Bayesian Compression for Natural Language Processing

Nadezhda Chirkova*1 Ekaterina Lobacheva*1 Dmitry Vetrov1,2

1Higher School of Economics, Samsung-HSE Laboratory; 2Samsung AI Center, Russia * Equal contribution

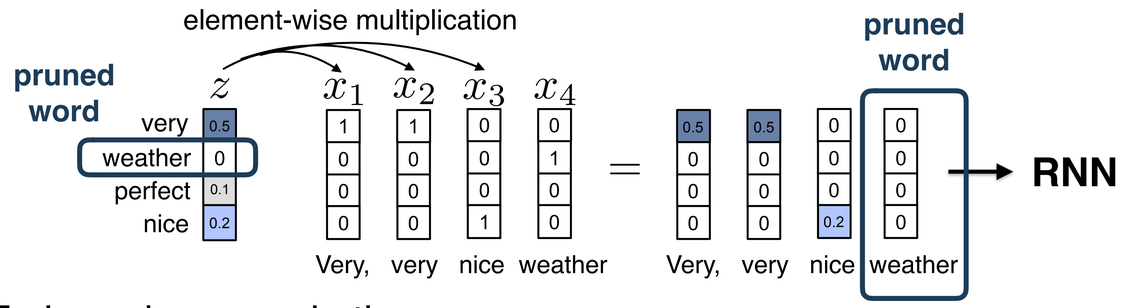
Goal Compress RNNs for NLP tasks: text classification and text generation

Method
- Apply Sparse Variational Dropout [1] taking into account RNN specifics ⇨ Weight sparsity
- Introduce group variables for elements of vocabulary ⇨ Vocabulary sparsity

Results 6—10K times weights compression and 2—70 times vocabulary compression depending on a task

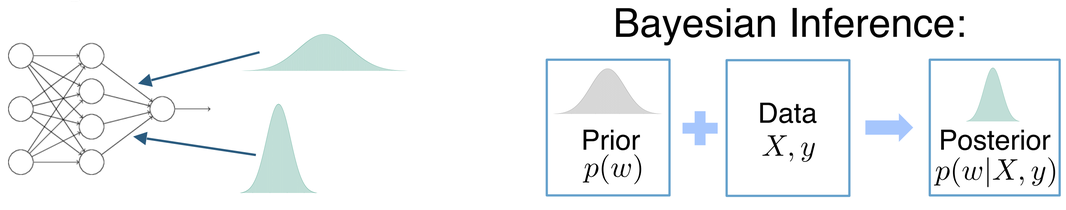
Vocabulary sparsification

Input layer: $x_t \to x_t \odot z$ where x_t is one-hot encoded token



Train and prune z in the same way as w

Baeysian neural networks



Posterior is intractable in NNs ⇨ approximate it with $q(w|\lambda)$:

$$KL(q(w|\lambda)||p(w|X,y)) \to \min_{\lambda}$$

Equivalent to variational lower bound optimization:

$$\sum_{i=1}^{N} \underbrace{\mathbb{E}_{q(w|\lambda)} \log p(y^i|x^i, w)}_{\text{Data term}} - \underbrace{KL(q(w|\lambda)||p(w))}_{\text{Regularizer}} \to \max_{\lambda}$$

Final algorithm

Training on a sequence $x = x_1 \dots x_T$:

1. Sample ϵ^w, ϵ^z from standard normal distribution
2. $w = \theta^w + \epsilon^w \odot \sigma^w; \quad z = \theta^z + \epsilon^z \odot \sigma^z$
3. $y = \text{RNN}([x_1 \odot z, \dots, x_T \odot z], w, B)$
4. update $\theta^w, \sigma^w, \theta^z, \sigma^z, B$ based on ∇ELBO

Sparsification:

$$\text{if } \theta^w_{ij}/\sigma^w_{ij} < 0.05 \Rightarrow \theta^w_{ij} = 0,\ \sigma^w_{ij} = 0$$
$$\text{if } \theta^z_{ij}/\sigma^z_{ij} < 0.05 \Rightarrow \theta^z_{ij} = 0,\ \sigma^z_{ij} = 0$$

Prediction for a sequence $x = x_1 \dots x_T$:

1. $w = \theta^w, \quad z = \theta^z$
2. $y = \text{RNN}([x_1 \odot z, \dots, x_T \odot z], w, B)$

