

Low-Rank Factorizations for Fast Inference in Structured Models

Justin Chiu*¹ Yuntian Deng*² Alexander Rush¹

¹Cornell Tech



²Harvard University

October 18, 2021

Structured Models

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

Scaling Structured Models

- ▶ Prior work demonstrated: Size  Performance 
 - ▶ Hidden Markov Models (HMM)¹
 - ▶ Probabilistic Context-Free Grammars (PCFG)²
- ▶ This work: low-rank matrix factorizations
 - ▶ Generalize to hypergraph models

¹Dedieu et al., *Learning higher-order sequential structure with cloned HMMs*; Chiu 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 matvec methods
- ▶ Applies to a large family of structured models

Fast Matrix-Vector Products

- ▶ Matvecs take $O(L^2)$ computation
- ▶ Various fast methods
 - ▶ Sparsity (nnz entries)
 - ▶ Fast Fourier Transform ($L \log L$)
 - ▶ Low-Rank factorization (LR)
- ▶ Connected to efficient attention and 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*.

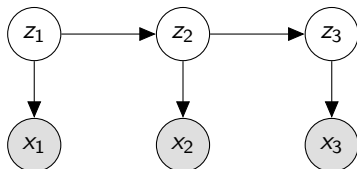
Roadmap

- ▶ Inference in HMMs and PCFGs as matvecs
- ▶ Low-rank matvec inference
- ▶ Generalization to hypergraph inference
- ▶ Experiments

Inference as Matvecs

Hidden Markov Models (HMMs)

For times t , model states $z_t \in [L] = \mathcal{L}$, and tokens $x_t \in [X] = \mathcal{X}$,



with joint distribution

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

Inference in HMMs

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

$$p(x) = \sum_{z_1} \cdots \sum_{z_T} p(x, z) = \mathbf{1}^\top \psi_1 \psi_2 \cdots \psi_T \mathbf{1},$$

where

$$\begin{aligned} [\psi_t]_{z_t, z_{t+1}} &= p(z_{t+1}, x_t \mid z_t) \\ [\psi_1]_{z_1, z_2} &= p(z_2, x_1 \mid z_1) p(z_1) \end{aligned}$$

Matvec Inference in HMMs

Algorithm 1 HMM Inference

for $t \leftarrow (t + 1)$ in right-to-left order **do**

$$\beta_t \stackrel{+}{\leftarrow} \Psi_t \beta_{t+1}$$

return $\beta_0^\top \mathbf{1}$

Probabilistic Context-Free Grammars (PCFG)

A context-free grammar $\mathcal{G} = (\mathcal{L}, \mathcal{R})$ where

\mathcal{L} : Node labels; \mathcal{X} : Tokens; \mathcal{R} : Rules,

where rules take the form

$$A \rightarrow B C, \quad A, B, C \in \mathcal{L}$$

$$P \rightarrow x, \quad P \in \mathcal{L}, x \in \mathcal{X}$$

In a PCFG, each rule has probability mass

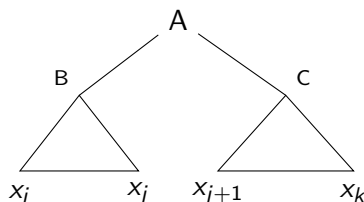
$$p(r) = p(B, C \mid A)$$

The joint distribution over rules in a tree t

$$p(t) = \prod_{r \in t} p(r)$$

Inference in PCFGs

- ▶ For a given observation x , compute $p(x) = \sum_{t: \text{yield}(t)=x} p(t)$ via dynamic programming
- ▶ For each span (i, k) , sum over split point $j \in (i, k)$:



- ▶ Similar to HMMs, define

$$[\Psi]_{z_u, (z_1, z_2)} = p(B = z_1, C = z_2 \mid A = z_u),$$

for each rule

Matvec Inference in PCFGs

Algorithm 2 PCFG Inference

```
for  $(i, k) \leftarrow (i, j), (j, k)$  in span-size order do  
  for  $z_1, z_2 \in \mathcal{L}_{i,j} \times \mathcal{L}_{j,k}$  do  
     $[\beta_{i,j,k}]_{(z_1, z_2)} = [\alpha_{i,j}]_{z_1} [\alpha_{j,k}]_{z_2}$   
     $\alpha_{i,k} \stackrel{+}{\leftarrow} \Psi_{i,j,k} \beta_{i,j,k}$   
return  $\alpha_{1,T}^\top \mathbf{1}$ 
```

