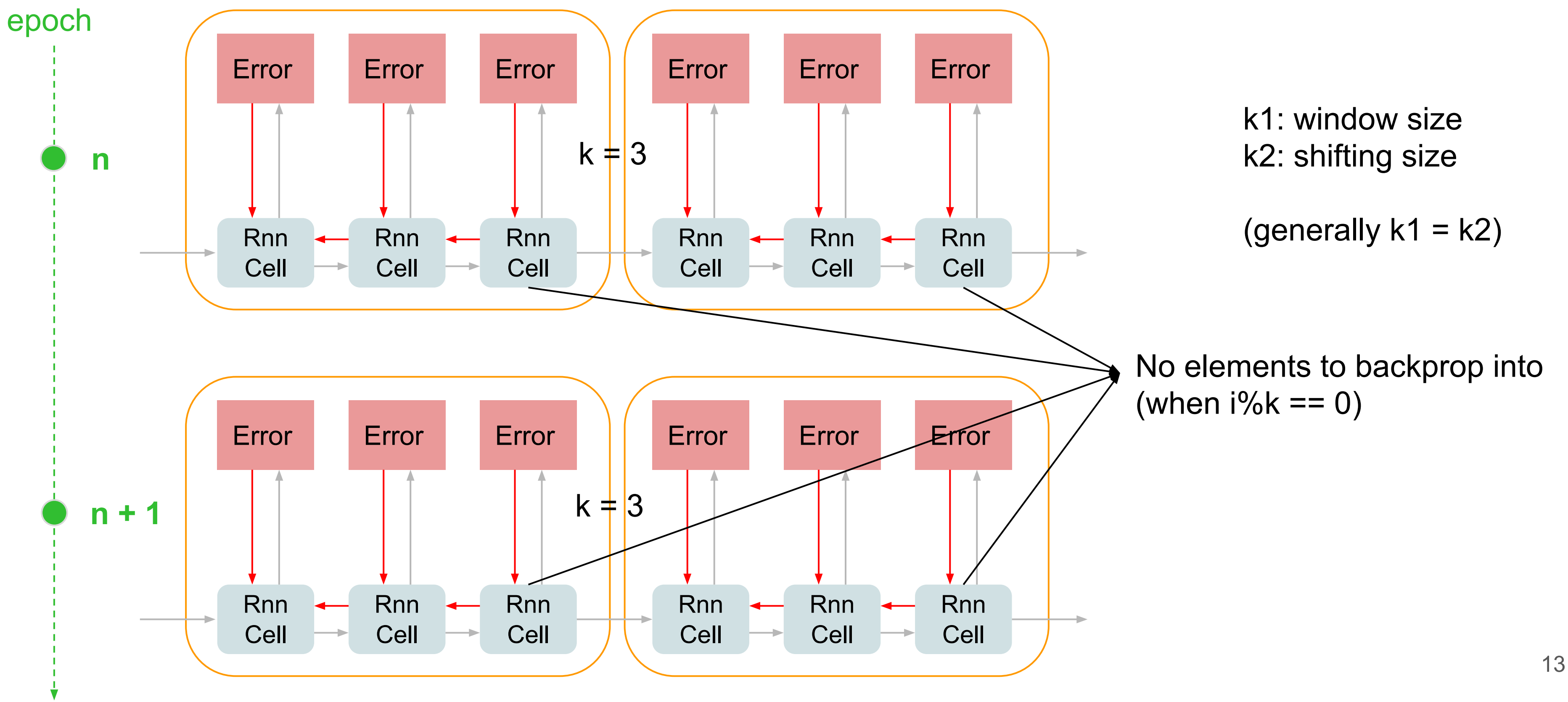
k = 3



k1: window size
k2: shifting size

(generally k1 = k2 = k)

