Low-Rank Constraint 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 HMMs[1]
  - Low-rank tensor decompositions for PCFGs[2]
- This work: low-rank matrix constraints
  - More general
  - Less speedup

---

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

[2] Yang, 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 approximations[3]

[3] Choromanski 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

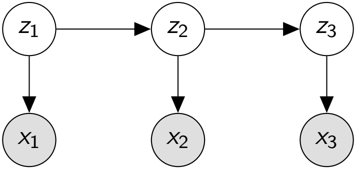
Two Examples

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

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

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 $U, V \in \mathbb{R}^{Z \times R}$

$$\boxed{\Lambda} \times \boxed{\beta} = \boxed{U} \times \left(\boxed{V^\top} \times \boxed{\beta} \right)$$

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

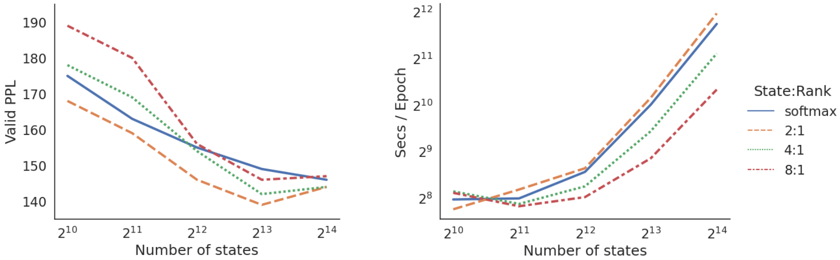
Asdf

Experiments

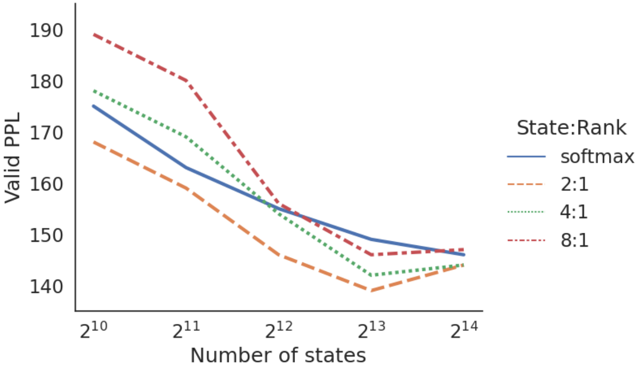
Experiments

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

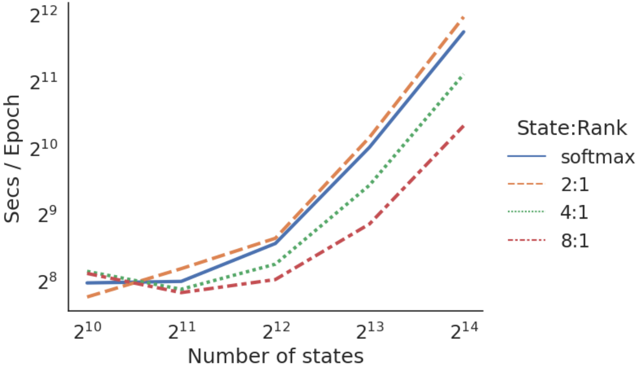
HMM Performance



HMM Accuracy

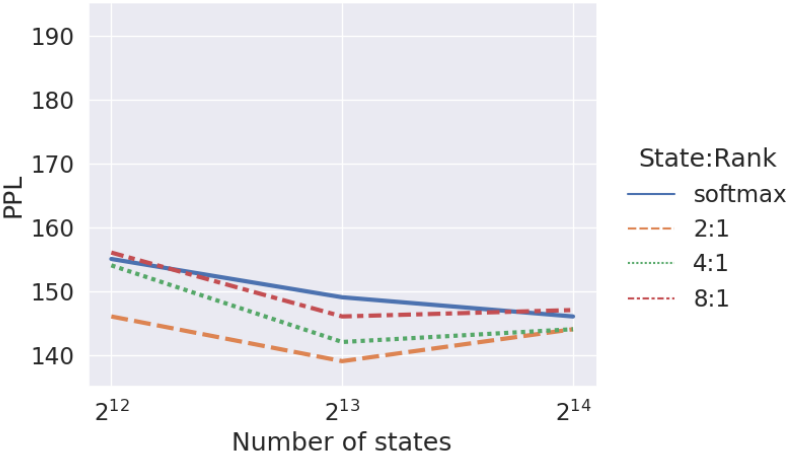


HMM Speed

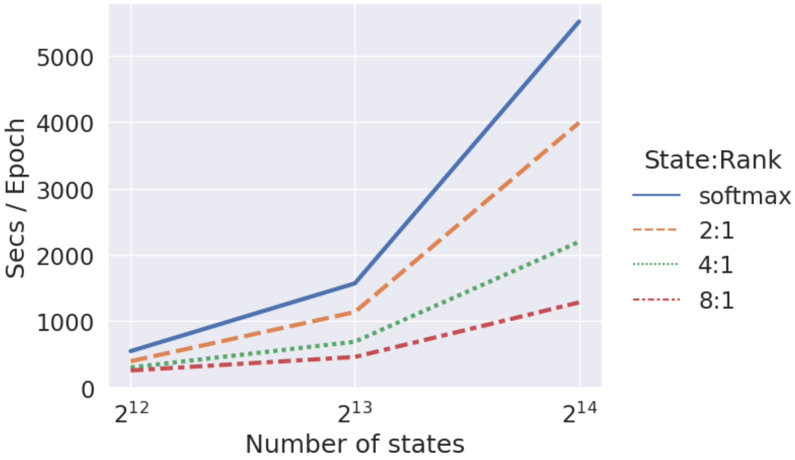


Secs / Epoch
2^{12}
2^{11}
2^{10}
2^9
2^8
2^{10}
2^{11}
2^{12}
2^{13}
2^{14}
Number of states
State:Rank
softmax
2:1
4:1
8:1

Further Scaling on PTB with Dropout (Validation)



Speed Comparison[4]



[4] 2^{14} (16k) state SE-HMM takes 506 s/epoch on the same data

Discussion

- Reduced computation complexity of inference by 4x with a low rank assumption
- Scaling factor not as large as SE-HMM
- Validation PPL worse than SE-HMM

Conclusion

- Extended techniques from neural networks to HMMs
- Sped up inference by constraining structure in both the emission and transition matrices
- Demonstrated improvements in perplexity with larger state spaces

Future Work

- Explore the performance of more complex interpretable models
  - Hierarchical / Factorial HMMs
  - Probabilistic context-free grammars
  - Switching linear dynamical systems[5]
- Explore other structure for fast matrix-vector products and tensor generalizations
  - FFT-inspired algorithms[6]
- Learn sparsity constraints in SE-HMM

[5] Foerster et al., 'Intelligible Language Modeling with Input Switched Affine Networks'.

[6] Dao et al., 'Kaleidoscope: An Efficient, Learnable Representation For All Structured Linear Maps'.

EOS

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].

Dao, Tri et al. 'Kaleidoscope: An Efficient, Learnable Representation For All Structured Linear Maps'. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=BkgrBgSYDS.

Foerster, Jakob N. et al. 'Intelligible Language Modeling with Input Switched Affine Networks'. In: *CoRR* abs/1611.09434 (2016). arXiv: 1611.09434. URL: http://arxiv.org/abs/1611.09434.

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

Generalized Softmax

- Softmax

$$p(z_t \mid z_{t-1}) = \frac{\exp(\mathbf{u}_{z_{t-1}}^\top \mathbf{v}_{z_t})}{\sum_z \exp(\mathbf{u}_{z_{t-1}}^\top \mathbf{v}_z)}$$

- Generalized Softmax

$$p(z_t \mid z_{t-1}) = \frac{K(\mathbf{u}, \mathbf{v})}{\sum_z K(\mathbf{u}, \mathbf{v}_z)} = \frac{\phi(\mathbf{u})^\top \phi(\mathbf{v})}{\sum_z \phi(\mathbf{u})^\top \phi(\mathbf{v}_z)},$$

  for positive kernel $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$ and feature map $\phi : \mathbb{R}^D \to \mathbb{R}^R$

Generalized Softmax: Inference

- The key $O(Z^2)$ step in the forward algorithm:

$$p(z_t \mid x_{<t}) = \sum_{z_{t-1}} p(z_t \mid z_{t-1}) p(z_{t-1} \mid x_{<t})$$

- In matrix form,

$$\boldsymbol{\gamma}_t = \underbrace{\boldsymbol{\alpha}_{t-1}}_{\mathbb{R}^Z} \underbrace{\Lambda}_{\mathbb{R}^{Z \times Z}},$$

where we have the probability of the

$$\begin{aligned}
&\text{current state,} && [\boldsymbol{\gamma}_t]_{z_t} = p(z_t \mid x_{<t}), \\
&\text{last state,} && [\boldsymbol{\alpha}_{t-1}]_{z_{t-1}} = p(z_{t-1} \mid x_{<t}), \\
&\text{transition probability,} && [\Lambda]_{z_{t-1}, z_t} = p(z_t \mid z_{t-1})
\end{aligned}$$

Generalized Softmax: Inference

- Use generalized softmax in transition distribution

$$[\Lambda]_{z_{t-1},z_t} = p(z_t \mid z_{t-1}) \propto \phi(\mathbf{u}_{z_{t-1}})^\top \phi(\mathbf{v}_{z_t})$$

- Allows us to apply associative property of matrix multiplication

$$\begin{aligned}
\boldsymbol{\gamma}_t &= \boldsymbol{\alpha}_{t-1}\Lambda \\
&= \boldsymbol{\alpha}_{t-1}(\mathrm{diag}(d)\phi(U)\phi(V)^\top) \\
&= \underbrace{(\boldsymbol{\alpha}_{t-1} \circ d)}_{\mathbb{R}^Z}\underbrace{\phi(U)}_{\mathbb{R}^{Z\times f}}\underbrace{\phi(V)^\top}_{\mathbb{R}^{f\times Z}},
\end{aligned}$$

  with stacked embeddings $\phi(U), \phi(V) = [\phi(\mathbf{v}_1), \ldots, \phi(\mathbf{v}_Z)]$ and normalizing constants d

- Takes $O(Zf)$ time from left to right!