Speeding Up Inference

Low-Rank Factorization

- ▶ Factor matrices $\Psi = UV^T$, $U \in \mathbb{R}^{L \times R}$, $V \in \mathbb{R}^{L' \times R}$

$$\boxed{\Psi} \times \boxed{\beta} = \boxed{U} \times \left(\boxed{V^T} \times \boxed{\beta} \right)$$

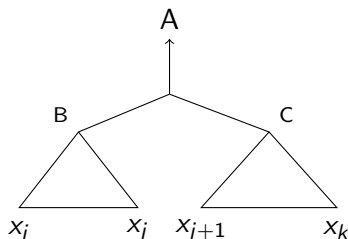
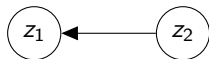
- ▶ Two matrix-vector products of cost $O(LR)$ and $O(L'R)$

Expressiveness and Generality

- ▶ Rank constraints limit expressivity
- ▶ Replace $\Psi = UV^\top$ for a subset of parameter matrices
 - ▶ Transition matrix for HMMs
 - ▶ Subset of the transition matrix for PCFGs
- ▶ An L -state HMM with rank R ($< L$) is more expressive than an R -state HMM
- ▶ What other models does this work for?

Hypergraph Marginalization

- ▶ Models where exact inference is a directed acyclic hypergraph
- ▶ Hypergraph contains a node set and hyperedge set
 - ▶ Nodes have label set \mathcal{L}
 - ▶ Hyperedges join a single head node u and a list of tail nodes v



Hyperedge representations for HMMs and PCFGs

Hypergraph Marginalization Algorithms

Algorithm 3 Hypergraph marginalization

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

$$\beta_v \leftarrow \alpha_{v_1} \alpha_{v_2}^\top \quad \triangleright O(L^{|e|})$$

$$\alpha_u \stackrel{+}{\leftarrow} \Psi_e \beta_v \quad \triangleright O(L^{|e|+1})$$

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

Algorithm 4 Low-rank marginalization

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

$$\beta_v \leftarrow \alpha_{v_1} \alpha_{v_2}^\top \quad \triangleright O(L^{|e|})$$

$$\gamma \leftarrow V_e^\top \beta_v \quad \triangleright O(L^{|e|}R)$$

$$\alpha_u \stackrel{+}{\leftarrow} U_e \gamma \quad \triangleright O(LR)$$

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

Experiments

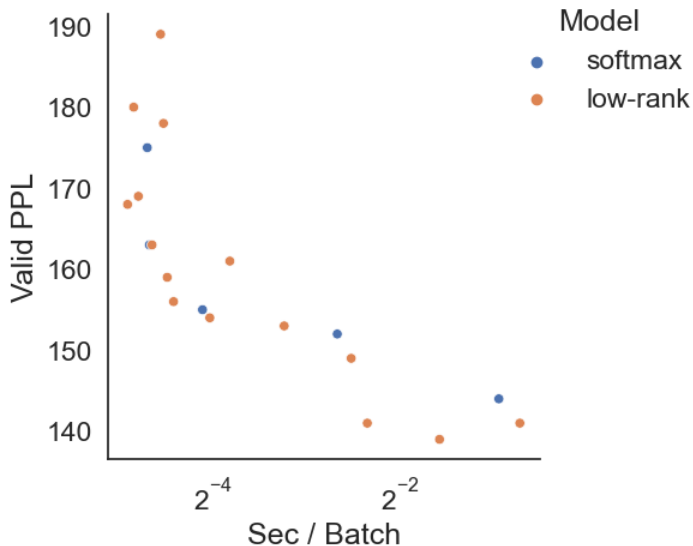
Experiments

- ▶ Compare speed vs accuracy frontier
- ▶ Language modeling on PENN TREEBANK⁴
 - ▶ Softmax HMM and PCFG vs low-rank versions (LHMM, LPCFG)
 - ▶ Evaluate accuracy with perplexity, a function of likelihood
- ▶ Video modeling on CROSSTASK⁵
 - ▶ Softmax HSMM vs low-rank HSMM
 - ▶ Evaluate accuracy with negative log-likelihood