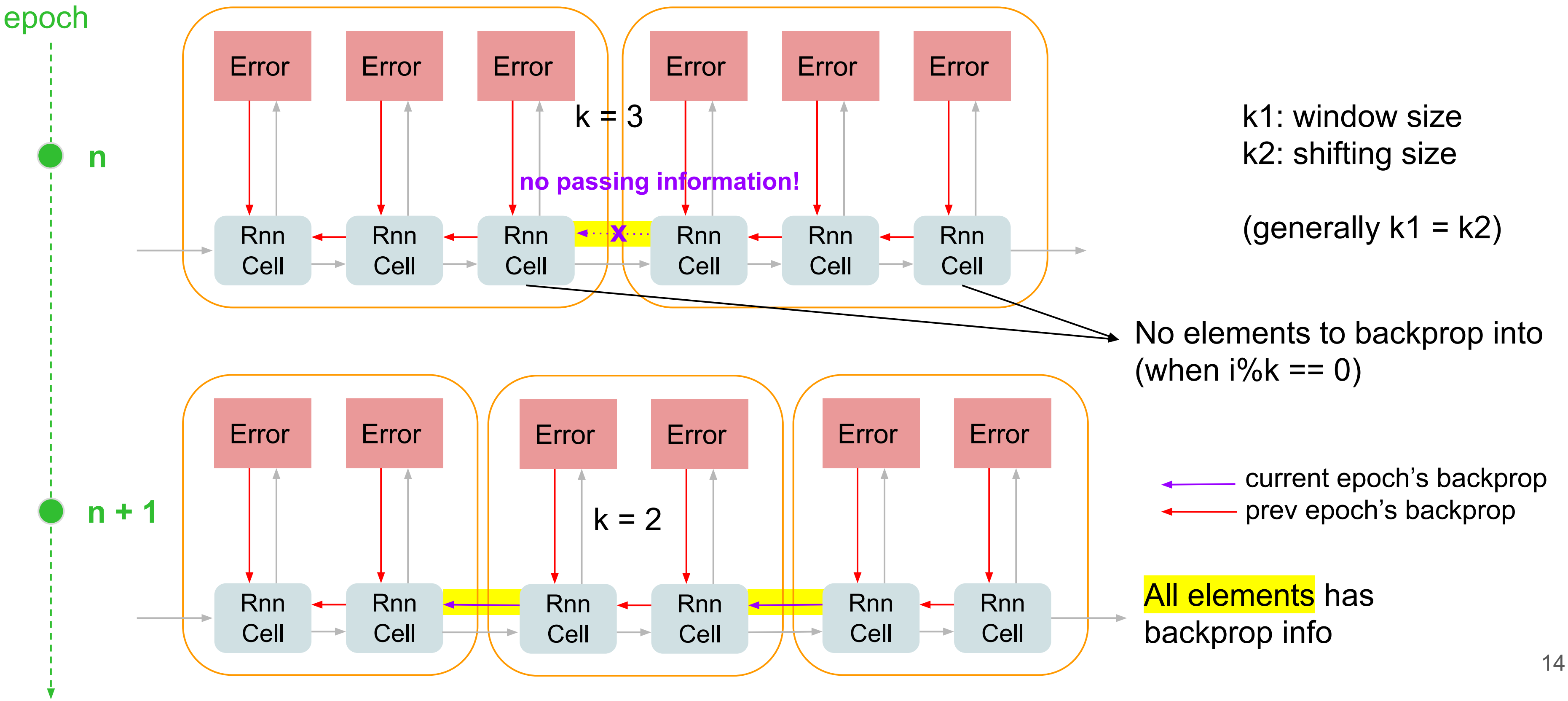
Truncated BPTT



k1: window size
k2: shifting size

(generally k1 = k2)

