The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Leaf-Smoothed Hierarchical Softmax for Ordinal Prediction

Wesley Tansey[*]
Columbia University
New York, NY 10027
wt2274@cumc.columbia.edu

Karl Pichotta,[†] **James G. Scott**[‡§]
University of Texas at Austin
Austin, TX 78712
pichotta@cs.utexas.edu
james.scott@mccombs.utexas.edu

Abstract

We propose a new approach to conditional probability estimation for ordinal labels. First, we present a specialized hierarchical softmax variant inspired by k-d trees that leverages the inherent spatial structure of (potentially-multivariate) ordinal labels. We then adapt ideas from signal processing on noisy graphs to develop a novel regularizer for such hierarchical softmax models. Both our tree structure and regularizer independently boost the sample efficiency of a deep learning model across a series of simulation studies. Furthermore, the combination of these two techniques produces additive gains and the model does not suffer from the pathologies of other approaches in the literature. We validate our approach empirically on a suite of real-world datasets, in some cases reducing the error by nearly half in comparison to other popular methods in the literature. Our results demonstrate that our method is a powerful new modeling technique for conditional probability estimation of ordinal labels, especially in the low-to-mid sample size regimes such as those often found in biological and other physical sciences.

1 Introduction

Recently there has been a flurry of interest in using deep learning methods for conditional probability estimate (CPE). The applications of such models cover a wide variety of scientific areas, from cosmology (Ravanbakhsh et al. 2016) to health care (Ranganath et al. 2016; Ng et al. 2017). A subset of this area deals specifically with discrete conditional distributions for ordinal labels, where an explicit estimation of the likelihood is desired—as opposed to simply the ability to sample the distribution, as with GAN-based models (Goodfellow et al. 2014). Deep learning models that predict such distributions have achieved state-of-the-art results in text-to-speech synthesis (van den Oord et al. 2016a), image generation (van den Oord et al. 2016b; van den Oord, Kalchbrenner, and Kavukcuoglu 2016; van den Oord et al. 2016c; Gulrajani et al. 2016; Salimans et al. 2017), image super-resolution (Dahl, Norouzi, and Shlens 2017), image colorization (Deshpande et al. 2016), and EHR survival modeling (Ranganath et al. 2016). There have been broad calls from other fields for methods that efficiently handle more moderate numbers of samples, as in medical imaging, where one predicts diagnostic scores from histological, MRI, or other high-dimensional medical data that is very expensive to gather (Greenspan, van Ginneken, and Summers 2016). Addressing this challenge is crucial for adapting deep learning from the engineering realm of "big data" to the scientific realm, frequently characterized by smaller high-quality datasets.

In this paper we focus on the form of the output layer for CPE models that predict (potentially-multivariate) ordinal labels, such as the next waveform (1d) in an audio clip or the next pixel (1d or 3d) in an image. Typically, this distribution is modeled by the output layer of a deep neural network and contains either the logits of a multinomial distribution or the parameters of a mixture model, such as a Gaussian mixture model (mixture density networks, see (Bishop 1994)). Previous work (van den Oord et al. 2016a; 2016b) has found empirically that using a multinomial model often outperforms GMMs on ordinal labels. Methods to improve performance over the naive multinomial model often involve domain-specific heuristic compression of the space into a smaller number of bins (van den Oord et al. 2016a) or hand-crafting a mixture model to better-suit the marginal distribution of the data (Salimans et al. 2017).

We propose Leaf-Smoothed Hierarchical Softmax (LSHS) as a flexible, general alternative model. LSHS leverages the structure of the ordinal label space by first using a hierarchical decomposition of the output probabilities in a manner similar to kd-trees. This increases the overall sample efficiency of the model, but increases the error at labels with a neighbor close in ordinal space but far in leaf-to-leaf path length along the tree. To overcome this, we draw from the signal processing literature to develop a graph-based trend filtering regularizer that locally smooths the area around each target value. These two techniques leverage the spatial structure in the discrete distribution to enable points to borrow statistical strength from their nearby neighbors, improving the estimation of the latent conditional distribution.

This paper makes the following novel contributions:

- An in-depth analysis of existing output models for ordinal CPE models, including an exploration of failure modes of each model.

[*]Department of Systems Biology
[†]Department of Computer Science
[‡]Department of Information, Risk, and Operations Management
[§]Department of Statistics and Data Sciences

- A scalable graph-based regularization approach for large (finely discretized) domains.
- Leaf-Smoothed Hierarchical Softmax: A new non-parametric approach for estimating discrete probability distributions with deep learning models. LSHS does not suffer from any of the biases identified in previous approaches, and it out-performs all previous models on both synthetic and real datasets.

The remainder of this paper proceeds as follows. Section 2 outlines our hierarchical softmax variant. Section 3 details our approach to smoothing the discrete probability space via graph-based trend filtering. Section 4 shows our experimental analysis of LSHS and our benchmarks confirming its strong performance. Section 5 provides a discussion of related work and the limitations of our model. Finally, Section 6 gives concluding remarks.