Sparse Variational Dropout for RNNs

model

Stochastic weights w + deterministic biases B

Prior [2]: $p(w_{ij}) \propto 1/|w_{ij}|$

Approximate posterior [2]: $q(w_{ij}|\theta_{ij}, \sigma_{ij}) = \mathcal{N}(w_{ij}|\theta_{ij}, \sigma^2_{ij})$

training

Optimize ELBO to find θ, σ, B [2]:

$$-\sum_{i=1}^{N} \int \overbrace{q(w|\theta,\sigma) \log p(y^i|x^i_0, \dots, x^i_T, w, B) dw}^{\text{data term}} + \sum_{w_{ij} \in \omega} \underbrace{KL(q(w_{ij}|\theta_{ij}, \sigma_{ij})||p(w_{ij}))}_{\text{sparsity inducing regularizer}} \to \min_{\theta,\sigma,B}$$

- Integral estimation: 1 sample, reparametrization trick [2]:

$$w_{ij} = \theta_{ij} + \epsilon_{ij}\sigma_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}(\epsilon_{ij}|0, 1)$$

- KL-divergence is approximated analytically [1]

pruning

After training: $q(w_{ij}) = \delta(0)$ if $\frac{\theta_{ij}}{\sigma_{ij}} < \text{threshold}$

RNN specifics

Recurrent layer: $h_{t+1} = g(W^x x_{t+1} + W^h h_t + b)$

1. The same sample w for all t in one sequence
2. Local reparametrization trick (LRT) [2] is not applicable.

$$(Wx)_i = \sum_j \theta_{ij} x_j + \epsilon_i \sqrt{\sum_j (\sigma_{ij})^2 x_j^2}$$

normal r. v. constant

2d noise on W ⇩ 1d noise on Wx

In RNN: h_t is a random variable depending on W^h ⇨ $W^h h_t$ is not normal

Sampling the same noise on W^x for all t $\neq$ Sampling the same noise on preactivation $W^x x_t$ for all t

$$\sum_j \theta^x_{ij} x_{t,j} + \sum_j \epsilon^x_{ij} \sigma^x_{ij} x_{t,j} \neq \sum_j \theta^x_{ij} x_{t,j} + \epsilon^x_i \sqrt{\sum_j (\sigma^x_{ij})^2 x^2_{t,j}}$$

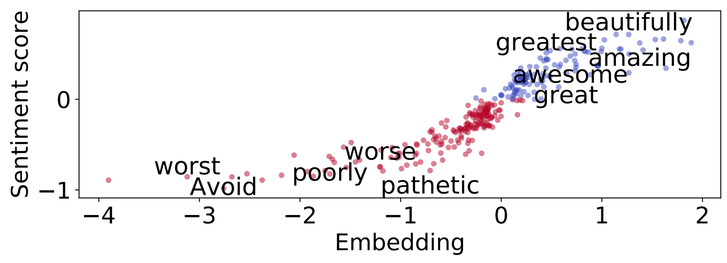
Experiments: text classification

IMDb (25K texts, 2 classes), AGNews (120K texts, 4 classes)

Architecture: Embedding (300) → LSTM (128 / 512) → FC on h_T

Task	Method	Accuracy %	Compression	Vocabulary
IMDb	Original	84.1	1x	20000
	SparseVD	**85.1**	1135x	4611
	SparseVD-Voc	83.6	**18792x**	**292**
AGNews	Original	**90.6**	1x	20000
	SparseVD	88.8	322x	5727
	SparseVD-Voc	89.2	**469x**	**2444**

IMDb: the only remaining embedding component for 292 remaining words is highly correlated with sentiment score



Experiments: text generation

PTB Corpus: char-level (6M chars), word-level (0.9M words)

Architecture: LSTM (1000 / 256)→ FC

Task	Method	Valid	Test	Compression	Vocabulary
Char PTB (Bits-per-char)	Original	1.499	1.454	1x	50
	SparseVD	1.472	1.429	**7.9x**	50
	SparseVD-Voc	**1.458**	**1.417**	6.0x	**46**
Word PTB (Perplexity)	Original	135.6	129.3	1x	10000
	SparseVD	**115.0**	**109.2**	**22.1x**	9990
	SparseVD-Voc	126.0	120.2	19.3x	**3164**

Paper

Source Code

Our follow-up on group sparsity for LSTMs

References

[1] Molchanov D., Ashukha A., Vetrov D. Variational dropout sparsifies deep neural networks. ICML 2017

[2] Kingma D., Salimans T., Welling M. Variational dropout and the local reparameterization trick. NIPS 2015