Variable length backpropagation sequences

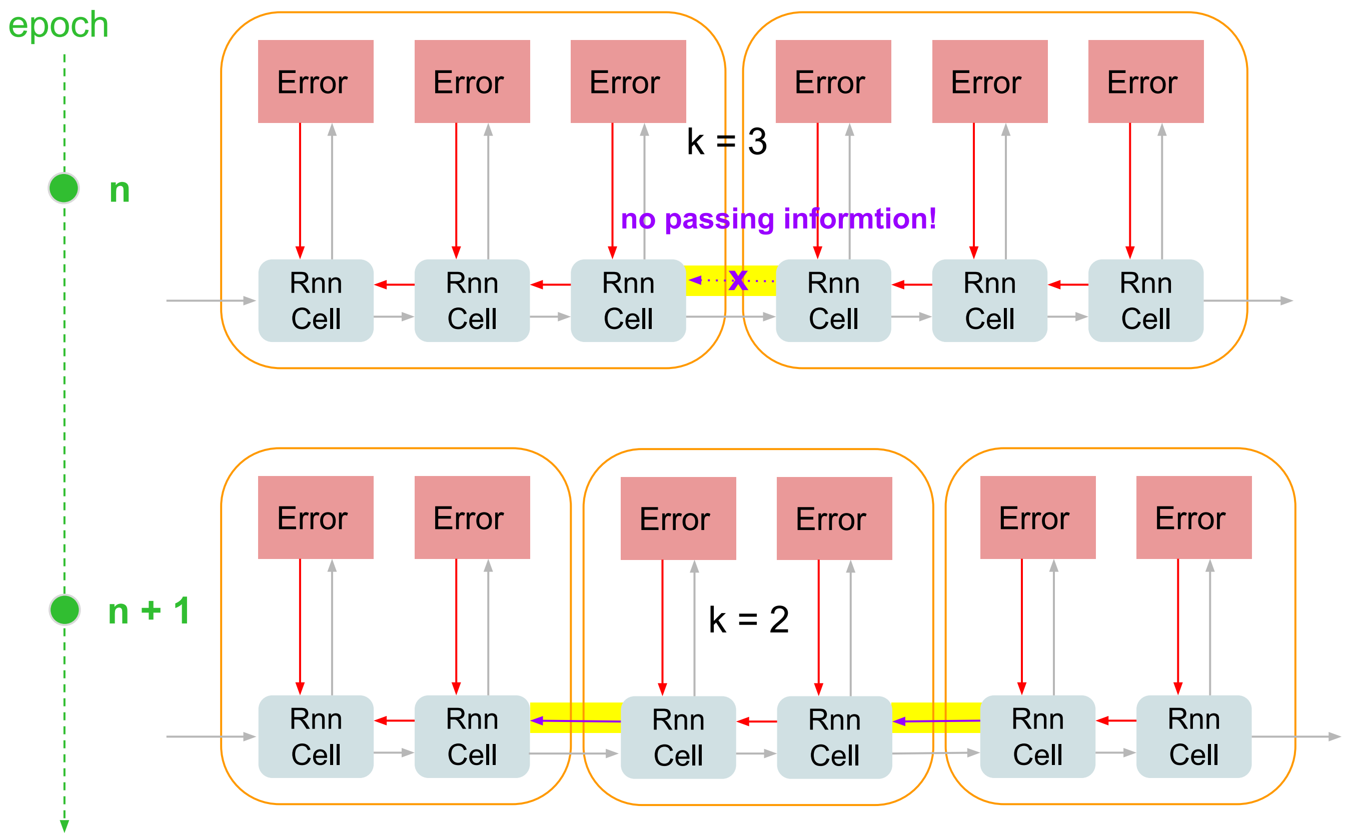


k1: window size
k2: shifting size

(generally k1 = k2)