2 Hierarchical softmax variant

Our model relies on representing the target ordinal distribution using a tree rather than a flat set of class probabilities. Tree-based models for distribution estimation have a long history in machine learning. This includes seminal work using *k-d* trees for nonparametric density estimation (Gray and Moore 2003) and hierarchical softmax (HS) for neural language models (Morin and Bengio 2005). We draw inspiration from these past works in the choice of our hierarchical softmax model, essentially combining the space-splitting strategy of the former with the hierarchical CPE model of the latter to yield a fast, sample-efficient baseline architecture.

Dyadic decomposition

Rather than outputting the logits of a multinomial distribution directly, we instead create a balanced binary tree with its root node in the center of the ordinal space. From the root, we recursively partition the space into a series of half spaces, resulting in $n-1$ nodes for a discrete space of size n. The deep learning model then outputs the splitting probabilities for every node, $\mathcal{E}_i$, parameterized as the logits in a series of independent binary classification tasks,

$$p(y > b_i|x) = \frac{1}{1+\exp(-\mathcal{E}_i)}\,, \tag{1}$$

where b_i is the center of the node.

Figure 1 illustrates the hierarchical softmax approach. For ease of exposition, we denote the conditional probabilty of y being greater than the node value as simply $p(N)$ for a given node N. For a target value of $y_i = 4$ with some training example x_i, we calculate the log probability during training as $\log(p(y_i = 4|x_i)) = \log\left[p(A)(1-p(C))(1-p(F))\right]$. The training objective for the model is then the sum of the log probabilities of the training data.

There are several computational advantages to using this particular structure compared to a multinomial. For large spaces, multinomial models typically require some form of negative sampling (Mikolov et al. 2013; Jean et al. 2014) at training time to remain copmutationally efficient. In the HS model, however, every split is conditionally independent of the rest of the tree and there is no partition function to estimate. Instead, we only require the $\mathcal{O}(\log_2 n)$ path from the root to the target node to estimate the probability of a given training example. Using a balanced tree also guarantees that every path has a fixed-length of $\lceil \log_2 n \rceil$, making vectorization on a GPU straightforward. Finally, the resulting computations are more numerically stable in the case of positive dependency between adjacent labels, and rarely result in very small or large log-probabilities, a problem that often arises in both multinomial and mixture-model approaches.

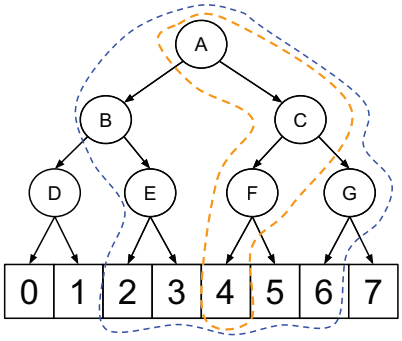


Figure 1: An illustration of our algorithm. The ordinal space is recursively partitioned into a series of binary left-right splits and the model outputs the splitting probability for each node. During training, computing the log probability of a target label only requires calculating the path to the label in the tree and its local smoothing neighborhood. In the single-dimensional example above, the target label is 4 and the neighborhood radius is 2, resulting in the need to calculate the target path (orange) and the paths of the surrounding 2 labels on each side (blue). As the size of the ordinal space grows larger, especially in multi-dimensional spaces, the computational savings of this approach become substantial.

Multiple dimensions

We extend the HS approach to multi-dimensional distributions in a manner similar to a balanced *k-d* tree. We enumerate the splits in the tree in a breadth-first fashion, alternating dimensions at each level of the tree. This has two distinct advantages over a depth-first approach of enumerating the first dimension before proceeding to the next dimension. The breadth-first approach means that all nodes close in Euclidean space will share more coarse-grained parents. This makes training more efficient by imposing a more principled notion of structure on the ordinal space. It also improves computational efficiency for the local smoothing strategy described in Section 3, as nearby values have heavily overlapping paths; this results in a well-utilized GPU cache when training.

We note that while many of the computational advantages of a generic HS approach are well-known (Morin and Bengio 2005), the sample efficiency of the structure is our top priority. Anecdotally, for instance, we have found that a balanced binary tree that begins from the center of the space and expands breadth-wise tends to be more sample efficient and robust across a wider range of problems.

3 Leaf smoothing

Our HS approach leverages the coarse-to-fine-grained spatial structure of the ordinal space through a multiresolution hierarchical decomposition. While this allows for larger-scale blocks of space to effectively share statistical strength, it also leads to local biases between nearby pairs of labels that lay on opposite sides of a high-level tree split. Consider the example from Figure 1. A sample of $y_i = 4$ is likely to result in an increase in probability of $p(y_i = 5|x_i)$ as well, since both A and C will increase in the direction of 5 and only F will be downweighted. However, it will clearly *decrease* the likelihood of $p(y_i = 3|x_i)$, since it will shift the probability of A away from 3 and leave the other nodes in the path of 3 unchanged. This imabalance in updates leads to a jagged distribution of errors in the estimation of the underlying conditional distribution.

Figure 2 shows a concrete example of the bias introduced by the HS model in a 64-label, 1000-sample, 1d experiment when the labels have positive dependence similar to that found in many ordinal prediction scenarios; labels 15, 31, and 47 all have outsized error due to being on the boundary of the top two levels of the HS tree. These biases diminish as the sample size grows sufficiently large, but for modest datasets can be substantial. Furthermore, these errors are exacerbated in higher dimensions since the likelihood of two neighboring labels falling on the opposite side of a high-level split increases due to the breadth-first tree construction.

