Online Density Estimation of Nonstationary Sources Using Exponential Family of Distributions

Kaan Gokcesu and Suleyman S. Kozat, *Senior Member, IEEE*

***Abstract*— We investigate online probability density estimation (or learning) of nonstationary (and memoryless) sources using exponential family of distributions. To this end, we introduce a truly sequential algorithm that achieves Hannan-consistent log-loss regret performance against true probability distribution without requiring any information about the observation sequence (e.g., the time horizon T and the drift of the underlying distribution C) to optimize its parameters. Our results are guaranteed to hold in an individual sequence manner. Our log-loss performance with respect to the true probability density has regret bounds of $O((CT)^{1/2})$, where C is the total change (drift) in the natural parameters of the underlying distribution. To achieve this, we design a variety of probability density estimators with exponentially quantized learning rates and merge them with a mixture-of-experts notion. Hence, we achieve this square-root regret with computational complexity only logarithmic in the time horizon. Thus, our algorithm can be efficiently used in big data applications. Apart from the regret bounds, through synthetic and real-life experiments, we demonstrate substantial performance gains with respect to the state-of-the-art probability density estimation algorithms in the literature.**

***Index Terms*— Big data, exponential family, mixture-of-experts, nonstationary source, online density estimation, online learning.**

I. Introduction

Real-life engineering applications are often probabilistic in nature, since most practical systems are subject to random components via input, interference, or noise [1]. In this brief, we investigate probability density estimation (or learning) of these random components, which arise in several different machine learning applications such as big data [2], pattern recognition [3], novelty detection [4], data mining [5], anomaly detection [6], and feature selection [7]. In particular, we investigate online probability density estimation [8], where we sequentially observe the sample vectors $\{x_1, x_2, \ldots\} \in \mathbb{R}^{d_x}$ and learn a probability distribution at each time t based on the past observations $\{x_1, x_2, \ldots, x_{t-1}\}$. We assume that the observations are generated from a possibly nonstationary memoryless (piecewise independent identically distributed) source (discrete or continuous), since, in most engineering applications, statistics of a data stream may change over time (especially in big data) [9].

We approach this problem from a competitive algorithm perspective where the competing strategy is the true probability density function. At each time t, we observe a sample feature vector x_t distributed according to some unknown density function $f_t(x_t)$, and based on our past observations $\{x_\tau\}_1^{t-1}$, we produce an estimate of this density as $\hat{f}_t(x_t)$. As the loss, we use the log-loss function, i.e., $-\log(\hat{f}_t(x_t))$, since it is the most obvious and widely used loss function for probability distributions [10]. To provide strong results in an individual sequence manner [11], we use the notion of "regret" to define our performance, such that the regret at time t is

$$r_t = -\log(\hat{f}_t(x_t)) + \log(f_t(x_t)) \tag{1}$$

and the cumulative regret up to time T is

$$R_T = \sum_{t=1}^{T} (-\log(\hat{f}_t(x_t)) + \log(f_t(x_t))). \tag{2}$$

The instantaneous regret definition in (1) can either be positive or negative in a specific round just like in any other expert competition settings [12]. However, the cumulative regret in (2) is bound to be positive, since the competition (i.e., true distribution) minimizes the cumulative log-loss.

We seek to achieve the performance of the best nonstationary distribution from an exponential family. In this sense, we assume that there exists a density function $f_t(x_t)$ that exactly or most closely represents the true distribution and $f_t(x_t)$ belongs to an exponential family [13] with a possibly changing natural parameter vector $\alpha_t \in \mathbb{R}^d$ (cumulatively representing the mean, sufficient statistics, and normalization.) at each time t. We specifically investigate the exponential family of distributions, since exponential families cover a wide range of parametric statistical models [6] and accurately approximates many nonparametric classes of probability densities [14].

We denote the total drift of α_t in T rounds by C_α, such that

$$C_\alpha \triangleq \sum_{t=2}^{T} \|\alpha_t - \alpha_{t-1}\| \tag{3}$$

where $\|\cdot\|$ is the l^2-norm. As an example, for stationary sources, i.e., distributions with unchanging natural parameter, the drift C_α is 0. Following [6] and [15], one can show that a regret bound of order $O(\log(T))$ can be achieved for a stationary source with fixed computational complexity. However, for nonstationary sources, the logarithmic regret bound is infeasible under low computational complexity [6]. The results of [16] imply fixed complexity learning algorithms that achieve a regret bound of $O((C_\alpha T)^{1/2})$ when the time horizon T and the total drift in parameter vector C_α are known *a priori* to optimize their parameters. For unknown time horizon, one can utilize the doubling trick [11] for the algorithm given in [16], since a simple modification of the algorithm given in [16] also implies a regret bound of order $O((C_{\max} T)^{1/2})$ if an upper bound on the total drift is known *a priori*, such that $C_{\max} \geq C_\alpha$. However, if no prior knowledge about C_α is given, an algorithm that achieves only the regret bound $O(C_\alpha T^{1/2})$ is proposed in [6]. Hence, achieving $O((C_\alpha T)^{1/2})$ is not possible with the state-of-the-art methods if no prior information is given about C_α to optimize their parameters.

To this end, our contributions are as follows.