All elements has backprop info

Variable length backpropagation sequences



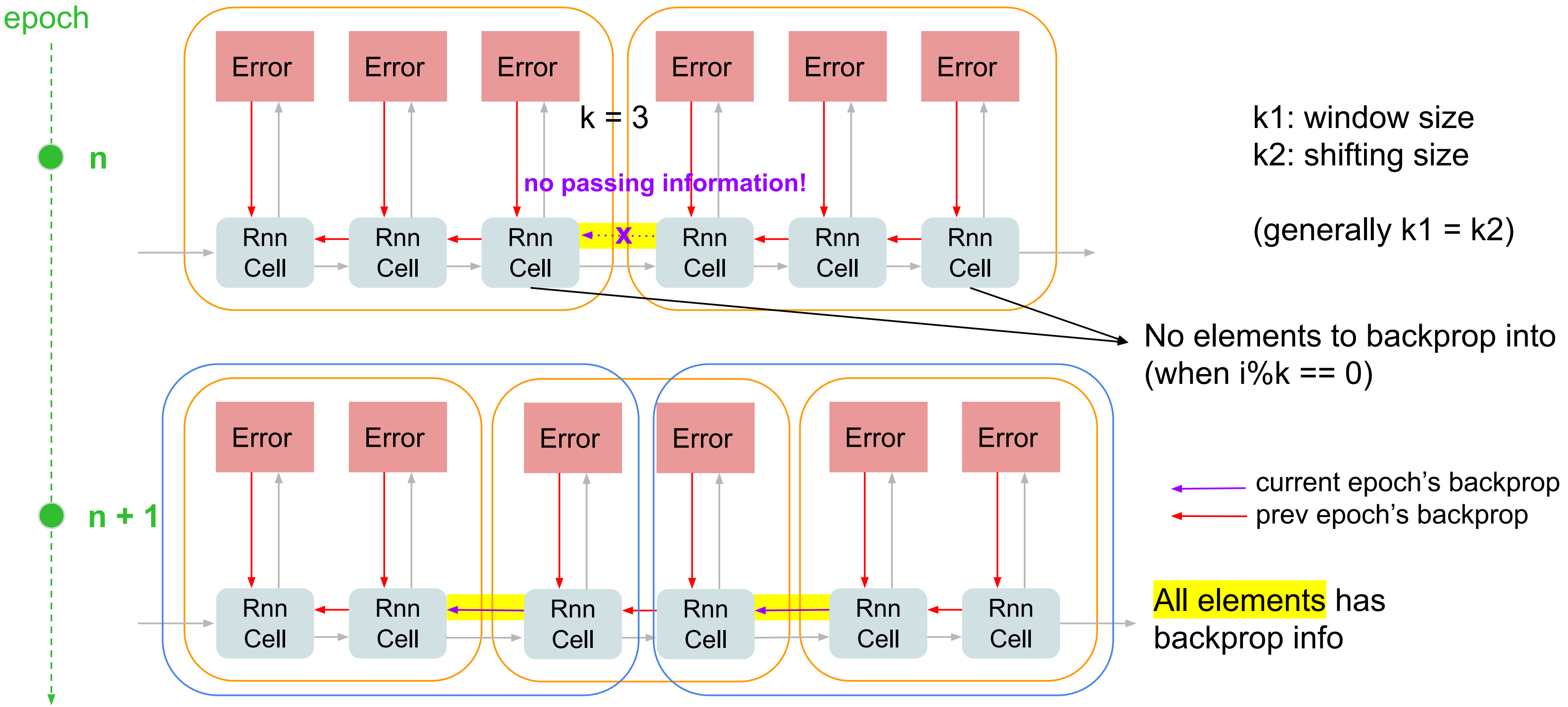
k1: window size
k2: shifting size

(generally k1 = k2)

No elements to backprop into
(when i%k == 0)

All elements has backprop info

Variable length backpropagation sequences

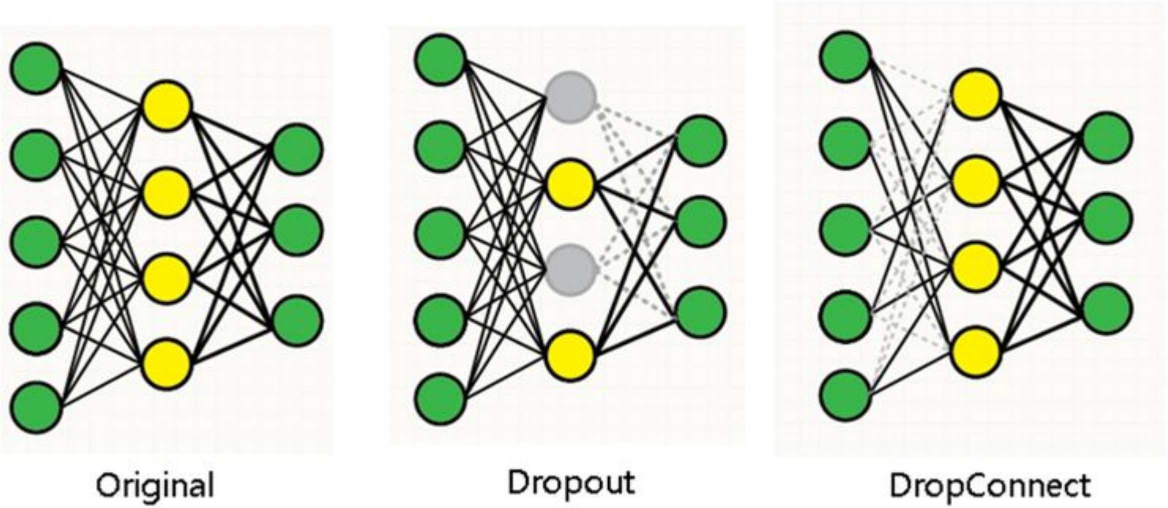


k1: window size
k2: shifting size

(generally k1 = k2)