To address this issue, we next develop a structured smoothing regularizer that spreads out the probability mass to nearby neighbors as a function of distance in the underlying ordinal space, rather than only their specific paths on the tree. The resulting LSHS model then uses the HS structure to capture long-range dependencies between labels and the leaf-smoothing regularizer to enforce local smoothness.

Trend filtering logits

Smoothing over discrete, graph-structured spaces where dependency between observations can be represented as edges between nodes is a well-studied problem in the signal processing literature. A common approach in such problems is to take a known graph structure with noisy observations at each node and leverage the inherent dependency between adjacent observations to reduce the overall estimation error of the true signal. Trend filtering (TF) (Kim et al. 2009; Tibshirani and others 2014) is one such recently-proposed technique for denoising that solves the following convex optimization problem:

$$\underset{\boldsymbol{\beta}\in\mathbb{R}^N}{\text{minimize}} \quad \ell(\mathbf{y}, \boldsymbol{\beta}) + \lambda \left|\left|\Delta^{(k)}\boldsymbol{\beta}\right|\right|_1 , \tag{2}$$

where $\mathbf{y}$ is a vector of noisy observations with some smooth convex loss function ℓ and a generalized lasso (Tibshirani and Taylor 2011) penalty term parameterized by the matrix $\Delta^{(k)} = D^{(k+1)}$. D is the oriented edge adjaceny matrix with each row corresponding to an edge (i, j) in the graph such that the i^{th} and j^{th} columns have value -1 and 1 respectively, and all other entries in that row have value 0. The Δ matrix thus encodes the $(k + 1)^{\text{th}}$-order differences and the L_1 penalty creates sparsity in these differences.

In the case of univariate Gaussian loss, solving the TF minimization problem results in a piecewise polynomial fit similar to a spline with adaptive knot placement. The order of the polynomial k and the weight λ are hyperparameters chosen to minimize some objective criterion such as AIC or BIC. Recent work (Wang et al. 2016) extends trend filtering to arbitrary graphs, and theoretical results show that trend filtering has strong minimax rates (Sadhanala, Wang, and Tibshirani 2016) and is optimally spatially adaptive for univariate discrete spaces (Guntuboyina et al. 2017).

We adapt TF to ordinal CPE by first noting that the label space defines a d-dimensional lattice graph, with each label in the ordinal space having an edge to its immediate neighbors. For example, if we have a two-dimensional label space then $y_{2,2}$ would be connected to $y_{1,2}$, $y_{2,1}$, $y_{3,2}$, and $y_{2,3}$. However, even given this natural dependency structure, TF is not immediately applicable to deep learning models. First, the optimization problems for most deep models are highly non-convex and cannot leverage specialized TF solvers. Furthermore, although the graph structure is given, it is over the set of conditional probabilities, rather than the coefficients of a linear model as in the generalized lasso setup.

Rather than smooth the coefficients in our model, we instead regularize the output log-probabilities. This encourages the model to produce smooth conditional distributions for every sample and yields the following loss function,

$$\mathcal{L}_i = -\log\left[p(y = y_i|x_i)\right] + \lambda \left|\left|\Delta^{(k)}\texttt{vec}(\log\left[p(y|x_i)\right])\right|\right|_1 . \tag{3}$$

We choose the hyperparameters k and λ via a validation set. As we show in Section 4, as dataset size increases, the best LSHS setting will drift towards smaller values of λ. Thus, in small-to-moderate sample size regimes, LSHS relies on trend filtering to smooth out the underlying space, and in large sample regimes it converges to the pure HS model.

Local smoothing via neighborhood filtering

A naive implementation of the trend filtering regularizer would require evaluating all the nodes in the discrete space. This would remove many of the computational performance advantages of the HS model described in Section 2. To ensure that HS scales to large spaces, we smooth only over a local neighborhood around the target value. Specifically, for a given y_i, we smooth over all nodes in the hypercube of radius r in output space, centered at y_i. The resulting regularization loss is then only over this subset of the space,

$$\mathcal{L}_i = -\log\left[p(y = y_i|x_i)\right] + \lambda \left|\left|\tilde{\Delta}^{(k)}\ell(y_i, x_i)\right|\right|_1 . \tag{4}$$

In (4), $\tilde{\Delta}^{(k)}$ is the graph trend filtering matrix for a discrete grid graph of size $(2r + 1)^d$ and $\ell(\cdot)$ is the neighborhood selection function that returns the vector of local conditional logits to smooth. Figure 1 illustrates this local filtering for a neighborhood radius of size 2 and a target label of 4.

By only needing to compute the values of a local neighborhood, the HS model regains its computational efficiency. For instance, in the case of a neighborhood radius of size 5 in a 3d scenario where each dimension is of size 64, the full smoothing model would have to calculate $\approx 262K$ output

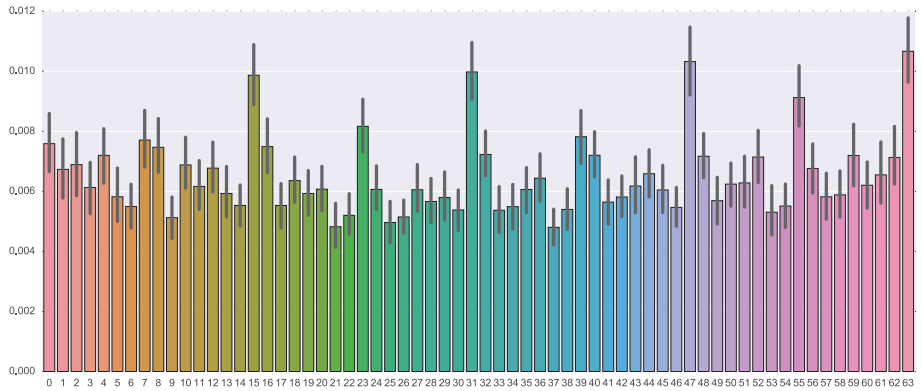


Figure 2: Average label error for a 64-label, 10-coefficient synthetic ordinal CPE problem, averaged over 300 trials with bars corresponding to standard error. For each trial, the true coefficients are drawn from a Gaussian process with squared exponential kernel and bandwidth 1 to simulate the smoothness of many real-world ordinal datasets. The HS model errors are generally higher at the labels that lay on the boundary of high-level splits in the HS tree.

HS nodes. The local smoother on the other hand only needs paths of size 24 for 1331 logits for an upper bound of $\approx 32K$ nodes. Even though this is already a sharp reduction ($\approx 88\%$), most of the local neighborhood will have highly-overlapping paths and thus the average number of nodes sampled is much lower than the upper bound.

4 Experiments

We evaluate LSHS on a series of benchmarks against real and synthetic data. First, we compare LSHS against approaches found in the recent literature and highlight the particular pathologies of each method. We then measure the performance of each method on real datasets of one, two, and three-dimensional discrete conditional target distributions, including a comparison of LSHS modifying a large-scale model, WaveNet (van den Oord et al. 2016a). Finally, in the appendix we show how hierarchical softmax is effected by the trend filtering with different neighborhood sizes.

Synthetic conditional distributions

We created a synthetic benchmark to evaluate the sample efficiency and systematic pathologies of both our method and other methods used in the recent literature. Our task is a variant on the well-known MNIST classification problem but with the twist that rather than mapping each digit to a latent class, each digit is mapped to a latent discrete distribution. For each sample image, we generate a label (y) by first mapping the digit to its corresponding distribution and then sampling y as a draw from that distribution, resulting in a training set of (X, y) values where X is an image (whose digit class is not explicitly known by the model) and y is an integer.

We compare six methods:

- **Multinomial (MN)**: A multinomial model with no knowledge of the structure of the output space.
- **Gaussian Mixture Model (GMM)**: An m-component GMM or Mixture Density Network (MDN) (Bishop 1994). For multi-dimensional data, we use a Cholesky parameterization of the covariance matrix.
- **Logistic Mixture Model (LMM)**: An m-component mixture of logistics, implemented using the CDF method of PixelCNN++ (Salimans et al. 2017).
- **Unsmoothed Hierarchical Softmax (HS)**: Our hierarchical softmax model with no smoothing.
- **Smoothed Multinomial (SMN)**: A multinomial model where structure of the space is smoothed by applying our trend filtering regularizer.
- **Leaf-Smoothed Hierarchicl Softmax (LSHS)**: Hiearchical softmax with a local smoothing window.

The first three methods appear in recent works in the literature. The HS and SMN models are ablation models that enable us to evaluate the effectiveness of LSHS components separately.

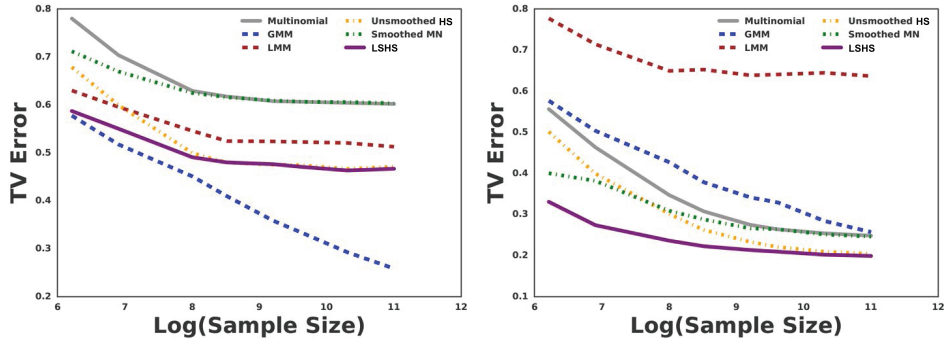
We consider two different ground truth distribution classes, both one dimensional. The first uses a 3-component GMM where component means and standard deviations are sampled uniformly from the range $[1, 7]$ and $[0.3, 2]$, respectively. The model is then discretized by evaluating the PDF at an evenly-spaced (zero-indexed) 128-bin grid along the range $[0.1, 10]$. The resulting distribution always has modes that fall far away from the boundaries at 0 and 127.

