Predefined Sparseness in Recurrent Sequence Models

Thomas Demeester, Johannes Deleu, Frederic Godin, Chris Develder

thomas.demeester@ugent.be



November 1st, 2018 CoNLL, Brussels, Belgium

Sparse Neural Networks

dense model





sparse model



`smaller' model
(lower memory footprint)

Sparsifying by weight pruning





Highly sparse with accuracy close to dense models [1]

Large sparse networks can be better than small dense models [2]

BUT THEN: large dense network needed during training!

GOAL: models that are sparse from the start?

"predefined sparseness"

[1] Narang et al. "Exploring Sparsity in RNNs" (ICLR 2017)

[2] Kalchbrenner et al. "Efficient Neural Audio Synthesis" (ICML 2018)

Any recurrent cell (RNN, LSTM, GRU...): 2 types of matrices



input-to-hidden

With sparse W_{hh} and W_{hi}

- strongly reduced number of hidden-to-hidden interactions (cfr. weight dropping in W_{hh} $_{\mbox{\scriptsize [5]}}$)
- not all hidden dimensions have access to each input dimension.

why this particular choice?

Consider vanilla RNN



Consider vanilla RNN



Consider vanilla RNN - made sparse



Consider vanilla RNN - made sparse



Resulting RNN equivalent to N smaller dense RNNs in parallel

- only possible with output divided into disjoint segments
- but input can be (partly) shared between components
- holds for vanilla RNN, LSTM, GRU,...
- allows standard tools (CuDNN) / parallel processing

- baseline: AWD-LSTM model [5] with 3-layer stacked LSTM
- sparse counterpart:
 - middle LSTM hidden size x 1.5 (from 1150 to 1725)
 - **sparse**; same number of parameters
 - same regularization settings

• first train run (500 epochs)

Model	Penn Treebank test perplexity
reported [5]	58.8
baseline	58.8 ± 0.3
sparse LSTM	57.9 ± 0.3

• train further ('finetune') : sparse model overfits

• hypothesis:

the regularization effect of a priori limiting interactions between dimensions does not compensate for increased expressiveness due to larger hidden state size

supported by additional experiment "learning to recite" (see paper e)

• Goal:

decide upfront which entries in embedding matrix $\mathbf{E} \in \mathbb{R}^{Vxk}$ are 0.

• Word occurrence frequencies have Zipfian nature



source: Manning, Schütze, Raghavan, "Introduction to Information Retrieval", Cambridge UP, 2009

• Goal:

decide upfront which entries in embedding matrix $\mathbf{E} \in \mathbb{R}^{Vxk}$ are 0.

• Word occurrence frequencies have Zipfian nature

representing long tail of rare terms with short embeddings would greatly reduce memory requirements



source: Manning, Schütze, Raghavan, "Introduction to Information Retrieval", Cambridge UP, 2009

Predefined sparse embedding matrix E ?



trainable parameters = kV

Predefined sparse embedding matrix E?



Predefined sparse embedding matrix E?



- Experimental setup:
 - POS tagging on Penn Treebank
 - very small model (else too easy!)
 - 20-D word embeddings (876k params)
 - BiLSTM state size 10+10 (3k params)

- Experimental setup:
 - POS tagging on Penn Treebank
 - very small model (else too easy!)
 - 20-D word embeddings (876k params)
 - BiLSTM state size 10+10 (3k params)
- Embedding matrix



- Experimental setup:
 - POS tagging on Penn Treebank
 - very small model (else too easy!)
 - 20-D word embeddings (876k params)
 - BiLSTM state size 10+10 (3k params)
- Embedding matrix





Conclusions

- Simple ideas for predefined sparseness in RNNs and embedding layers
- Predefined Sparseness has potential in NLP
- Further investigation needed

(for very large representation sizes for large vocabularies, etc.)

• Need some "predefined sparseness" code?

https://github.com/tdmeeste/SparseSeqModels

🛛 tdmeest	e / SparseSe	qModels					⊙ Um	watch +	2	★ Star	2	Y Fork	0
↔ Code	() Issues (0)	(*) Pull requests	0 📃 Proj	jects 0	💷 Wiki	di Insigh	nts	O Settir	ngs				
Code for pa Manage topics	aper "Predefine	ed Sparseness in F	lecurrent Se	equence	Models"								Edit
6	2 commits	ű.	1 branch			🛇 0 release	95			11 1 cr	ontrib	utor	
Branch: mast	ter + New pull r	equest				Create new !	file U	pload file	s F	ind file	Clone	or downlo	ad •
tdmeest	e update README.	md							Late	ist commit	39e43	129 on 23	Aug
🖿 languag	e_modeling			first cor	nmît						2	months	ago
is sequence_labeling				first commit 2 m					months	ago			
sparse_seq				first commit			2 months ago						
E README.md update				README.m	E.md 2 months ago								
	.md												1

Predefined Sparseness in Recurrent Sequence Models

This repository contains code to run the exeriments presented in our paper Predefined Sparseness in Recurrent Sequence Nedels, presented at CONLL 2018. The package sparse_seq contains the implementation of predefined sparse LSTW is and embedding layers, as described in that paper.

- rnn.py: contains SparzetSTM, a pytorch module that allows composing a sparze single-layer LSTM based on elementary dense LSTM's, for a given parameter density, or given fractions in trems of input and hidden representation size For example, with reduce_time.5 and reduce_out=8; the sparze LSTM would have the same number of trainable parameters as a dense LSTM with half the number of input and output dimensions. Next step would be rewriting SparseLSTM for running in parallel on multiple devices, to gain in speed and memory capacity compared to the dense LSTM.
- embedding.py: contains SparseEmbedding, a pytorch module that composes a sparse embedding layer by building the total embedding matrix as a composition of a user-specified number individual trainable embedding blocks with smaller dimensions. As shown in the paper, this only behaves as intended, if the vocabulary is sorted from least to most frequent terms. Both embedding regularization mechanisms described in Merity's paper Regularizing and Optimizing LSTN Language Models are included in the code.

The folders language_medeling and sequence_labeling contain the code for the language modeling and part-ofspeech tagging experiments described in our paper. The code was developed on Python 3.6.4, with pytorch 0.4.0 (CUDA V8.0, CuDNN 6.0) and all experiments were run on a GeForce GTX 1080 core.

The code is not heavily documented. I've cleaned it a little, but it's still dynamically grown research code (you know what I mean). I'll be happy to provide more detailed descriptions if needed. Don't hesitate to drop me an email if you have any questions: thomas.demeester@ugent.be

and a second second

Thank you!

- baseline:
 - AWD-LSTM model [5]
 - 400D word embeddings, 10k words; 4M params
 - **3-layer stacked LSTM** (dimensions 400 1150 400); 20M params

- sparse counterpart:
 - similar 3-layer LSTM; 20M params
 - but: middle LSTM scaled from 1150 to 1725 units (factor 1.5) sparse: to retain same number of parameters
 - no tuning (exactly same regularization parameters)

Inspiration from literature

"application of sparse coding in language processing is far from extensive, when compared to speech processing" [3]

Need for sparse models in NLP!

"natural language is high-rank" [4]

How to train large sparse representations despite memory constraints?

[3] Wang et al. "Deep and sparse learning in speech and language processing: An overview." BICS 2016

[4] Yang et al. "Breaking the softmax bottleneck: a high-rank rnn language model." ICLR 2018

• first train run (500 epochs)

Model	Penn Treebank test perplexity
reported [5]	58.8
baseline	58.8 ± 0.3
sparse LSTM	57.9 ± 0.3

• train further ('finetune') : sparse model overfits

• train again ("finetune step" [5])