Variational Dropout

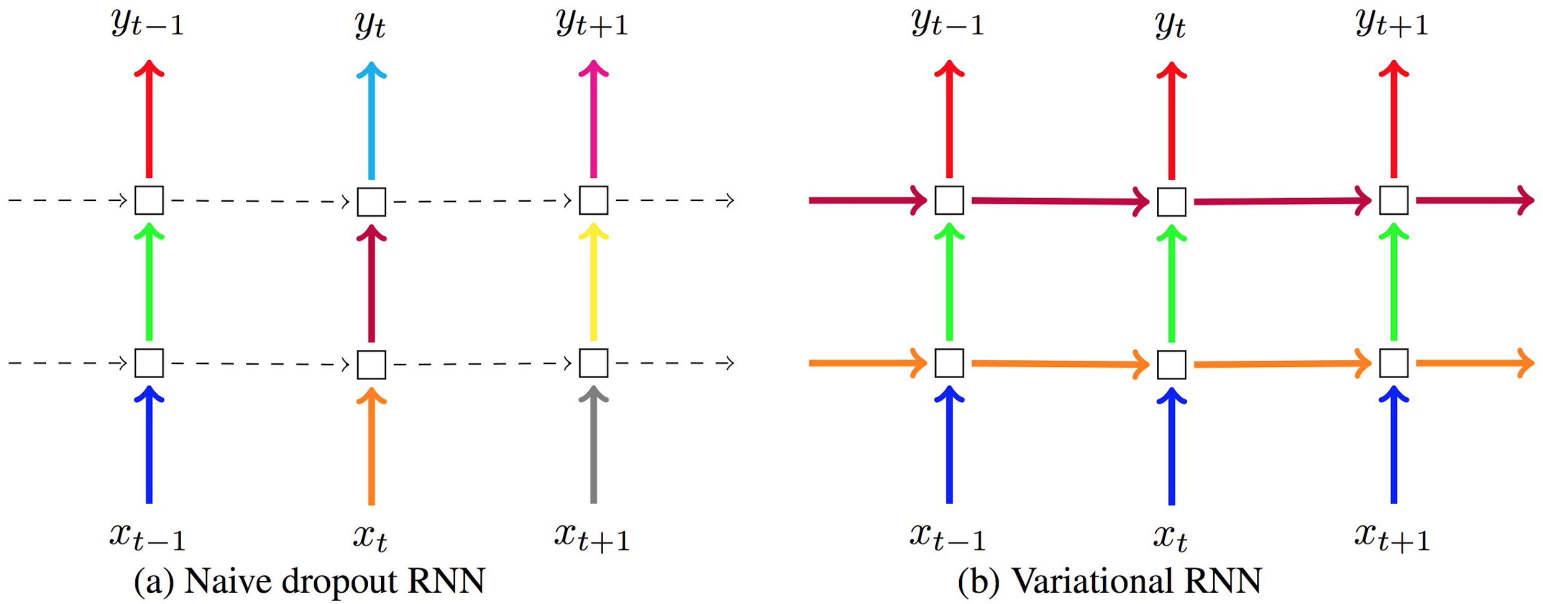
- Known Techniques: DropConnect



-> delete node

-> delete connections(weights)

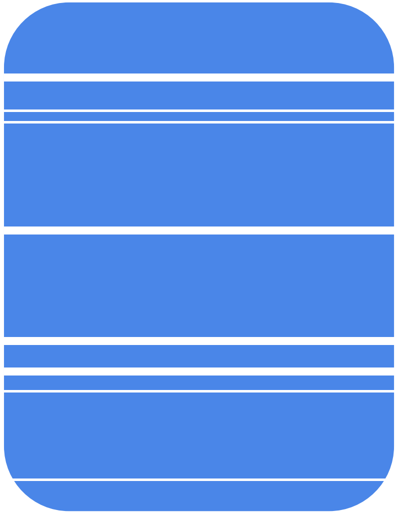
Variational Dropout



same dropout mask for multiple connections!