1) As the first time in literature, we introduce an algorithm that achieves an $O((C_\alpha T)^{1/2})$ regret bound without requiring any knowledge about the source (e.g., C_α, T) to optimize its parameters.
2) Our results are guaranteed to hold in a strong deterministic sense for all possible observation sequences.

Manuscript received July 5, 2016; revised February 2, 2017 and July 10, 2017; accepted August 1, 2017. This work was supported in part by the Turkish Academy of Sciences Outstanding Researcher Programme and in part by the Scientific and Technological Research Council of Turkey under Contract 113E517. *(Corresponding author: Kaan Gokcesu.)*

The authors are with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara 06800, Turkey (e-mail: gokcesu@ee.bilkent.edu.tr; kozat@ee.bilkent.edu.tr).

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2740003

3) Our algorithm is truly sequential, such that neither T nor the total drift C_α is required. We achieve this performance with a computational complexity only log-linear in the data length by designing density estimators with exponentially quantized learning rates and merging them with a mixture-of-experts notion. Thus, our algorithm is suitable for applications involving big data.

Through synthetic and real-life experiments, we demonstrate significant performance gains with respect to the state-of-the-art methods in the literature.

In Section II, we first introduce the basic density estimators, which will be subsequently used to build our universal algorithm. Then, in Section III, we introduce the universal density estimator that merges the beliefs of the basic density estimators. In Section IV, we illustrate significant performance gains over both real and synthetic data, and finish with concluding remarks in Section V.

II. Basic Density Estimator

In this section, we first construct basic density estimators that can only achieve the minimum regret bound with *a priori* information on the underlying sequence (e.g., C_α, T) to optimize its learning rate. These estimators are subsequently used in Section III to construct the final algorithm that achieves the minimum regret bound without requiring any information to optimize its parameters. Here, at each time t, we observe $x_t \in \mathbb{R}^{d_x}$ distributed according to a memoryless (i.e., independent of the past samples) exponential-family distribution

$$f_t(x_t) = \exp(-\langle \alpha_t, z_t \rangle - A(\alpha_t)) \tag{4}$$

where $\alpha_t \in \mathbb{R}^d$ is the natural parameter of the distribution belonging to a bounded convex feasible set S, such that

$$D = \max_{\alpha \in S} \|\alpha\|. \tag{5}$$

$A(\cdot)$ is the normalization or log-partition function, that is

$$A(\alpha) = \log\left(\int_{\mathcal{X}} \exp(-\langle \alpha, \mathcal{T}(x) \rangle) dx\right) \tag{6}$$

and z_t is the d-dimensional sufficient statistic of x_t [13], that is

$$z_t = \mathcal{T}(x_t). \tag{7}$$

Instead of directly estimating the density $f_t(x)$, we estimate the natural parameter α_t at each time t according to our observations $\{x_\tau\}_{\tau=1}^{t-1}$. The estimated density is given by

$$\hat{f}_t(x_t) = \exp(-\langle \hat{\alpha}_t, z_t \rangle - A(\hat{\alpha}_t)). \tag{8}$$

We use online gradient descent [16] to sequentially produce our estimation $\hat{\alpha}_t$, where we first start from an initial estimate $\hat{\alpha}_1$, and update our recent estimation $\hat{\alpha}_t$ based on our new observation x_t. To update $\hat{\alpha}_t$, we first observe a sample x_t and incur the loss $l(\hat{\alpha}_t, x_t)$ according to our estimation $\hat{\alpha}_t$, which is $-\log(\hat{f}_t(x_t))$ (log-loss). From (8), the loss is

$$l(\hat{\alpha}_t, x_t) = \langle \hat{\alpha}_t, z_t \rangle + A(\hat{\alpha}_t). \tag{9}$$

Then, we calculate the gradient of the loss with respect to $\hat{\alpha}_t$

$$\begin{aligned} \nabla_\alpha l(\hat{\alpha}_t, x_t) &= z_t + \nabla_\alpha A(\hat{\alpha}_t) \\ &= z_t + \frac{\int_{\mathcal{X}} -\mathcal{T}(x) \exp(-\langle \hat{\alpha}_t, \mathcal{T}(x) \rangle) dx}{\int_{\mathcal{X}} \exp(-\langle \hat{\alpha}_t, \mathcal{T}(x) \rangle) dx} = z_t - \mu_{\hat{\alpha}_t} \end{aligned} \tag{10}$$

Algorithm 1 Basic Density Estimator

1: Initialize learning rate $\eta \in \mathbb{R}^+$
2: Select initial parameter $\hat{\alpha}_1$
3: Calculate the mean $\mu_{\hat{\alpha}_1}$
4: **for** $t = 1$ **to** T **do**
5: Declare estimation $\hat{\alpha}_t$
6: Observe x_t
7: Calculate $z_t = \mathcal{T}(x_t)$
8: Update parameter: $\tilde{\alpha}_{t+1} = \hat{\alpha}_t - \eta(z_t - \mu_{\hat{\alpha}_t})$
9: Project onto convex set: $\hat{\alpha}_{t+1} = P_S(\tilde{\alpha}_{t+1})$
10: Calculate the mean $\mu_{\hat{\alpha}_{t+1}}$
11: **end for**

where we used the definition in (6) in the second equality and $\mu_{\hat{\alpha}_t}$ is the mean of $\mathcal{T}(x_t)$ (i.e., z_t) if x_t were distributed according to $\hat{f}_t(x_t)$ as in (8). We update $\hat{\alpha}_t$, such that

$$\hat{\alpha}_{t+1} = P_S(\hat{\alpha}_t - \eta(z_t - \mu_{\hat{\alpha}_t})) \tag{11}$$

where $P_S(\cdot)$ is the projection onto the set S and is defined as

$$P_S(x) = \arg\min_{y \in S} \|x - y\|. \tag{12}$$

The complete algorithm is provided in Algorithm 1. Next, we provide performance bounds of Algorithm 1. Theorem 1 shows that Algorithm 1 can achieve the minimum regret bound $O((C_\alpha T)^{1/2})$ if C_α is known *a priori* to optimize η.

Theorem 1: When Algorithm 1 is used with parameter η to estimate the distribution $f_t(x_t)$, its regret is upper bounded by

$$R_T \leq \frac{1}{\eta} DC + \eta T G$$

where D is defined as in (5), $C = 2.5D + C_\alpha$, such that C_α is defined as in (3), and $G = (\phi_2 + 2\phi_1 M + M^2)/2$, such that $M = \max_{\alpha \in S} \mu_\alpha$, $\phi_1 = \sum_{t=1}^T \|z_t\|/T$, and $\phi_2 = \sum_{t=1}^T \|z_t\|^2/T$.

Proof of Theorem 1: The regret at time t is defined as

$$r_t = l(\hat{\alpha}_t, x_t) - l(\alpha_t, x_t) \tag{13}$$

where $l(\alpha, x)$ is as in (9). Since the loss function is convex

$$r_t \leq \langle \nabla_\alpha l(\hat{\alpha}_t, x_t), (\hat{\alpha}_t - \alpha_t) \rangle. \tag{14}$$

We bound the right-hand side of (14) using the update rule (11). By definition of projection in (12), we have

$$\|P_S(\hat{\alpha}_t - \eta \nabla_\alpha l(\hat{\alpha}_t, x_t)) - \alpha_t\| \leq \|\hat{\alpha}_t - \eta \nabla_\alpha l(\hat{\alpha}_t, x_t) - \alpha_t\|. \tag{15}$$

Substituting (11) in the left-hand side provides

$$\|\hat{\alpha}_{t+1} - \alpha_t\| \leq \|\hat{\alpha}_t - \eta \nabla_\alpha l(\hat{\alpha}_t, x_t) - \alpha_t\|. \tag{16}$$

Hence, we get

$$\begin{aligned} \|\hat{\alpha}_{t+1} - \alpha_t\|^2 \leq \|\hat{\alpha}_t - \alpha_t\|^2 &- 2\eta \langle \nabla_\alpha l(\hat{\alpha}_t, x_t), (\hat{\alpha}_t - \alpha_t) \rangle \\ &+ \eta^2 \|\nabla_\alpha l(\hat{\alpha}_t, x_t)\|^2. \end{aligned} \tag{17}$$

Combining (14) and (17) results in

$$\begin{aligned} r_t &\leq \frac{1}{2\eta}(\|\hat{\alpha}_t - \alpha_t\|^2 - \|\hat{\alpha}_{t+1} - \alpha_t\|^2) + \frac{\eta}{2}\|\nabla_\alpha l(\hat{\alpha}_t, x_t)\|^2 \\ &\leq \frac{1}{2\eta}(\|\hat{\alpha}_t\|^2 - \|\hat{\alpha}_{t+1}\|^2 - 2\langle \hat{\alpha}_t - \hat{\alpha}_{t+1}, \alpha_t \rangle) + \frac{\eta}{2}\|\nabla_\alpha l(\hat{\alpha}_t, x_t)\|^2 \end{aligned} \tag{18}$$

since $\eta > 0$. Using (10) in the right-hand side yields

$$r_t \leq \frac{1}{2\eta}(\|\hat{\alpha}_t\|^2 - \|\hat{\alpha}_{t+1}\|^2) - \frac{1}{\eta}\langle \hat{\alpha}_t - \hat{\alpha}_{t+1}, \alpha_t \rangle + \frac{\eta}{2}\|z_t - \mu_{\hat{\alpha}_t}\|^2. \tag{19}$$

Thus, summing (19) from $t = 1$ to T, we have the cumulative regret up to time T, which is given by

$$R_T \leq \frac{1}{2\eta}(\|\hat{\alpha}_1\|^2 - \|\hat{\alpha}_{T+1}\|^2) + \frac{\eta}{2}\sum_{t=1}^{T}\|z_t - \mu_{\hat{\alpha}_t}\|^2 - \frac{1}{\eta}\left(\langle\hat{\alpha}_1, \alpha_1\rangle + \sum_{t=2}^{T}\langle\hat{\alpha}_t, \alpha_t - \alpha_{t-1}\rangle - \langle\hat{\alpha}_{T+1}, \alpha_T\rangle\right). \quad (20)$$

Using (3) and (5), we get

$$R_T \leq \frac{1}{\eta}(2.5D^2 + DC_\alpha) + \frac{\eta}{2}\sum_{t=1}^{T}(\|z_t + \mu_{\hat{\alpha}_t}\|^2) \leq \frac{1}{\eta}(2.5D^2 + DC_\alpha) + \frac{\eta T}{2}(\phi_2 + 2\phi_1 M + M^2) \quad (21)$$

where M, ϕ_1, and ϕ_2 are given by

$$M = \max_{\alpha\in S}\mu_\alpha, \quad \phi_1 = \sum_{t=1}^{T}\frac{\|z_t\|}{T}, \quad \phi_2 = \sum_{t=1}^{T}\frac{\|z_t\|^2}{T}.$$

We denote $G = (\phi_2 + 2\phi_1 M + M^2)/2$, which is related to the gradient of the log-loss and $C = C_\alpha + 2.5D$, which is the effective change parameter. Hence

$$R_T \leq \frac{1}{\eta}DC + \eta TG \quad (22)$$

which concludes the proof of the theorem. □

The result in Theorem 1 is for an estimator that uses the fixed learning rate, which will be used to prove the performance bound of the universal estimator in Section III.

Remark 1: The construction of z_t requires the knowledge of sufficient statistics mapping $\mathcal{T}(\cdot)$ beforehand. Since the sufficient statistics of different kinds of exponential family distributions may differ, $\mathcal{T}(\cdot)$ requires the knowledge of the exact distribution kind, e.g., whether the distribution is normal, exponential, gamma, and so on. This requirement can be easily bypassed by creating an extended statistics vector $\tilde{z}_t = \tilde{\mathcal{T}}(x_t)$, such that $\tilde{z}_t$ encompasses all sufficient statistics of different distributions that the true density may belong to. This would also solve the problem of estimating a distribution that changes types, e.g., from Gaussian to gamma, gamma to Bernoulli, and so on. However, extending the sufficient statistics $\tilde{z}_t$ effectively increases S, and hence D, C, G in Theorem 1 as well.

Remark 2: Suppose instead of x_t, we observe a distorted version, such that $y_t = Q(x_t)$, where $Q(\cdot)$ is the distortion channel, e.g., an additive noise channel. Then, using an unbiased estimator $\bar{z}_t = \bar{\mathcal{T}}(y_t)$ such that $\mathbf{E}[\bar{z}_t] = \mathcal{T}(x_t)$ produces the same results for the expected regret.

Remark 3: In general, an exponential-family distribution has the form $f(x) = \exp(-\langle\alpha, \mathcal{T}(x)\rangle - A(\alpha) - B(x))$, where $B(x)$ is only a function of the observation x. However, this function can simply be included inside of $\mathcal{T}(x)$, whose corresponding parameter in the inner product will simply be 1 in the true probability density. Hence, all the analyses still hold.

III. Universal Online Density Estimation

In Section II, we constructed the basic estimators that can only achieve the minimum regret bound with *a priori* information. In this section, we construct a universal density estimator (UDE) that achieves the minimum regret with no *a priori* information by mixing the beliefs of the basic density estimators with exponentially quantized learning rates.

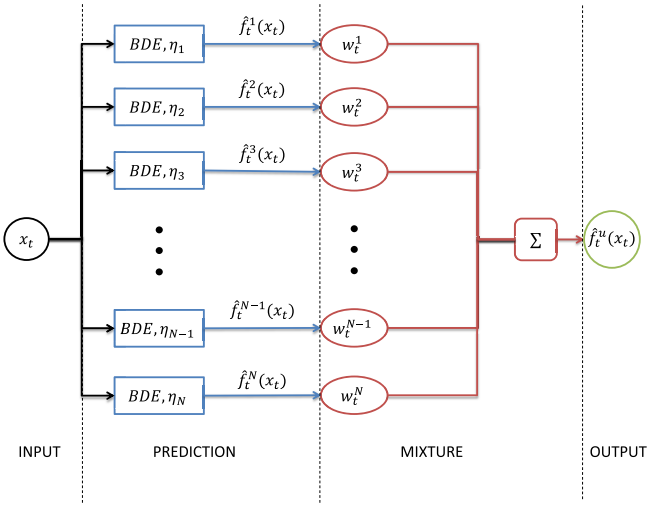

Fig. 1. Illustration of a universal density estimator.

When used with parameter η, Algorithm 1 achieves the regret

$$R_T \leq \sqrt{DCGT}\left(\frac{\eta_*}{\eta} + \frac{\eta}{\eta_*}\right) \quad (23)$$

where $\eta_* \triangleq ((DC)/(GT))^{1/2}$, which is the bound in Theorem 1. To achieve the minimum regret with Algorithm 1, one must optimize η with some knowledge of η_*. However, with limited or no prior information, it is not possible to achieve the minimum regret using Algorithm 1. Therefore, instead of just using Algorithm 1 with a fixed learning rate, we combine different runs of Algorithm 1 with different learning rates, which will contain (or approximate) η_* to a sufficient degree to achieve the minimum regret.

To this end, we first construct a parameter vector $\boldsymbol{\eta}$ of size N, such that $\boldsymbol{\eta}[r] = \eta_r$, for $r \in \{1, 2, \ldots, N\}$. We construct N experts, each of which runs Algorithm 1 with parameter η_r, i.e., r^{th} element of the parameter vector $\boldsymbol{\eta}$. As shown in Fig. 1, each one of the N experts takes the input x_t and outputs a belief $\hat{f}_t^r(x_t)$ at each round t (prediction stage). Then, we mix all of the beliefs in a weighted combination such that

$$\hat{f}_t^u(x_t) = \sum_{r=1}^{N} w_t^r \hat{f}_t^r(x_t) \quad (24)$$

where w_t^r is the combination weight of the belief of the r^{th} expert at time t (mixture stage). Initially, we assign uniform weights to all expert outputs, such that their combination weights are given by $w_1^r = 1/N$. Then, at each time t, we update their weights according to the rule

$$w_{t+1}^r = w_t^r \hat{f}_t^r(x_t)/\hat{f}_t^u(x_t) \quad (25)$$

where $\hat{f}_t^u(x_t)$ acts as the normalizer. Instead of our weighting, different methods [12], [17]–[20] can also be used. Our combination in (24) and (25) makes use of the mixability property of density functions under log-loss (it is 1-mixable) [12], where $\log N$ redundancy is optimal [20]–[22]. We have provided a complete description of the universal algorithm in Algorithm 2.

Next, we provide the performance bounds of the universal density estimator, i.e., Algorithm 2. The results of Theorem 2 and Corollary 1 show that the optimal regret bound $O((CT)^{1/2})$ is achieved without any prior information on C.

Algorithm 2 Universal Density Estimator

1: Initialize constants η_r, for $r \in \{1, 2, \ldots, N\}$
2: Create N copies Alg. 1, where the r^{th} algorithm runs with the parameter η_r and its belief is given by $\hat{f}_t^r(x)$.
3: Initialize weights $w_1^r = 1/N$
4: **for** $t = 1$ **to** T **do**
5: Receive the beliefs $\hat{f}_t^r(x)$ for $r \in \{1, 2, \ldots, N\}$
6: Declare estimation $\hat{f}_t^u(x) = \sum_{r=1}^N w_t^r \hat{f}_t^r(x)$
7: Observe x_t
8: Calculate $z_t = \mathcal{T}(x_t)$
9: **for** $r = 1$ **to** N
10: Update parameters $\hat{\alpha}_t^r$ of r^{th} algorithm according to Alg. 1
11: $w_{t+1}^r = w_t^r \hat{f}_t^r(x_t)/\hat{f}_t^u(x_t)$
12: **end for**
13: **end for**

Theorem 2: Algorithm 2 has the regret bound

$$R_T \leq \log(N) + \sqrt{DCGT}\left[\min_{i\in\{1,2,\ldots,N\}}\left(\frac{\eta_*}{\eta_i} + \frac{\eta_i}{\eta_*}\right)\right]$$

where D is defined as in (5), $C = 2.5D + C_\alpha$ such that C_α is defined as in (3), $G = (\phi_2 + 2\phi_1 M + M^2)/2$ such that $M = \max_{\alpha\in S}\mu_\alpha$, $\phi_1 = \sum_{t=1}^T \|z_t\|/T$, $\phi_2 = \sum_{t=1}^T \|z_t\|^2/T$, $\eta_* = ((DC)/(GT))^{1/2}$, and η_i is the parameter of the ith expert.

Proof of Theorem 2: The regret at time t is given by

$$r_t = -\log\left(\hat{f}_t^u(x_t)\right) + \log(f_t(x_t)). \tag{26}$$

Summing (26) from $t = 1$ to T gives

$$R_T = -\log\left(\prod_{t=1}^T \hat{f}_t^u(x_t)\right) + \sum_{t=1}^T \log(f_t(x_t)). \tag{27}$$

Using (24), we have

$$R_T = -\log\left(\prod_{t=1}^T\left(\sum_{r=1}^N w_t^r \hat{f}_t^r(x_t)\right)\right) + \sum_{t=1}^T \log(f_t(x_t)). \tag{28}$$

From (25), we can infer that the weights are given by

$$w_t^r = \frac{\prod_{\tau=1}^{t-1}\hat{f}_\tau^r(x_\tau)}{\sum_{r=1}^N\prod_{\tau=1}^{t-1}\hat{f}_\tau^r(x_\tau)}. \tag{29}$$

Hence, substituting (29) in (28) produces

$$\begin{aligned} R_T &= -\log\left(\prod_{t=1}^T\left(\frac{\sum_{r=1}^N\prod_{\tau=1}^t\hat{f}_t^r(x_t)}{\sum_{r=1}^N\prod_{\tau=1}^{t-1}\hat{f}_t^r(x_t)}\right)\right) + \sum_{t=1}^T\log(f_t(x_t)) \\ &= -\log\left(\sum_{r=1}^N\prod_{\tau=1}^T\hat{f}_t^r(x_t)\right) + \log(N) + \sum_{t=1}^T\log(f_t(x_t)) \\ &\leq \log(N) - \max_r\left(\sum_{t=1}^T\log\left(\hat{f}_t^r(x_t)\right)\right) + \sum_{t=1}^T\log(f_t(x_t)) \qquad (30) \\ &\leq \log(N) + \sqrt{DCGT}\left[\min_{i\in\{1,2,\ldots,N\}}\left(\frac{\eta_*}{\eta_i} + \frac{\eta_i}{\eta_*}\right)\right] \qquad (31) \end{aligned}$$

and concludes the proof. □

The result of Theorem 2 shows that the performance bound is dependent on the set of learning rates used in the algorithm. In Corollary 1, we show that we can achieve the minimum regret bound with log-linear complexity.

Corollary 1: Suppose we run the experts with parameters between η' and η''. We denote $K = \eta''/\eta'$ and $N = \lceil\log_2 K\rceil + 1$. Then, running Algorithm 2 with parameter vector $\eta_i = 2^{i-1}\eta'$ for $i \in \{1, 2, \ldots, N\}$ gives the following regret bounds:

1) *If* $\eta' \leq \eta_* \leq \eta''$

$$R_T \leq \log(\lceil\log_2\eta''/\eta'\rceil + 1) + \frac{3\sqrt{2}}{2}\sqrt{DCGT}$$

since $((\eta_*/\eta_i) + (\eta_i/\eta_*))$ is maximum if $\eta_* = 2^{(a+1/2)}$ for some a.

2) *If* $\eta_* \geq \eta''$

$$R_T \leq \log(\lceil\log_2\eta''/\eta'\rceil + 1) + \left(1 + \frac{\eta_*}{\eta''}\right)\sqrt{DCGT}$$

Since $\eta_* \leq ((4 + 1/T)D^2M^{-2})^{1/2}$, by letting $\eta'' \geq ((4 + 1/T)D^2M^{-2})^{1/2}$, we can make this case invalid.

3) *If* $\eta_* \leq \eta'$

$$R_T \leq \log(\lceil\log_2\eta''/\eta'\rceil + 1) + \left(1 + \frac{\eta'}{\eta_*}\right)\sqrt{DCGT}.$$

Since $\eta_* \geq (2.5D^2/(TG))^{1/2}$, setting $\eta' \leq (2.5D^2/T)^{1/2}$ gives

$$R_T \leq \log(\lceil\log_2\eta''/\eta'\rceil + 1) + (1 + \sqrt{G})\sqrt{DCGT}.$$

Note that we may not be able to make $\eta' \leq (2.5D^2/T)^{1/2}$ to make this case invalid, since we may not be able to bound G. However, we may be able to bound G with high probability, which will in turn create the regret bound in 1 with high probability.

Remark 4: If $\eta_* = ((DC)/(GT))^{1/2}$ is known completely beforehand, then running Algorithm 2 with the parameter vector $\boldsymbol{\eta} = \{\eta_*\}$, i.e., $N = 1$, produces the regret bound

$$R_T \leq 2\sqrt{DCGT}$$

which is equivalent to achieving the optimal regret bound using Algorithm 1 with the *a priori* information about the source.

Hence, by running Algorithm 2 with an appropriate parameter vector, we achieve $O((CT)^{1/2})$ regret with $O(\log T)$ computational complexity, since the separation between η' and η'' is mainly dependent on C, which is bounded as $2.5D \leq C \leq (2T + 0.5)D$.

Remark 5: Note that we have only included probability density estimators of Algorithm 1 as experts for Algorithm 2. However, the result in (30) is general and is true for any expert used in the mixture. Therefore, incorporating various different density estimators (parametric or nonparametric) in Algorithm 2, we can achieve the optimum performance in the mixture.

IV. Experiments

In this section, we demonstrate the performance of our algorithm both on real and synthesized data in comparison with the-state-of-art techniques maximum likelihood (ML) [23], online convex programming with static (OCP.static) [16] and dynamic (OCP.dynamic) learning rates [6], gradient descent (GD) [24], Momentum [25], Nesterov accelerated gradient (NAG) [26], Adagrad [27], Adadelta [28], and Adam [29].

All the algorithms are implemented as instructed in their respective papers. OCP.static uses the doubling trick [11] to run in an online manner and is implemented with the fixed learning rate $T^{-1/2}$ for an epoch of length T. OCP.dynamic, on the other hand, uses a dynamic learning rate of $t^{-1/2}$ at time t. For an online behavior, ML uses a sliding window to determine its estimations. In our implementation, ML uses the doubling trick and is run in epochs of durations $\{1, 2, 4, 8, \ldots\}$. Before the start of each epoch at time t, we run ML for the past $t - 1$ observations with window lengths of $\{1, 2, 4, 8, \ldots, t - 1\}$. Then, in its current epoch, we use the

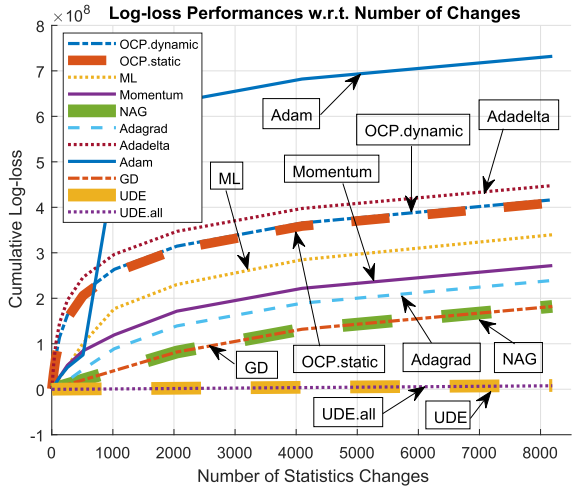

Fig. 2. Cumulative log-loss regret performances of the density estimation algorithms with respect to the number of changes in the statistics of a nonstationary Gaussian process.

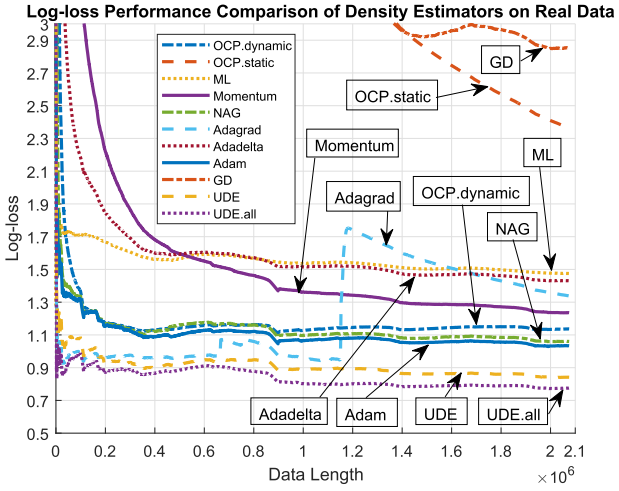

Fig. 3. Average log-loss performance of the density estimators over the Individual Household Electric Power Consumption Data Set [30].

window length that provides the minimum log-loss. Hence, ML has a time complexity of $O(\log T)$ per round. The other algorithms are also run with the doubling trick and used a similar approach to search their parameter spaces (like the window size for ML), since implementing them with their default parameters provided poor performance. We have run GD, Momentum, NAG, Adagrad, and Adam in the past observations for the step sizes $\eta = \{(1/T), (2/T), (4/T) \ldots, (1/4), (1/2), 1, 2, 4, \ldots, (T/4), (T/2)T\}$, and selected the step size that provided the minimum log-loss and use it in their next epoch. We have optimized only the step size and left the momentum term in Momentum, NAG to be its default value, i.e., $\gamma = 0.9$, since the step size is the main parameter that affects the performance and optimization of γ also would have increased the time complexity to $O(\log^2 T)$, which would have been unfair. We have set the smoothing term of Adagrad to its default value $\epsilon = 10^{-8}$. In Adam, we have set the parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ as instructed in its paper. For Adadelta, the parameter that most influences the performance is the exponential update parameter γ. Therefore, we optimize γ among the set $\{0, (1/T), (2/T), (4/T) \ldots, (1/8), (1/4), (1/2), (3/4), (7/8), \ldots, 1-(4/T), 1-(2/T), 1-(1/T), 1\}$ and set the smoothing parameter to its default value $\epsilon = 10^{-8}$.

We have run our algorithm, UDE, with learning rates in the range $1/T \leq \eta \leq T$ for a T length epoch of the algorithm. We have also created a variant UDE.all that combines not only the subroutines of UDE but also all the competing algorithms to demonstrate the option of using various density estimators in combination. The experiments[1]." consists of two parts, which are performance comparison in synthetic and real data sets.

A. Synthetic Data Set

In this section, we compare the cumulative log-loss regrets of the algorithms with respect to the number of changes in the statistics of the source. To this end, we compare the algorithms' performances when the source has $C \in \{1, 2, 4, 8, \ldots\}$ changes in its statistics. For each value of C, we synthesize a data set of size 10000 from a univariate Gaussian process with a unit standard deviation, i.e., $\sigma = 1$ and mean value alternating between 100 and -100 in every T/C samples (equal length time segments), such that in the first segment $\mu = 100$, in the next segment $\mu = -100$, and alternates as such. No prior information about the data is given to the algorithms (including the switching times) except the variance of the distribution. All of algorithms start from an initial mean estimate of 0.

In Fig. 2, we have illustrated the regret performance of the algorithms. We observe in Fig. 2 that the Adam algorithm performs substantially worse in high number of changes in the statistics. Since Adam gets rid of the step size for a completely adaptive behavior, its convergence rate is simply not good enough in this setting. OCP.static, OCP.dynamic, and Adadelta have similar performances but still perform worse than ML. Even though momentum and Adagrad have better performance than ML, they are still outperformed by GD and NAG. Nonetheless, our algorithm, UDE, outperforms all the other algorithms by huge margins, because it does not try to optimize its parameters, but rather combines them to ensure that the optimal one will survive. UDE.all performs basically the same as UDE, since UDE has substantially greater performance.

B. Real Data Set

We use "Individual Household Electric Power Consumption Data Set"[2]." for real big data benchmark purposes, which is readily accessible online [30]. This data set includes measurements of electric power consumption in one household with a 1-min sampling rate over a period of almost 4 years [30]. We have assumed a possibly nonstationary multivariate Gaussian process for this data set and run the algorithms to estimate its distribution. Since the true distribution is not known, we have compared the performances of the algorithms directly with their log-losses instead of their regrets. All the algorithms are initialized to zero-mean, unit variance.