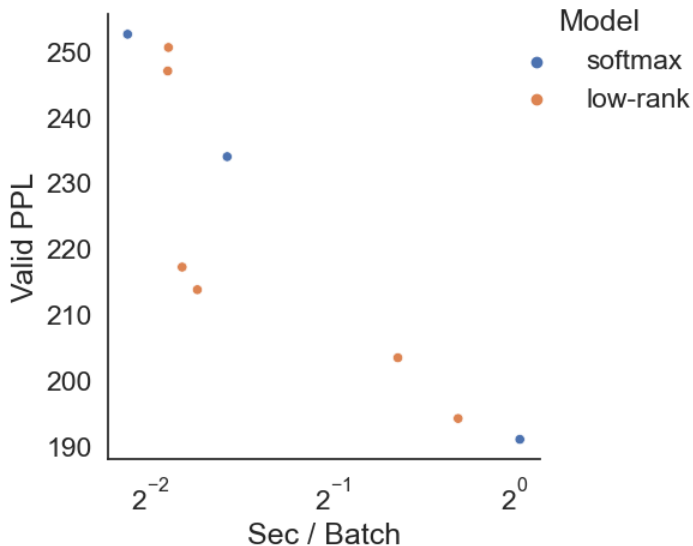
⁴Marcus, Santorini, and Marcinkiewicz, 'Building a Large Annotated Corpus of English: The Penn Treebank'.

⁵Zhukov et al., 'Cross-task weakly supervised learning from instructional videos'.

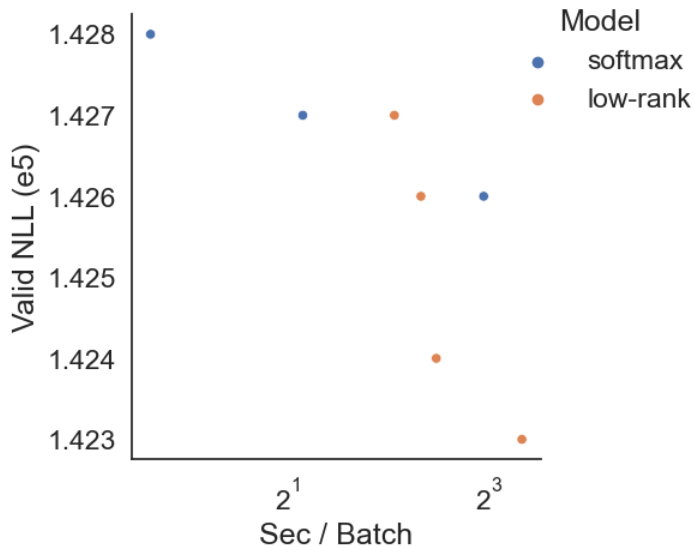
HMM Speed vs Accuracy



PCFG Speed vs Accuracy



HSMM Speed vs Accuracy



Conclusion

- ▶ Introduce a low-rank factorization to speed up inference
- ▶ Applies to models with hypergraph inference
- ▶ Most effective with large models

Citations I

-  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].
-  Dedieu, Antoine et al. *Learning higher-order sequential structure with cloned HMMs*. 2019. arXiv: 1905.00507 [stat.ML].
-  Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 'Building a Large Annotated Corpus of English: The Penn Treebank'. In: *Computational Linguistics* 19.2 (1993), pp. 313–330. URL: <https://www.aclweb.org/anthology/J93-2004>.
-  Peng, Hao et al. *Random Feature Attention*. 2021. arXiv: 2103.02143 [cs.CL].

Citations II



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>.



Zhukov, Dimitri et al. 'Cross-task weakly supervised learning from instructional videos'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3537–3545.

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