Embedding dropout

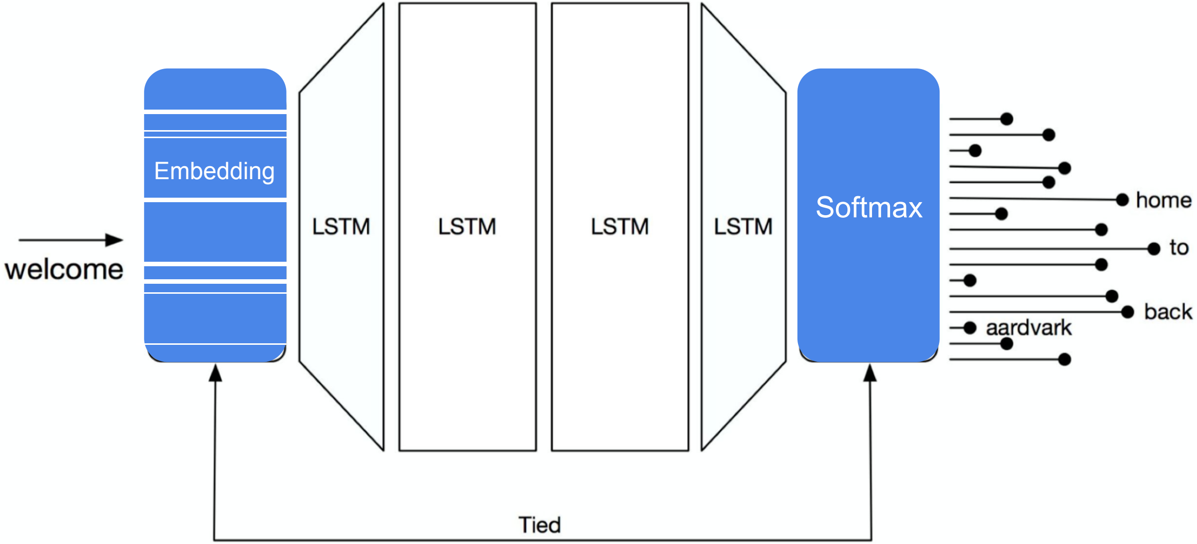
- Dropout on the embedding matrix at a word level



Embedding Matrix

Weight tying

embedding layer weight
==
softmax layer weight



Independent embedding size and hidden size

Reduce embedding size

=> reduce total parameters

AR and TAR

AR (Activation Regularization)

on individual unit activations

$$\alpha \, L_2(m \odot h_t)$$

alpha <- scale coefficient
m <- dropout mask

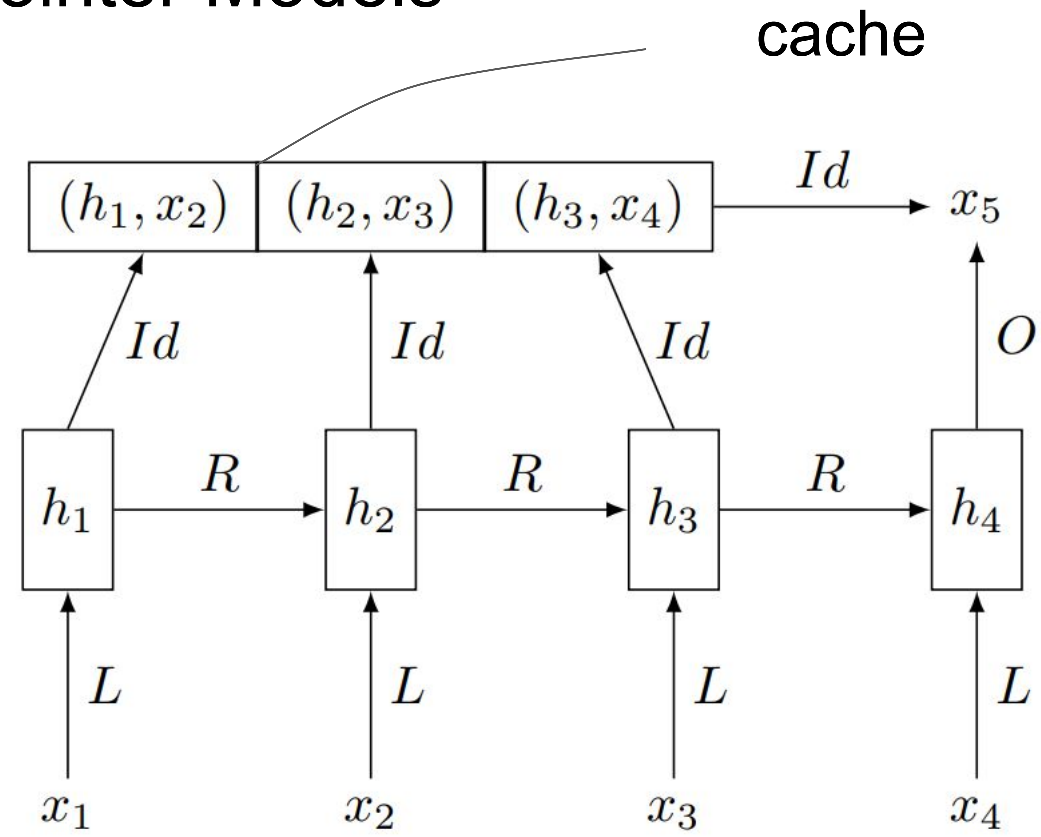
TAR(Temporal Activation Regularization)