In Fig. 3, we have illustrated the log-loss performances of all the algorithms. Interestingly, GD performed the worst with OCP.static a close second, even though it was one of the best performing competing algorithms in the previous experiments. ML provided an average performance similar to the first set of experiments. However, this time performed worse than the OCP.dynamic and the Adadelta algorithms. NAG performed similarly by being one of the best performing competing algorithms. Even though Adagrad was able

[1]The codes used in the experiments are made publicly available at http://www.ee.bilkent.edu.tr/~gokcesu/density_codes.zip

[2]This data set is the most popular ($\approx$ 125 000 hits) large data set (greater than 10^5 samples) in the University of California, Irvine, Machine Learning Repository, which is publicly available at https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption

to outperform all the other competing algorithms up to the middle of the data set, it encountered a sudden increase in its log-loss and was unable to recuperate fast enough. The most interesting result was for the Adam algorithm, which performed the best among the competing algorithms, even though it performed the worst in the first set of experiments. From the distinct performances of the competing algorithms in real and synthetic data sets, we can infer that our algorithm UDE is able to outperform them in various different environments. In Fig. 3, we again observe a substantial performance gain in comparison with the other algorithms. In addition, we are also able to observe a small performance increase in UDE.all on top of UDE, which supports the notion that combining different estimators would lead to better performance because of the low regret redundancy of the mixture.

V. Conclusion

We have introduced a truly sequential and online algorithm, which estimates the density of a nonstationary memoryless exponential-family source with Hannan consistency. Our algorithms are truly sequential, such that neither the time horizon T nor the total drift of the natural parameter is required to optimize its parameters. Here, the regret of our algorithm is increasing with only the square-root of time horizon T and the total drift of the natural parameter C. The results we provide are uniformly guaranteed to hold in a strong deterministic sense in an individual sequence manner for all possible observation sequences, since we refrain from making any assumptions on the observations. We achieve this performance with a computational complexity only log linear in the data length by carefully designing different probability density estimators and combining them in a mixture-of-experts setting. Due to such efficient performance and storage need, our algorithm can be effectively used in big data applications.

References

[1] H. Wang, "Minimum entropy control of non-Gaussian dynamic stochastic systems," *IEEE Trans. Autom. Control*, vol. 47, no. 2, pp. 398–403, Feb. 2002.

[2] Y. Nakamura and O. Hasegawa, "Nonparametric density estimation based on self-organizing incremental neural network for large noisy data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 1, pp. 8–17, Jan. 2017.

[3] A. Penalver and F. Escolano, "Entropy-based incremental variational Bayes learning of Gaussian mixtures," *IEEE Trans. Neural Netw.*, vol. 23, no. 3, pp. 534–540, Mar. 2012.

[4] X. Ding, Y. Li, A. Belatreche, and L. P. Maguire, "Novelty detection using level set methods," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 3, pp. 576–588, Mar. 2015.

[5] Y. Cao, H. He, and H. Man, "SOMKE: Kernel density estimation over data streams by sequences of self-organizing maps," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1254–1268, Aug. 2012.

[6] M. Raginsky, R. M. Willett, C. Horn, J. Silva, and R. F. Marcia, "Sequential anomaly detection in the presence of noise and limited feedback," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5544–5562, Aug. 2012.

[7] P. Padungweang, C. Lursinsap, and K. Sunat, "A discrimination analysis for unsupervised feature selection via optic diffraction principle," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 10, pp. 1587–1600, Oct. 2012.

[8] R. J. Carroll, "On sequential density estimation," *Zeitschrift Wahrscheinlichkeitstheorie Verwandte Gebiete*, vol. 36, no. 2, pp. 137–151, 1976.

[9] K. B. Dyer, R. Capo, and R. Polikar, "Compose: A semisupervised learning framework for initially labeled nonstationary streaming data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 12–26, Jan. 2014.

[10] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[11] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth, "How to use expert advice," *J. ACM*, vol. 44, no. 3, pp. 427–485, May 1997.

[12] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[13] B. O. Koopman, "On distributions admitting a sufficient statistic," *Trans. Amer. Math. Soc.*, vol. 39, no. 3, pp. 399–409, 1936.

[14] A. R. Barron and C.-H. Sheu, "Approximation of density functions by sequences of exponential families," *Ann. Statist.*, vol. 19, no. 3, pp. 1347–1369, 1991.

[15] E. Hazan, A. Agarwal, and S. Kale, "Logarithmic regret algorithms for online convex optimization," *Mach. Learn.*, vol. 69, nos. 2–3, pp. 169–192, 2007.

[16] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proc. ICML*, 2003, pp. 928–936.

[17] V. Vovk, "Aggregating strategies," in *Proc. 3rd Annu. Workshop Comput. Learn. Theory*, 1990, pp. 371–383.

[18] N. Cesa-Bianchi, Y. Freund, D. P. Helmbold, D. Haussler, R. E. Schapire, and M. K. Warmuth, "How to use expert advice," in *Proc. Annu. ACM Symp. Theory Comput.*, 1993, pp. 382–391.

[19] A. C. Singer and M. Feder, "Universal linear prediction by model order weighting," *IEEE Trans. Signal Process.*, vol. 47, no. 10, pp. 2685–2699, Oct. 1999.

[20] V. Vovk and C. Watkins, "Universal portfolio selection," in *Proc. 11th Annu. Conf. Comput. