Scaling Hidden Markov Language Models

March 19, 2021

Abstract

Modern methods for language models based purely on neural networks are opaque and difficult to analyze, while alternative methods based on probabilistic graphical models are interpretable but not performant. We hypothesize that probabilistic methods are not as performant as neural methods because they have not been scaled to massive sizes. A significant barrier to scaling probabilistic methods is inference, which typically takes computation superlinear in the size of the model. In order to make probabilistic models performant while maintaining interpretability, we present techniques for circumventing the heavy computational cost of inference. We first focus on scaling one of the simplest models, hidden Markov models, then show our techniques generalize to more complex models, such as probabilistic context-free grammars.

1 Introduction

Neural network-based generative models have led to progress in difficult tasks such as language modeling and machine translation. However, this progress comes at the cost of interpretability. While neural networks are flexible function approximators, they are also opaque and difficult to analyze. This leads to difficulty in both trusting their predictions as well as using them to validate scientific hypothesis. There are three paths forward towards interpretability: The first is post-hoc test-time analysis of models [14], the second is training-time augmentations of models [], and the third is model-based approaches []. We explore the third path, formulating models that interpretable from the start. In particular, we focus on probabilistic graphical models, which allow for the execution of arbitrary probabilistic queries. These queries include the ability to calculate conditional and marginal probabilities given evidence. This interpretability comes at a cost: Performing these queries via inference is expensive in both time and space. The computational complexity of inference is a major impediment to scaling graphical models to large sizes, which we hypothesize prevents them from reaching the performance of neural networks. We seek to answer the question of whether graphical models can be both performant and interpretable.

We use language modeling as a benchmark task, as the complex and sometimes long-range phenomena in natural language provides a good testbed. Language model benchmarks are currently dominated by autoregressive neural models, which makes it difficult to analyze what properties of language the models are actually capturing.

In this work, we investigate scaling probabilistic graphical models which explicitly reason about latent variables. We focus on the one of the simplest latent variable models, the hidden Markov model (HMM). Despite the simplicity of HMMs, they are still computationally expensive to scale due to the cost of inference. In order to scale HMMs, we utilize a kernel-based generalized softmax that greatly reduces the complexity of inference. Before diving into the generalized softmax, we first review HMMs.

2 Hidden Markov Models for Language Modeling

Hidden Markov models (HMMs) have a rich history in natural language processing. They have been used for a series of tasks, including speech recognition [8], part of speech tagging [6], and word alignment in machine translation [15]. They have also been used as components in neural models, such as in attention [10]. We focus on applying HMMs to the task of language modeling, where the goal is to model the sequence of tokens in a sentence $x = (x_1, \ldots, x_T)$.

HMMs are one of the simplest latent variable models for language modeling with a per-token latent variable. The simplicity is a result of the strong conditional independence assumptions: Every timestep has a single discrete state variable used to represent context, which is then the only information used to emit a token. Additionally, the next state is a function of only the previous state. These independence assumptions result in a bottleneck where all information must flow through the single

state variable at each timestep, causing performance to be limited by the number of states. Limiting dependencies in this way also gives the model interpretability. (Expand on what that means)

Formally, HMMs have the following generative process: for each $t \in [T]$ timestep, first choose a latent state $z_t \in \mathcal{Z}$, where $|\mathcal{Z}|$ is the number of latent states, then choose a token to emit $x_t \in \mathcal{X}$. This defines the joint distribution:

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

with transitions $p(z_t \mid z_{t-1})$, emissions $p(x_t \mid z_t)$, and a distinguished start state $z_0 = S$.

Training an HMM requires marginalizing over the unobserved sequence $z = (z_1, \ldots, z_T)$ in order to obtain the evidence, $p(x) = \sum_z p(x, z)$, via the forward algorithm. The forward algorithm can be written as a sequence of matrix-vector multiplications: Let the start vector, $\boldsymbol{\pi} \in \mathbb{R}^{|\mathcal{Z}|}$, have entries

$$[\boldsymbol{\pi}]_{z_1} = p(x_1 \mid z_1) p(z_1 \mid z_0 = S), \tag{2}$$

and the subsequent transition operators, $\Lambda_t \in \mathbb{R}^{|\mathcal{Z}| \times |\mathcal{Z}|}$ for $t \in [2, \ldots, T]$, have entries

$$[\Lambda_t]_{z_{t-1}, z_t} = p(x_t \mid z_t) p(z_t \mid z_{t-1}). \tag{3}$$

The evidence is then given by

$$p(x) = \boldsymbol{\pi}^\top \Lambda_2 \cdots \Lambda_T \mathbf{1}, \tag{4}$$

where $\mathbf{1} \in \mathbb{R}^{|\mathcal{Z}| \times 1}$ is the column vector of all ones.

On a serial machine, the cost of computing each matrix-vector product in the above equation takes time $O(|\mathcal{Z}|^2)$, resulting in a total running time of $O(T|\mathcal{Z}|^2)$ for the forward algorithm. The quadratic dependence on the number of states precludes scaling to extremely large state spaces.

In the following sections we will introduce techniques that aim to reduce the dependence of the time and space complexity of inference on the size of the state space.

3 Generalized Softmax with Kernels

Parameterization of the probability distributions in a model affects both computational performance as well as generalization. Recent work on efficient attention in neural models shows that careful choice of parameterization of the softmax kernel, the main component of attention, results in large computational gains at little cost in accuracy [3, 7]. In this section we will cover generalized softmax with kernels and its use in efficient inference.

Softmax is commonly used to parameterize conditional distributions in neural nets, and has origins in smooth approximations of argmax as well as the conditional max entropy problem []. Focusing on the transition distribution $p(z_t \mid z_{t-1})$, the softmax parameterization is as follows:

$$p(z_t \mid z_{t-1}) = \frac{\exp(\mathbf{u}_{z_t-1}^\top \mathbf{v}_{z_{t-1}})}{\sum_{z'} \exp(\mathbf{u}_{z_{t-1}}^\top \mathbf{v}_{z'})},$$

with $\mathbf{u}_{z_{t-1}}, \mathbf{v}_{z_t} \in \mathbb{R}^d$ vector-space embeddings of values of z_{t-1}, z_t.[1] The numerator, $\exp(\mathbf{u}_{z_{t-1}}^\top \mathbf{v}_{z_t})$, is the exponential kernel [9].

Although the parameterization of softmax is specific to the exponential kernel, softmax can be generalized with arbitrary nonnegative kernels $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$. Replacing the exponential kernel with the nonnegative kernel K yields generalized softmax,

$$p(z_t \mid z_{t-1}) = \frac{K(\mathbf{u}_{z_{t-1}}, \mathbf{v}_{z_t})}{\sum_{z'} K(\mathbf{u}_{z_{t-1}}, \mathbf{v}_{z'})}, \tag{5}$$

where we again have embeddings $\mathbf{u}_{z_{t-1}}, \mathbf{v}_{z_t} \in \mathbb{R}^d$.[2]

Kernels have a long history in machine learning, where they are used to extend linear classification models to nonlinear feature spaces, such as in support vector machines or regression. In those settings, kernels are intuitively used to measure the similarity between two data points, hinging on a connection to an inner product in a reproducing kernel Hilbert space. In the generalized softmax setting, kernels instead parameterize conditional distributions by measuring the propensity of one random variable taking a particular value given the value of another random variable. However, the connection to inner products in reproducing kernel Hilbert spaces is still useful. In particular, the

[1] Softmax also has a temperature parameter that rescales the dot product, which we omit.

[2] The term generalized softmax is inspired by generalized linear models, which generalized linear models to non-Gaussian errors with a link function, similar to the use of kernels in generalized softmax.

connection guarantees the existence of a feature map $\phi : \mathbb{R}^d \to \mathbb{F}$, a map from the input space $\mathbb{R}^d$ to the feature space $\mathbb{F}$, which allows us to write generalized softmax in matrix form:

$$p(z_t \mid z_{t-1}) = [\text{diag}(\mathbf{d})\phi(U)\phi(V)^\top]_{z_{t-1}, z_t}, \tag{6}$$

where we have the stacked embeddings $\phi(U), \phi(V) \in \mathbb{R}^{|\mathcal{Z}| \times \dim(\mathbb{F})}$ and the normalizing constants $\mathbf{d} \in \mathbb{R}^{|\mathcal{Z}|}$,

$$\begin{aligned}
\phi(U) &= [\phi(\mathbf{u}_1)^\top, \phi(\mathbf{u}_2)^\top, \ldots] \in \mathbb{R}^{|\mathcal{Z}| \times \dim(\mathbb{F})}, \\
\phi(V) &= [\phi(\mathbf{v}_1)^\top, \phi(\mathbf{v}_2)^\top, \ldots] \in \mathbb{R}^{|\mathcal{Z}| \times \dim(\mathbb{F})}, \\
[\mathbf{d}]_{z_{t-1}} &= \frac{1}{\phi(\mathbf{u}_{z_{t-1}})^\top \sum_z \phi(\mathbf{v}_z)}.
\end{aligned}$$

We will rely on the dimension of the feature space being small relative to the number of states, $\dim(\mathbb{F}) < |\mathcal{Z}|$, in order to improve the time and space complexity of inference.[3]

4 Inference with Generalized Softmax

Unfortunately, in the case of softmax, which relies on the exponential kernel, the feature space $\mathbb{F}_{\exp}$ has infinite dimension. This can be seen from the Taylor expansion:

$$\exp(\mathbf{u}^\top \mathbf{v}) = \sum_{n=0}^{\infty} \frac{(\mathbf{u}^\top \mathbf{v})^n}{n!} = \sum_{n=0}^{\infty} \frac{(\sum_{i=1}^{d} [\mathbf{u}]_i [\mathbf{v}]_i)^n}{n!} = \sum_{n=0}^{\infty} \frac{\sum_{\mathbf{j} \in [d]^n} (\prod_{i=1}^{n} [\mathbf{u}]_{[\mathbf{j}]_i})(\prod_{i=1}^{n} [\mathbf{v}]_{[\mathbf{j}]_i})}{n!},$$

where we have expanded the terms of the numerator by summing over all subsets $\mathbf{j}$ of size n elements of $[d]$, including repeats and different orderings. This yields a feature map of

$$[\phi(\mathbf{u})]_{n,\mathbf{j}} = \frac{\prod_{i=1}^{n} [\mathbf{u}]_{[\mathbf{j}]_i}}{\sqrt{n!}}$$

for all $n \in \mathbb{N}$.[4] We can therefore express the exponential kernel as an inner product of infinite dimensional vectors [4],

$$\exp(\mathbf{u}^\top \mathbf{v}) = \sum_{n=0}^{\infty} \sum_{\mathbf{j} \in [d]^n} [\phi(\mathbf{u})]_{n,\mathbf{j}} [\phi(\mathbf{v})]_{n,\mathbf{j}} = \langle \phi(\mathbf{u}), \phi(\mathbf{v}) \rangle,$$

where the dimension of the feature space $\dim(\mathbb{F}_{\exp}) = \dim(\phi(\mathbf{u})) = \infty$. Since the infinite dimensional feature space of the exponential kernel is larger than the finite $|\mathcal{Z}|$, recent work has instead turned to approximations of the exponential kernel based on random Fourier features [3, 7, 9]. These approaches use a finite-dimensional unbiased estimator of the exponential kernel.

We propose to take a different strategy, where we directly learn a feature map for the generalized softmax while disregarding the explicit functional form of the kernel. This allows us to use the matrix representation of generalized softmax to efficiently perform inference.

Previously, Equation 4 shows how to compute the evidence, $p(x)$, with a series of matrix-vector products. This was done by first defining the transition operator, Λ_t as in Equations 3 and 2. We can perform a similar procedure with the transition distribution using generalized softmax. With a generalized softmax parameterization of the transition distribution and a feature space $\mathbb{F}_\phi$ with feature map ϕ and finite dimension $\dim(\mathbb{F})$, we have the transition operators for all $t \in [2, \ldots, T]$,

$$\Lambda_t = \underbrace{\text{diag}(\mathbf{d})\phi(U)}_{L_t} \underbrace{\phi(V)^\top \text{diag}(\mathbf{o}_t)}_{R_t}, \tag{7}$$

where the entries of $\mathbf{o}_t \in \mathbb{R}^{|\mathcal{Z}|}$ are given by

$$[\mathbf{o}_t]_{z_t} = p(x_t \mid z_t),$$

the emission probability and $\mathbf{d} \in \mathbb{R}^{|\mathcal{Z}|}$ contains the normalizing constant for each state. We can interpret $L_t \in \mathbb{R}^{|\mathcal{Z}| \times \dim \mathbb{F}}$ as a projection into feature space, and $R_t \in \mathbb{R}^{\dim(\mathbb{F}) \times |\mathcal{Z}|}$ as a projection back into state space. The evidence can then be computed using L_t and R_t as follows:

$$p(x) = \boldsymbol{\pi} \Lambda_2 \cdots \Lambda_T \mathbf{1} = \boldsymbol{\pi}(L_2 R_2) \cdots (L_T R_T)\mathbf{1} = \boldsymbol{\pi} L_2 R_2 \cdots L_T R_T \mathbf{1}, \tag{8}$$

[3] Matrix multiplication with a diagonal matrix is equivalent to rescaling rows, and is therefore computationally cheap.
[4] While this is one possible feature map, there may exist multiple valid feature maps.

where we have substituted Equation 7 into the computation of the evidence and applied the associative property of matrix multiplication. When performed from left to right, the matrix vector products now take $O(|\mathcal{Z}|\dim(\mathbb{F}))$ time, rather than $O(|\mathcal{Z}|^2)$, with the overall complexity of inference now linear in the number of states, $O(T|\mathcal{Z}|\dim(\mathbb{F}))$, with generalized softmax.

We parameterize the feature map $\phi(\mathbf{u}) = \exp(W_\phi \mathbf{u} - \frac{\|\mathbf{u}\|_2^2}{2})$, with paramater $W_\phi \in \mathbb{R}^{\dim(\mathbb{F})\times d}$. We then train all parameters, including W_ϕ and the parameters of the transition and emission distributions,[5] by optimizing the evidence with gradient ascent.

5 Connection to Kernel Mean Embeddings and Kernel Belief Propagation

A related line of work uses kernel methods to extend belief propagation. Belief propagation is commonly constrained to models with parametric assumptions such as small, discrete or Gaussian conditional distributions. Kernel belief propagation (KBP) is a method that utilizes reproducing kernels to run inference without the common parametric assumptions [12, 13]. At a high level, KBP rewrites the messages in belief propagation as conditional expectations, then applies kernel-based conditional embedding operators to approximate the conditional expectations nonparametrically. Whereas generalized softmax can be used to directly parameterize conditional distributions, KBP instead relies on feature space operators in a manner that resembles Bayes' rule. The generality of KBP comes at a cost: in order to avoid parametric assumptions the operations in KBP are much more expensive than generalized softmax.

In this section, we will derive the kernel conditional embedding operator used in KBP and compare it to generalized softmax. However, before covering the conditional embedding operator, we first detail the kernel mean embedding. Rather than computing conditional expectations, the simpler kernel mean embedding computes unconditional expectations. The kernel mean embedding extends the reproducing property, which says that function application can be expressed as an inner product, by showing that the expected values of functions of x can also be computed via inner products.

Given a nonnegative kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$,[6] there exists an associated reproducing kernel Hilbert space (RKHS) $\mathbb{F}$ which we call the feature space.[7] The RKHS has a feature mapping, $\phi_x : \mathcal{X} \to \mathbb{F}$, such that, for all functions $f \in \mathbb{F}$, function application can be expressed with the inner product $\langle \cdot, \cdot \rangle_\mathbb{F} : \mathbb{F} \times \mathbb{F} \to \mathbb{R}_+$:

$$f(x) = \langle f, \phi_x(x) \rangle_\mathbb{F}.$$

This is known as the reproducing property.[8] We will omit the space associated with the inner product as well as the subscript of the feature mapping when it is clear from context.

The kernel mean embedding [11], $\mu_x \in \mathbb{F}$, associated with the $\mathcal{X}$-valued random variable x is defined as

$$\mu_x = \mathbb{E}_x \left[\phi(x)\right]. \tag{9}$$

It has the property that, for all $f \in \mathbb{F}$,

$$\mathbb{E}_x \left[f(x)\right] = \mathbb{E}_x \left[\langle f, \phi(x) \rangle\right] = \langle f, \mathbb{E}_x \left[\phi(x)\right] \rangle = \langle f, \mu_x \rangle \tag{10}$$

by the reproducing property and linearity of expectation. (to be super rigorous i think you have to show expectation is a self-adjoint linear operator) This extends the reproducing property, which says that function application can be expressed as an inner product, by showing that the expected values of functions of x can also be computed via inner products. (repeated)

Example Let $x \sim N(\mu, \Sigma)$ be a d-dimensional Gaussian, feature mapping $\phi(x) = x$, and inner product $\langle x, y \rangle = x^\top y$. The kernel mean embedding of x is given by $\mu_x = \mathbb{E}_x \left[\phi(x)\right] = \mu$. Let function $f(x) = w^\top x$, with $w \in \mathbb{R}^d$. We then have $\mathbb{E}_x \left[f(x)\right] = \langle w, \mu_x \rangle = w^\top \mu$.

Example Let $x \sim \text{Cat}(\theta)$ be a d-dimensional categorical distribution, the one-hot feature mapping $\phi(x) = e_x \in \mathbb{R}^d$, and inner product $\langle x, y \rangle = x^\top y$. The kernel mean embedding of x is then given by $\mu_x = \mathbb{E}\left[\phi(x)\right] = \theta$. We can then model a single instance of attention as an expectation under this framework. Let the function $f(x) = \mathbf{u}_x \in \mathbb{R}^n$ map values of x to their corresponding embedding. With some abuse of notation, given the stacked $f = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_d] \in \mathbb{R}^{d\times n}$, where n is the embedding dimension, we then have $\mathbb{E}_x \left[f(x)\right] = \langle f, \mu_x \rangle = f^\top \theta$.[9]

Make sure notation is consistent

Also potentially clarify that no speedups are possible here due to only a single instance of attn. Performer/RFA

[5] We give the full parameterization of all distributions in the appendix.

[6] A strictly positive kernel function must have the finitely positive semidefinite kernel matrix property, allowing us to apply the Moore-Aronszajn theorem, which constructively guarantees the existence of an associated RKHS.

[7] Check if we need separability.

[8] Show reproducing property does not hold for L_2? This is known as the reproducing property of $\phi(x) \in \mathbb{F}$. Terminology of reproducing property vs point evaluation property is murky. Pick one and stay consistent.

[9] The expectation is applied to each dimension of the codomain of f separately.

Both of the above examples assume the mean parameters are known, which requires parametric assumptions on the random variable x. [10]

We can then extend the kernel mean embedding to the conditional setting by introducing covariance operators, which offer an alternative representation for covariance using Hilbert spaces [1, 2]. Given $\mathcal{X}$ and $\mathcal{Y}$-valued RVs x, y, associated RKHS $\mathbb{F}_x, \mathbb{F}_y$, and respective feature mappings $\phi_x : \mathcal{X} \to \mathbb{F}_x, \phi_y : \mathcal{Y} \to \mathbb{F}_y$. We omit the feature mapping subscripts when they are clear from context. Let the outer product $(g \otimes h)f = \langle f, h\rangle g$ for elements $f, h \in \mathbb{F}_x$ and $g \in \mathbb{F}_y$. The uncentered cross-covariance operator $C_{yx} : \mathbb{F}_x \to \mathbb{F}_y$ is defined by

$$C_{yx} = \mathbb{E}_{yx}\left[\phi_y(y) \otimes \phi_x(x)\right], \tag{11}$$

for all $f \in \mathbb{H}, g \in \mathbb{G}$ where $\mathbb{H}, \mathbb{G}$ are RKHSs over $\mathcal{X}, \mathcal{Y}$.[11] The uncentered cross-covariance operator has the following property:

different symbol for feature mapping for y

$$\begin{aligned}
\langle g, C_{yx} f\rangle &= \langle g, \mathbb{E}\left[(\phi(y) \otimes \phi(x))\right] f\rangle && \text{(defn of } C_{yx}\text{)} \\
&= \mathbb{E}\left[\langle g, (\phi(y) \otimes \phi(x)) f\rangle\right] && \text{(linearity of expectation)} \\
&= \mathbb{E}\left[\langle g, \langle f, \phi(x)\rangle \phi(y)\rangle\right] && \text{(defn of } \otimes\text{)} \\
&= \mathbb{E}\left[\langle f, \phi(x)\rangle \langle g, \phi(y)\rangle\right] && \text{(linearity of } \langle\cdot,\cdot\rangle\text{)} \\
&= \mathbb{E}_{xy}\left[f(x) g(y)\right], && \text{(reproducing property)}
\end{aligned} \tag{12}$$

which is an extension of the kernel mean embedding to a joint distribution. The operator $C_{xx} : \mathbb{F}_x \to \mathbb{F}_x$ is called the covariance operator, as it takes the form $C_{xx} = \mathbb{E}_x\left[\phi(x) \otimes \phi(x)\right]$.

Example Let $(x, y) \sim N(0, \Sigma)$ bivariate Gaussian, where $\Sigma_{xy} = \mathbb{E}\left[xy\right]$, the off-diagonal entry of the covariance matrix. If we take $\mathbb{F}_x = \mathbb{F}_y = \mathbb{R}^d$ with ϕ_x, ϕ_y as the identity functions, then $C_{xy} = \Sigma_{xy}$.

Example Let $(x, y) \sim p(x, y)$, a discrete joint distribution, with $\phi(x) = e_x, \phi(y) = e_y$ both mapping x, y to one-hot representations. This results in $[C_{xy}]_{x,y} = p(x, y)$. If we also let $f = e_{x'}, g = e_{y'}$ be one-hot representations of particular values of x and y, then $\langle f, C_{xy} g\rangle = \langle C_{yx} f, g\rangle = p(x = x', y = y')$.

Given the cross-covariance and covariance operators, C_{yx} and C_{xx}, we can now define the conditional embedding operator. The conditional embedding operator uses the above covariance operators in a manner similar to computing conditional probabilities via Bayes' rule: $p(y \mid x) = p(y, x)/p(x)$. We define the conditional embedding operator $U_{y|x} : \mathbb{F}_x \to \mathbb{F}_y$ as follows:

$$U_{y|x} = C_{yx} C_{xx}^{-1}. \tag{13}$$

The conditional embedding operator allows conditional expectations to be expressed as inner products. For all $g \in \mathbb{F}_y$,

$$\langle g, U_{y|x} \phi(x)\rangle = \mathbb{E}_{y|x}\left[g(y)\right]. \tag{14}$$

The correctness of this property can be shown as follows: First, we apply the property of the covariance operator derived in Equation 12. For all $f \in \mathbb{F}_x$, we have

$$\langle f, C_{xx} \mathbb{E}_{y|x}\left[g(y)\right]\rangle = \mathbb{E}_x\left[f(x) \mathbb{E}_{y|x}\left[g(y) \mid x\right]\right] = \mathbb{E}_{xy}\left[f(x) g(y)\right] = \langle f, C_{xy} g\rangle. \tag{15}$$

With some care,[12] we can then multiply the second terms of the inner products on both sides of the equation by the left inverse C_{xx}^{-1}. This gives, forall $f \in \mathbb{F}_x$,

$$\langle f, \mathbb{E}_{y|x}\left[g(y)\right]\rangle = \langle f, C_{xx}^{-1} C_{xy} g\rangle = \langle g, C_{yx} C_{xx}^{-1} f\rangle.$$

As this holds for any $f \in \mathbb{F}_x$, it also holds for ϕ_x, yielding

$$\mathbb{E}_{y|x}\left[g(y)\right] = \langle g, U_{y|x} \phi(x)\rangle$$

as desired, by the reproducing property.

Example Consider discrete RVs x and y, with $y \sim p(y \mid x)$. We would like to compute $\mathbb{E}_{y|x}\left[g(y)\right]$ using the conditional embedding operator. Assuming $\mathbb{F}_x = \mathbb{F}_y = \mathbb{R}^d$, with inner products $\langle\cdot,\cdot\rangle_{\mathbb{F}_x} = \langle\cdot,\cdot\rangle_{\mathbb{F}_x}$ both equal to the dot product, the conditional expectation is given by

$$\mathbb{E}_{y|x}\left[g(y)\right] = g^\top U_{y|x} \phi(x) = g^\top C_{yx} C_{xx}^{-1} \phi(x).$$

[10] This can be extended to the nonparametric setting by estimating the kernel mean embedding via samples, $\mu_x = \mathbb{E}\left[\phi(x)\right] \approx \frac{1}{n}\sum_{i=1}^n \phi(x_i), x_i \overset{\text{iid}}{\sim} p(x)$.

[11] The uncentered terminology is a result of the definition of the (centered) cross-covariance as $\mathbb{E}\left[(x - \mathbb{E}\left[x\right])(y - \mathbb{E}\left[y\right])\right] = \mathbb{E}\left[x \otimes y\right] - \mathbb{E}\left[x\right]\mathbb{E}\left[y\right]^\top$, and the uncentered cross-covariance $\mathbb{E}\left[x \otimes y\right]$.

[12] See the appendix of Fukumizu et al. [5] for the full proof

With this representation, it takes computation $O((|\mathcal{Y}| + |\mathcal{X}|)d^2)$ to compute C_{yx} and C_{xx} and $O(d^3)$ to invert C_{xx}. This is more expensive than generalized softmax, which takes $O(|\mathcal{Y}|d)$ computation. The improved efficiency of generalized softmax comes from directly parameterizing the conditional distribution $p(y \mid x)$.

(End current progress) Is there a one sentence answer as to why we cannot estimate $p(y|x)$ with kernels directly? Something like, in general it is hard to model conditional expectations, but generalized softmax makes a particular assumption (tractable partition function, closed form expression for mean) that makes it easy.

Similarly, inference in an HMM can also be written with conditional expectations. At each timestep, the forward algorithm recursively computes the quantity

$$p(z_t \mid x_{<t}) = \sum_{z_{t-1}} p(z_t \mid z_{t-1}) p(z_{t-1} \mid x_{<t}) = \mathbb{E}_{z_{t-1}|x_{<t}} \left[p(z_t \mid z_{t-1}) \right]. \tag{16}$$

Applying the definition of generalized softmax to Equation 16 and rewriting $[\boldsymbol{\alpha}_t]_{z_t} = p(z_t \mid x_{\leq t})$ we have

$$\boldsymbol{\alpha}_t = \operatorname{diag}(d)\phi(U)\phi(V)^\top \boldsymbol{\alpha}_{t-1}, \tag{17}$$

by Equation 6. Identical to the analysis in the previous section, this reduces the computation performed at each timestep from $O(|\mathcal{Z}|^2)$ to $O(|\mathcal{Z}| \dim(\mathbb{F}))$. This can then be used in conjunction with $p(x_t \mid z_t)$ to compute the quantities

$$p(x_t \mid x_{<t}) = \sum_{z_t} p(x_t \mid z_t) p(z_t \mid z_{<t}), \qquad p(z_t \mid x_{\leq t}) = \frac{p(x_t, z_t \mid x_{<t})}{p(x_t \mid x_{<t})}, \qquad p(x) = \prod_t p(x_t \mid x_{<t}),$$

again yielding $O(T|\mathcal{Z}| \dim(\mathbb{F}))$ computation for computing the evidence.

However, the similarities between our work and KBP end there. KBP focuses on using the kernel matrix representation for nonparametric inference in non-Gaussian graphical models where exact computation of conditional expectations is infeasible. We eschew the explicit kernel representation and instead use the inner product in feature space explicitly in order to get computational speedups, which is made possible by the tractability of the conditional expectation.

6 Random Fourier Features

Scaling factor of performer versus ours (length vs num features / embedding dimension).

Performer (protein): num features = 256, emb dim = 512, length = 12k? MSE graph. Performer (toy mse): num features = 128, emb dim = 16, length = 4096? HMM: num features = length / 4, emb dim = 256

Performer and random feature attention find that num features does not scale with length. How can we explain this?

Quote from performer: 'The dependence on the radius means that the length of queries and keys cannot grow at a fixed number of features if we want to retain the quality of the approximation. In particular, this means that FAVOR cannot approximate hard attention on sequences of unlimited length with a fixed number of features.' Poorly written. Interpretation: Random projections from embedding space to sphere. If we increase the number of points in embedding space while keeping their projections separated, we have to increase the volume of the sphere.

Question: why dont they see that in their experiments? Possibly get away with a low degree of separation.

Explain argument from separation to entropy or rank.

7 Emission Sparsity Constraints

8 Efficient Sampling

9 Spectral Methods

References

[1] Charles R. Baker. Mutual information for gaussian processes. *SIAM Journal on Applied Mathematics*, 19(2):451–458, 1970. doi: 10.1137/0119044. URL `https://doi.org/10.1137/0119044`.

[2] Charles R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973. ISSN 00029947. URL http://www.jstor.org/stable/1996566.

[3] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2020.

[4] Andrew Cotter, Joseph Keshet, and Nathan Srebro. Explicit approximations of the gaussian kernel. *CoRR*, abs/1109.4603, 2011. URL http://arxiv.org/abs/1109.4603.

[5] Kenji Fukumizu, Francis Bach, and Michael Jordan. Kernel dimensionality reduction for supervised learning. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2004. URL https://proceedings.neurips.cc/paper/2003/file/84b20b1f5a0d103f5710bb67a043cd78-Paper.pdf.

[6] Bernard Merialdo. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–171, 1994. URL https://www.aclweb.org/anthology/J94-2001.

[7] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=QtTKTdVrFBB.

[8] Lawrence R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, page 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. ISBN 1558601244.

[9] Ankit Singh Rawat, Jiecao Chen, Felix Xinnan X Yu, Ananda Theertha Suresh, and Sanjiv Kumar. Sampled softmax with random fourier features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 13857–13867. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/e43739bba7cdb577e9e3e4e42447f5a5-Paper.pdf.

[10] Shiv Shankar and Sunita Sarawagi. Posterior attention models for sequence to sequence learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BkltNhC9FX.

[11] Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In Marcus Hutter, Rocco A. Servedio, and Eiji Takimoto, editors, *Algorithmic Learning Theory*, pages 13–31, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-75225-7.

[12] Le Song, Arthur Gretton, and Carlos Guestrin. Nonparametric tree graphical models. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 765–772, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL http://proceedings.mlr.press/v9/song10a.html.

[13] Le Song, Arthur Gretton, Danny Bickson, Yucheng Low, and Carlos Guestrin. Kernel belief propagation, 2011.

[14] Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *Association for Computational Linguistics*, 2019. URL https://arxiv.org/abs/1905.05950.

[15] Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, page 836–841, USA, 1996. Association for Computational Linguistics. doi: 10.3115/993268.993313. URL https://doi.org/10.3115/993268.993313.