on difference in outputs of an RNN

$$\beta \, L_2(h_t - h_{t+1})$$

beta <- scale coefficient

Pointer Models



Pointer models(neural cache model) can be directly added **on top** of a pre-trained language model

h : hidden state
x : word

Regularizing and Optimizing LSTM Language Models

1. Introduction
2. Previous Approaches
3. Optimization Techniques
4. Regularization Techniques
5. Evaluation ✓
6. Conclusion & Contributions

Evaluation

Penn Treebank(PTB) & WikiText-2(WT2)

Single model perplexity: lower is better.

Model	Parameters	Validation	Test
Zaremba et al. (2014) - LSTM (medium)	20M	86.2	82.7
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal & Ghahramani (2016) - Variational LSTM (medium)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (medium, MC)	20M	–	78.6 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (large)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal & Ghahramani (2016) - Variational LSTM (large, MC)	66M	–	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	–	78.9
Merity et al. (2016) - Pointer Sentinel-LSTM	21M	72.4	70.9
Grave et al. (2016) - LSTM	–	–	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	24M	75.7	73.2
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	25M	–	64.0
Zoph & Le (2016) - NAS Cell (tied)	54M	–	62.4
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
AWD-LSTM - 3-layer LSTM (tied)	24M	60.0	57.3
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	24M	53.9	52.8

+ variational dropout

Evaluation

Penn Treebank(PTB) & WikiText-2(WT2)

Single model perplexity: lower is better.

Model	Parameters	Validation	Test
Zaremba et al. (2014) - LSTM (medium)	20M	86.2	82.7
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal & Ghahramani (2016) - Variational LSTM (medium)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (medium, MC)	20M	–	78.6 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (large)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal & Ghahramani (2016) - Variational LSTM (large, MC)	66M	–	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	–	78.9
Merity et al. (2016) - Pointer Sentinel-LSTM	21M	72.4	70.9
Grave et al. (2016) - LSTM	–	–	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	24M	75.7	73.2
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	25M	–	64.0
Zoph & Le (2016) - NAS Cell (tied)	54M	–	62.4
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
AWD-LSTM - 3-layer LSTM (tied)	24M	60.0	57.3
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	24M	53.9	52.8

+ cache pointer

Evaluation

Penn Treebank(PTB) & WikiText-2(WT2)

Single model perplexity: lower is better.

Model	Parameters	Validation	Test
Zaremba et al. (2014) - LSTM (medium)	20M	86.2	82.7
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal & Ghahramani (2016) - Variational LSTM (medium)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (medium, MC)	20M	–	78.6 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (large)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal & Ghahramani (2016) - Variational LSTM (large, MC)	66M	–	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	–	78.9
Merity et al. (2016) - Pointer Sentinel-LSTM	21M	72.4	70.9
Grave et al. (2016) - LSTM	–	–	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	24M	75.7	73.2
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	25M	–	64.0
Zoph & Le (2016) - NAS Cell (tied)	54M	–	62.4
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
AWD-LSTM - 3-layer LSTM (tied)	24M	60.0	57.3
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	24M	53.9	52.8