Real discrete data, however, often exhibits spikes near the boundaries. To address this case, we generated a second set of experiments where the ground truth is a mixture model of the form

$$p(x) = \frac{1}{3}\text{Exp}(x|\lambda_1) + \frac{1}{3}\text{Exp}(10.1 - x|\lambda_2) + \frac{1}{3}\mathcal{N}(x|\mu, \sigma)\,, \tag{5}$$

where Exp is the exponential distribution. We sample λ_1 and λ_2 uniformly randomly from the range $[0.25, 2]$ and sample μ and σ as in the GMM, then discretize this method following the same procedure used for the GMM. This creates an *edge-biased* distribution, with a smooth mode somewhere in the middle of the space and exponentially increasing mass at the boundaries, similar to the observed marginal subpixel intensity in the CIFAR dataset (Salimans et al. 2017).

For both distributions, we evaluate all six models on sample sizes of 500, 1K, 3K, 5K, 10K, 15K, 30K, and 60K. The



(a) GMM Distribution

(b) Edge-Biased Distribution

Figure 3: Performance of each method on latent (a) GMM, and (b) edge-biased distributions as sample size increases. The smooth GMM distribution is a 3-component mixture of Gaussians with no modes near the boundaries, resulting in an easy task for a GMM (mixture density network) model, though LSHS outperforms the other (misspecified) models, and is competitive in the small-sample regime. The edge-biased distribution has peaks at both boundaries, similar to observed pixel intensities in natural images, and LSHS performs well. In the low-sample regime, LSHS outperforms both of its constituent methods (Unsmoothed HS and Smoothed MN).

base network architecture for each model uses two 5×5 convolution layers of size 32 and 64 with 2×2 max pooling, followed by a dense hidden layer of size 1024; all layers use ReLU activations and dropout. Models are trained for 100K steps using Adam with learning rate 10^{-4}, $\epsilon = 1$, and batch size 50, reserving 20% of the train set for validation, with validation every 100 steps to save the best model and prevent overfitting. For GMM and LMM, we evaluated over $m \in \{1, 3, 5, 10, 20\}$. For smoothed models, we fixed the neighborhood radius to 5 and evaluated at $k \in \{1, 2\}$ and $\lambda \in \{1e{-}4, 5e{-}4, 1e{-}3, 5e{-}3, 1e{-}2, 5e{-}2, 0.1, 0.5, 1.0\}$. Hyperparameters were set by validation performance.

Figure 3 shows results in terms of total variation distance from the true distribution, averaged across ten independent trials. For the GMM distribution (Figure 3a), the GMM model is well-specified and consequently performs very well. In the low-sample GMM regime, the LSHS model is competitive with the GMM model, despite the fact that the GMM matches the parametric form of the ground truth. As previously noted, however, most data sets do not follow such an ideal form; for example, previous work (van den Oord et al. 2016a; 2016b; van den Oord, Kalchbrenner, and Kavukcuoglu 2016) has noted a multinomial model often outperforms a GMM model. If the GMM distribution were reflective of real data, we would not expect the multinomial model to outperform it.

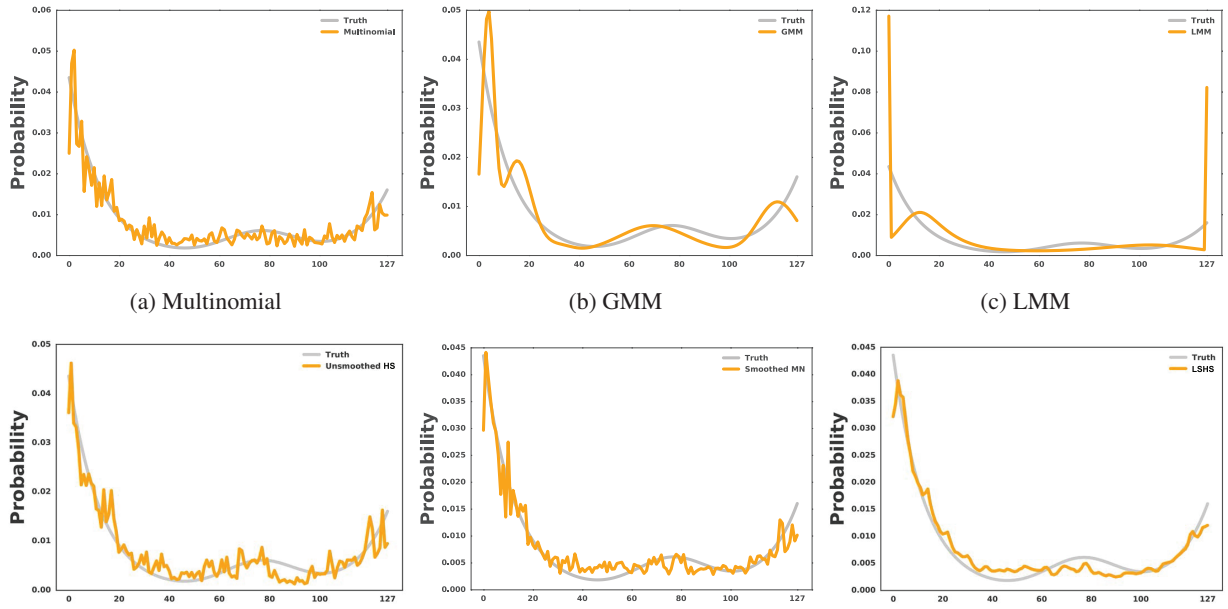
The edge-biased results in Figure 3b may be of more practical interest, as the design of this experiment is directly motivated by the real marginal subpixel intensity distributions seen in natural images. In the edge-biased scenario the multinomial model does in fact outperform the GMM model. However LSHS is clearly the best model here, with much stronger performance across all sample sizes. Interestingly, the LMM model performs very poorly, despite its design also being inspired by modeling pixel intensities. To better understand the performance of each of the models on the edge-biased dataset, we generated example conditional distributions when the model is trained with 3K samples.

