

Low-Rank Constraints for Fast Inference in Structured Models



Justin Chiu* and Yuntian Deng* and Alexander Rush

October 14, 2021

Structured Models

- ▶ Explicitly model output associations
 - ▶ Directly or through latent variables
- ▶ Focus on combinatorially large latent discrete structures
 - ▶ Complementary to distributed representations

Scaling Structured Models

- ▶ Prior work demonstrated: Size  Performance 
 - ▶ Hidden Markov Models (HMM)
 - ▶ Probabilistic Context-Free Grammars (PCFG)
- ▶ Prior work scaled via
 - ▶ Sparsity for HMMs¹
 - ▶ Low-rank tensor decompositions for PCFGs²
- ▶ This work: low-rank matrix constraints
 - ▶ More general
 - ▶ Less speedup

¹Chiu and Rush, *Scaling Hidden Markov Language Models*.

²Yang, Zhao, and Tu, 'PCFGs Can Do Better: Inducing Probabilistic Context-Free Grammars with Many Symbols'.

Inference as Matrix-Vector Products

- ▶ Inference: sequence of matrix-vector products
- ▶ Speed up via fast mat-vecs
- ▶ Applies to a large family of structured models

Fast Matrix-Vector Products

- ▶ Mat-vecs take $O(L^2)$ computation
- ▶ Various fast methods
 - ▶ Sparsity (nnz entries)
 - ▶ Fast Fourier Transform ($L \log L$)
 - ▶ Low-Rank factorization (LR)
- ▶ Connected to work in efficient attention and low-dimensional kernel approximations³

³Choromanski et al., *Rethinking Attention with Performers*; Peng et al., *Random Feature Attention*; Blanc and Rendle, *Adaptive Sampled Softmax with Kernel Based Sampling*.

Roadmap

- ▶ Speeding up HMM inference
- ▶ Speeding up PCFG inference
- ▶ Generalization to hypergraph inference
- ▶ Experiments

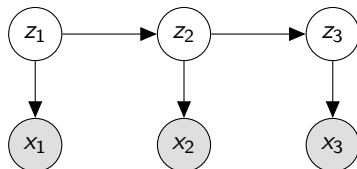
Two Examples

some text here some text here
some text here some text here
some text here

► Blah

Hidden Markov Models (HMMs)

For times t , model states $z_t \in [Z]$, and tokens $x_t \in [X]$,



with joint distribution

$$p(x, z) = \prod_t p(x_t \mid z_t) p(z_t \mid z_{t-1})$$

Inference

Given observed $x = (x_1, \dots, x_T)$ We wish to maximize

$$p(x) = \sum_{z_1} \cdots \sum_{z_T} p(x, z) = \alpha_1^\top \Lambda_2 \Lambda_3 \cdots \Lambda_T \mathbf{1},$$

where we have the

$$\begin{aligned} \text{start,} \quad & [\alpha_1]_{z_1} = p(x_1 \mid z_1)p(z_1), \\ \text{and transition operators,} \quad & [\Lambda_t]_{z_{t-1}, z_t} = p(x_t \mid z_t)p(z_t \mid z_{t-1}) \end{aligned}$$

Inference: Backward Algorithm

- ▶ Performing multiplications from right to left

$$p(x) = \alpha_1^\top (\Lambda_2(\Lambda_3 \mathbf{1}))$$

- ▶ Recursively

$$\beta_t = \Lambda_t \beta_{t+1}$$

- ▶ Requires $O(TZ^2)$ operations in total

Low-Rank Factorization

Factor matrices $\Lambda \in \mathbb{R}^{Z \times Z}$ into product of $U, V \in \mathbb{R}^{Z \times R}$

The diagram shows the equation $\Lambda \times \beta = U \times (V^T \times \beta)$ using boxes to represent dimensions. Λ is in a square box, β is in a tall vertical box, U is in a tall vertical box, V^T is in a wide horizontal box, and the final β is in a tall vertical box. The entire right-hand side expression is enclosed in large parentheses.

$$\Lambda \times \beta = U \times \left(V^T \times \beta \right)$$

resulting in two matrix-vector products of cost $O(ZR)$ each

Hypergraph Marginalization



Hypergraph Marginalization Algorithm

Algorithm 1 Hypergraph marginalization

[Matrix Form]

for $u \leftarrow v$ hyperedge e topologically **do**

$$\alpha_u \stackrel{+}{\leftarrow} \Psi_e \beta_v$$

return $\alpha_S^\top \mathbf{1}$

Algorithm 2 Low-rank marginalization

for $u \leftarrow v_1, v_2$ hyperedge e topologically **do**

$$\gamma \leftarrow V_e^\top \beta_v$$

$$\alpha_u \stackrel{+}{\leftarrow} U_e \gamma$$

return $\alpha_S^\top \mathbf{1}$

$\triangleright O(LN)$

Experiments

Experiments

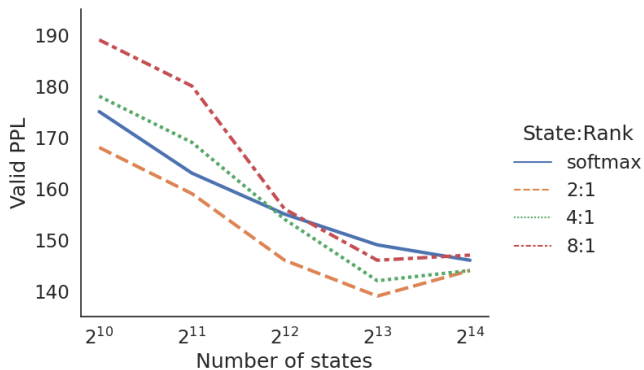
- ▶ Language modeling on PTB
- ▶ Feature map $\phi(U) = \exp(UW)$, with learned $W \in \mathbb{R}^{R \times R}$
- ▶ Baseline: Softmax HMM

HMM Accuracy

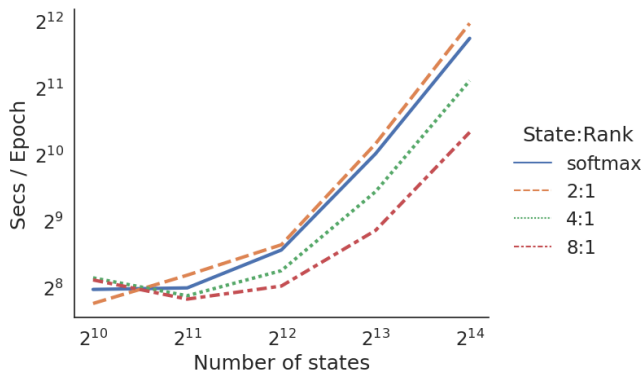
Model	Val	Test
AWD-LSTM	60.0	57.3
VL-HMM	128.6	119.5
HMM	144.3	136.8
LHMM	141.4	131.8

HMM Speed vs Accuracy Frontier

HMM Accuracy vs Rank



HMM Speed vs Rank



HMM Music Results

Model	Nott	Piano	Muse	JSB
RNN-NADE	2.31	7.05	5.6	5.19
R-Transformer	2.24	7.44	7.00	8.26
LSTM	3.43	7.77	7.23	8.17
LV-RNN	2.72	7.61	6.89	3.99
SRNN	2.94	8.20	6.28	4.74
TSBN	3.67	7.89	6.81	7.48
HMM	2.43	8.51	7.34	5.74
LHMM	2.60	8.89	7.60	5.80






PCFG Results

$ \mathcal{N} $	$ \mathcal{P} $	Model	N	PPL	Batch/s
30	60	PCFG	-	252.60	4.37
		LPCFG	8	247.02	3.75
		LPCFG	16	250.59	3.74
60	120	PCFG	-	234.01	2.99
		LPCFG	16	217.24	3.55
		LPCFG	32	213.81	3.35
100	200	PCFG	-	191.08	0.98
		LPCFG	32	203.47	1.56
		LPCFG	64	194.25	1.24

HSMM Results

Model	L	N	NLL	Batch/s
HSMM	2^6	-	1.428e5	1.28
HSMM	2^7	-	1.427e5	0.45
HSMM	2^8	-	1.426e5	0.13
LHSMM	2^7	2^7	1.427e5	0.24
LHSMM	2^8	2^6	1.426e5	0.20
LHSMM	2^9	2^5	1.424e5	0.18
LHSMM	2^{10}	2^4	1.423e5	0.10

Citations

-  Blanc, Guy and Steffen Rendle. *Adaptive Sampled Softmax with Kernel Based Sampling*. 2018. [arXiv: 1712.00527 \[cs.LG\]](#).
-  Chiu, Justin T. and Alexander M. Rush. *Scaling Hidden Markov Language Models*. 2020. [arXiv: 2011.04640 \[cs.CL\]](#).
-  Choromanski, Krzysztof et al. *Rethinking Attention with Performers*. 2021. [arXiv: 2009.14794 \[cs.LG\]](#).
-  Peng, Hao et al. *Random Feature Attention*. 2021. [arXiv: 2103.02143 \[cs.CL\]](#).
-  Yang, Songlin, Yanpeng Zhao, and Kewei Tu. 'PCFGs Can Do Better: Inducing Probabilistic Context-Free Grammars with Many Symbols'. In: *CoRR* [abs/2104.13727 \(2021\)](#). [arXiv: 2104.13727](#). URL: <https://arxiv.org/abs/2104.13727>.