+ weight tying

Evaluation

+ NT-AvSGD

Model	Parameters	Validation	Test
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$)	28M	92.3	87.7
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$) + augmented loss	28M	91.5	87.0
Grave et al. (2016) - LSTM	–	–	99.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	68.9
Melis et al. (2017) - 1-layer LSTM (tied)	24M	69.3	65.9
Melis et al. (2017) - 2-layer skip connection LSTM (tied)	24M	69.1	65.9
AWD-LSTM - 3-layer LSTM (tied)	33M	68.6	65.8
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	33M	53.8	52.0

Penn Treebank(PTB) & **WikiText-2(WT2)**

Evaluation

+ Pointer

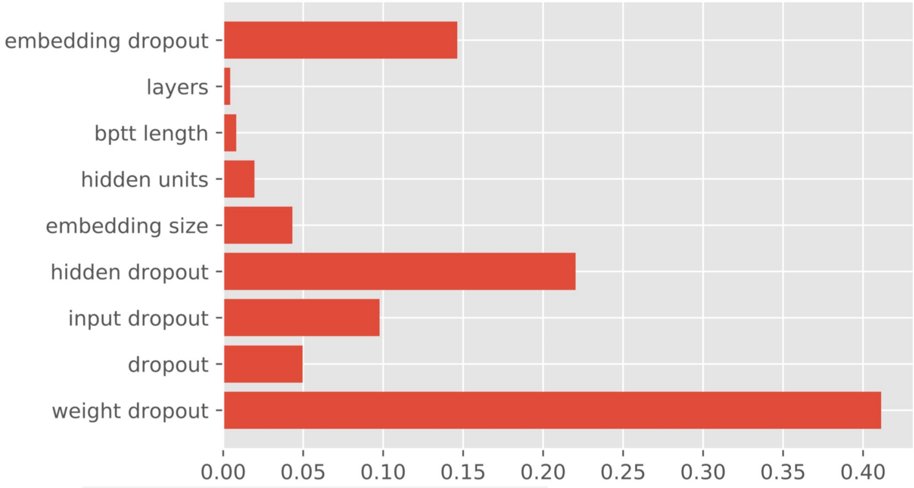
Model	Parameters	Validation	Test
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$)	28M	92.3	87.7
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$) + augmented loss	28M	91.5	87.0
Grave et al. (2016) - LSTM	–	–	$\boxed{99.3}$
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	$\boxed{68.9}$
Melis et al. (2017) - 1-layer LSTM (tied)	24M	69.3	65.9
Melis et al. (2017) - 2-layer skip connection LSTM (tied)	24M	69.1	65.9
AWD-LSTM - 3-layer LSTM (tied)	33M	68.6	65.8
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	33M	53.8	52.0

Penn Treebank(PTB) & **WikiText-2(WT2)**

Evaluation - Hyperparameter Importance!

Model	PTB Validation	PTB Test	WT2 Validation	WT2 Test
AWD-LSTM (tied)	60.0	57.3	68.6	65.8
– fine-tuning	60.7	58.8	69.1	66.0
– NT-ASGD	66.3	63.7	73.3	69.7
– variable sequence lengths	61.3	58.9	69.3	66.2
– embedding dropout	65.1	62.7	71.1	68.1
– weight decay	63.7	61.0	71.9	68.7
– AR/TAR	62.7	60.3	73.2	70.1
– full sized embedding	68.0	65.6	73.7	70.7
– weight-dropping	71.1	68.9	78.4	74.9

Each variant is evaluated by removing each feature



Importance of each feature
for decreasing the perplexity of the model

Regularizing and Optimizing LSTM Language Models

1. Motivation & Research Problem
2. Previous Approaches
3. Optimization Techniques
4. Regularization Techniques
5. Evaluation
6. Conclusion & Contributions ✓

Conclusion

Developed AWD-LSTM == ASGD Weight-Dropped LSTM (through DropConnect)

Optimization: Non-monotonic triggered ASGD >> SGD

Regularization: variable BPTT length, Variational/Embedding Dropout, AR & TAR, Independent embedding size & hidden size

Neural cache model: Further decrease perplexity

Contributions

No modifications are required for LSTM implementations

- Can be easily integrated to any blackbox LSTM layers (ex: can still use cuDNN LSTM)

- Generally applicable across other sequence learning tasks

Achieve State-of-the-Art Perplexity

- Became popular **baseline model** for LM papers (Universal Language Model Fine-tuning for Text Classification)