Figure 4 shows plots of each model's estimate of the conditional distribution of the label for a single example image, with the ground truth shown in gray. The multinomial model (Figure 4a) treats every value as independent, resulting in a jagged reconstruction, especially in the tails, where the variance is particularly high. The GMM (Figure 4b) provides a smooth estimation which captures the middle mode well but drastically underestimates the tails because of the symmetric assumption of the model components. Conversely, the LMM (Figure 4c) produces large spikes at the two boundaries, due to the fact that the model takes boundaries to be the total component mass from $(-\infty, 0]$ and $[127, \infty)$. This is an intentional bias in the model designed to better match CIFAR pixel intensities which also have spikes at the boundaries. This is quite a strong bias, effectively resulting in a two-point-inflated smooth model with a nontrivial bias towards the boundaries. Finally, the HS and SMN models (Figures 4d and 4e) result in slightly better fits than the simple multinomial model, but combining both into the LSHS model (Figure 4f) results in a smooth fit that is able to estimate the tails well.

In both distributions, we observe that the hierarchical softmax and local smoothing are jointly beneficial. Both HS and SMN outperform a simple multinomial, and combining them both in the LSHS model is superior to both. As the sample size grows, LSHS converges to the HS model in performance, as increased data results in a decreased need for smoothing. Indeed, as we show in the appendix, the average chosen λ (smoothing) value decreases as the sample size grows.

Real-world datasets

We compile a benchmark of real-world datasets with discrete conditional distributions as a target output (or where the target variable is discrete). We use seven datasets from the UCI database; three are one-dimensional targets, three are two-dimensional, and one is three-dimensional. Every model uses a network architecture of three hidden layers of sizes 256, 128, and 64 with ReLU activation, weight decay, and dropout.



(a) Multinomial (b) GMM (c) LMM

(d) Unsmoothed HS (e) Smoothed MN (f) LSHS

Figure 4: Example fits for benchmark methods with 3000 training samples on the edge-biased distribution. (a) The multinomial model is extremely noisy, as it ignores label structure. (b) The GMM never puts substantial mass outside the feasible range, resulting in misplaced modes near the tails. (c) The LMM over-estimates the boundaries due to the CDF formulation of the log-likelihood. (d,e) The unsmoothed HS and smoothed MN models both improve on the pure multinomial model, but are still noisy. (f) The LSHS model finds a smooth fit which does not grossly misestimate the tails.

Models were trained with Adam with decaying learning rate with initial rate 10^{-1}, minimum rate 10^{-4}, and decay rate of 0.25, decaying the rate after the current model has failed to improve for 10 epochs. Training stops after 1000 epochs or if the current learning rate is below the minimum learning rate. All results are averages using 10-fold cross-validation and we use 20% of the training data in each trial as a validation set. For all datasets, we select hyperparameter settings as in Section 4. We plot the marginal distributions of each real dataset in the appendix.

We also evaluate on a pixel prediction task for both MNIST and CIFAR-10, where we sample a 10×10 patch of the image and must predict the pixel located at $(11, 11)$, relative to the origin of the patch. For both image datasets, we consider 3 different training sample sizes (500, 5K, and 50K). Every model uses a network architecture of two 3×3 convolution layers of size 32 and 64, with 2×2 max pooling, followed by three dense hidden layers of size 1024, 128, and 32; all layers use ReLU activation, weight decay, and dropout. Other training details are identical to the UCI dataset, with the exception that we only perform a single trial on the CIFAR datasets due to computational constraints. Similarly, we reduce the resolution of the CIFAR dataset from 256^3 to 64^3. Plots of the marginal distributions of all our datasets are available in the supplementary material.

Table 1 presents the results on all the candidate datasets, with the best-performing score in bold for each dataset and metric. We measure performance both in log-probability of observed points and root mean squared error (RMSE), as the discrete space has a natural measurement of distance. In general, LSHS performs very well in cases where the size of the discrete space dominates the sample size. In datasets where this is not the case (Abalone and Parkinsons), the multinomial model has sufficient data to model the space well. LMM outperforms LSHS in terms of log-probs on the Housing dataset, likely due to the large peaks at the boundaries in the data (see plot in the supplement for details). The CIFAR dataset also has substantial peaks (especially at corners), resulting in the LMM outperforming the multinomial model, as demonstrated previously in (Salimans et al. 2017). However, the additional dataset structure is better modeled via the LSHS model, which has nearly half the RMSE of other methods.

WaveNet-LSHS

As a final validation, we modify the WaveNet generative audio model (van den Oord et al. 2016a) to use LSHS.* The WaveNet model performs a compression of the waveform to a pre-specified number of bins. The original experiments used 256 bins with a multinomial distribution, which was found empirically by (van den Oord et al. 2016a) to outperform GMMs and is thus the model we compare against. All trials were run on the VCTK corpus (Yamagishi 2012), with varying compression sizes, for 6K epochs; for LSHS we used a radius of 5 and $\lambda = 0.01$ for all experiments; all other parameters were set to defaults. Table 2 shows the results for

*We use the WaveNet implementation at https://github.com/ibab/tensorflow-wavenet.

			Multinomial		GMM		LMM		LSHS	
Model	Grid Size	Samples	log-probs	RMSE	log-probs	RMSE	log-probs	RMSE	log-probs	RMSE
Abalone	29	4177	**-822.83**	**2.17**	-907.78	2.42	-857.23	2.30	-851.88	2.31
Auto-MPG	377	392	-177.81	37.34	-187.02	31.59	-186.67	32.49	**-160.31**	**30.84**
Housing	451	506	-297.59	69.20	-247.23	40.81	**-240.20**	39.53	-246.44	**36.18**
MNIST-500	256	500	**-1416.69**	**80.90**	-2588.79	92.78	-1658.78	81.57	-1466.65	86.68
MNIST-5K	256	5000	-1229.77	64.10	-2096.24	94.16	-1231.01	64.29	**-1224.42**	**63.06**
MNIST-50K	256	50000	-1173.80	58.82	-2365.57	94.24	-1191.11	60.41	**-1161.69**	**57.16**
Students	21×20	395	-209.07	5.27	-219.67	5.44	-209.43	5.26	**-200.76**	**5.18**
Energy	38×38	768	-323.10	6.21	-492.49	10.89	-437.90	14.24	**-279.01**	**4.17**
Parkinsons	36×49	5875	**-1941.91**	**6.42**	-3969.63	10.91	-3633.29	13.53	-3530.22	14.93
Concrete	$30 \times 59 \times 43$	103	-115.88	21.34	-107.46	18.91	-108.63	21.07	**-102.34**	**18.09**
CIFAR-500	$64 \times 64 \times 64$	500	-9980.57	26.35	-9177.59	**24.69**	-9109.57	26.00	**-8519.21**	25.81
CIFAR-5K	$64 \times 64 \times 64$	5000	-9688.08	26.26	-9106.04	23.01	-9213.49	26.11	**-7504.35**	**14.89**
CIFAR-50K	$64 \times 64 \times 64$	50000	-8409.60	22.66	-9099.49	23.02	-9214.51	26.08	**-6796.39**	**13.42**

Table 1: Results for the four models on a series of discrete datasets from the UCI database and the MNIST and CIFAR-10 datasets. The best scores for each metric and dataset are bolded; grid size corresponds to the number of bins in the underlying discrete space. Overall, the LSHS model performs very strongly, especially where the grid size is much larger than sample size.

Model	Original (Multinomial)	LSHS
WaveNet-256	2.5780	**2.4820**
WaveNet-512	**3.2630**	3.2970
WaveNet-1024	3.9440	**3.8730**
WaveNet-2048	4.7100	**4.6240**

Table 2: Results for WaveNet on the VCTK corpus (in bits-per-sample) as sound fidelity is increased. LSHS outperforms the original model at most resolutions.

each model, given in bits-per-sample on the test set. Overall, LSHS outperforms the original WaveNet model in three out of four settings and remains close on the fourth setting.

5 Discussion

As our experiments demonstrate, the LSHS model outperforms several alternative models commonly used in the deep learning literature. In the one-dimensional case, other models have been proposed, ranging from more flexible parametric component distributions for mixture models (Carreau and Bengio 2009) to quantile regression (Taylor 2000; Lee and Yang 2006). Extending these models to higher dimensions is non-trivial, making them unsuitable for use in many of our target applications. Furthermore, even in the 1d case, it is often unclear *a priori* which parametric components should be included in a mixture model, and simply adding a large number may result in overfitting. A quantile regression model would also suffer from the same overfitting issues as the unsmoothed HS model in the appendix neighborhood experiments, as it does not explicitly impose smoothness. Quantile regression would also require all nodes to be calculated at every iteration and would therefore not scale well to large (i.e. finely-discretized) 1d spaces.

There have also been other multidimensional models, notably the line of work in neural autoregressive models such as NADE (Uria et al. 2016), RNADE (Uria, Murray, and Larochelle 2013), and MADE (Germain et al. 2015); and variational autoencoders (Kingma and Welling 2013) such as DRAW (Gregor et al. 2015). We see such models as complementary approaches rather than competitive approaches to LSHS. For instance, one could modify the outputs of MADE to be a separate discrete distribution for each dimension rather than a single likelihood. This would also address the main scalability issue of our model: currently LSHS requires $\mathcal{O}(n)$ output nodes for a space of n possible values. In the low-dimensional problems explored in this paper this was not a problem, but it quickly exceeds the memory of a GPU once one moves beyond three or four dimensions.

6 Conclusion

We have presented LSHS, a model for conditional probability estimation over ordinal labels. By dividing a label space into a series of half-spaces, LSHS transforms the distribution estimation task into a hierarchical classification task which overcomes many disadvantages of simple multinomial modeling. The leaf nodes in the tree are then smoothed at train time using graph-based trend filtering on the resulting logit probabilities in a local region around the target label. Hierarchical decomposition and leaf smoothing were shown to have an additive effect on total variation error reduction in synthetic datasets. The benchmark results on both real and synthetic datasets suggest that LSHS is a powerful method that sometimes substantially improves the performance of ordinal prediction models, especially when sample efficiency is a concern.

7 Acknowledgments

This work was generously supported by NSF CAREER grant DMS-1255187.

References

Bishop, C. M. 1994. Mixture density networks.

Carreau, J., and Bengio, Y. 2009. A hybrid Pareto mixture for conditional asymmetric fat-tailed distributions. *Neural Networks, IEEE Transactions on* 20(7):1087–1101.

Dahl, R.; Norouzi, M.; and Shlens, J. 2017. Pixel recursive super resolution. *arXiv preprint arXiv:1702.00783*.

Deshpande, A.; Lu, J.; Yeh, M.-C.; and Forsyth, D. 2016. Learning diverse image colorization. *arXiv preprint arXiv:1612.01958*.

Germain, M.; Gregor, K.; Murray, I.; and Larochelle, H. 2015. MADE: Masked autoencoder for distribution estimation. In *ICML*, 881–889.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Gray, A. G., and Moore, A. W. 2003. Nonparametric density estimation: Toward computational tractability. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, 203–211. SIAM.

Greenspan, H.; van Ginneken, B.; and Summers, R. M. 2016. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging* 35(5):1153–1159.

Gregor, K.; Danihelka, I.; Graves, A.; Rezende, D. J.; and Wierstra, D. 2015. DRAW: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*.

Gulrajani, I.; Kumar, K.; Ahmed, F.; Taiga, A. A.; Visin, F.; Vazquez, D.; and Courville, A. 2016. PixelVAE: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013*.

Guntuboyina, A.; Lieu, D.; Chatterjee, S.; and Sen, B. 2017. Spatial adaptation in trend filtering. *arXiv preprint arXiv:1702.05113*.

Jean, S.; Cho, K.; Memisevic, R.; and Bengio, Y. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

Kim, S.-J.; Koh, K.; Boyd, S.; and Gorinevsky, D. 2009. ℓ_1 trend filtering. *SIAM review* 51(2):339–360.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Lee, T.-H., and Yang, Y. 2006. Bagging binary and quantile predictors for time series. *Journal of econometrics* 135(1):465–497.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Morin, F., and Bengio, Y. 2005. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, 246–252. Citeseer.

Ng, N.; Gabriel, R. A.; McAuley, J.; Elkan, C.; and Lipton, Z. C. 2017. Predicting surgery duration with neural heteroscedastic regression. *arXiv preprint arXiv:1702.05386*.

Ranganath, R.; Perotte, A.; Elhadad, N.; and Blei, D. 2016. Deep survival analysis. *arXiv preprint arXiv:1608.02158*.

Ravanbakhsh, S.; Lanusse, F.; Mandelbaum, R.; Schneider, J.; and Poczos, B. 2016. Enabling dark energy science with deep generative models of galaxy images. *arXiv preprint arXiv:1609.05796*.

Sadhanala, V.; Wang, Y.-X.; and Tibshirani, R. J. 2016. Total variation classes beyond 1d: Minimax rates, and the limitations of linear smoothers. In *Advances in Neural Information Processing Systems*, 3513–3521.

Salimans, T.; Karpathy, A.; Chen, X.; and Kingma, D. P. 2017. PixelCNN++: Improving the PixelCNN with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*.

Taylor, J. W. 2000. A quantile regression neural network approach to estimating the conditional density of multiperiod returns. *Journal of Forecasting* 19(4):299–311.

Tibshirani, R. J., et al. 2014. Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics* 42(1):285–323.

Tibshirani, R. J., and Taylor, J. 2011. The solution path of the generalized lasso. *Annals of Statistics* 39:1335–71.

Uria, B.; Côté, M.-A.; Gregor, K.; Murray, I.; and Larochelle, H. 2016. Neural autoregressive distribution estimation. *Journal of Machine Learning Research* 17(205):1–37.

Uria, B.; Murray, I.; and Larochelle, H. 2013. Rnade: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, 2175–2183.

van den Oord, A.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016a. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

van den Oord, A.; Kalchbrenner, N.; Espeholt, L.; Vinyals, O.; Graves, A.; et al. 2016b. Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems*, 4790–4798.

van den Oord, A.; Kalchbrenner, N.; Vinyals, O.; Espeholt, L.; Graves, A.; and Kavukcuoglu, K. 2016c. Conditional image generation with PixelCNN decoders. *arXiv preprint arXiv:1606.05328*.

van den Oord, A.; Kalchbrenner, N.; and Kavukcuoglu, K. 2016. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*.

Wang, Y.-X.; Sharpnack, J.; Smola, A.; and Tibshirani, R. J. 2016. Trend filtering on graphs. *Journal of Machine Learning Research* 17(105):1–41.

Yamagishi, J. 2012. English multi-speaker corpus for cstr voice cloning toolkit.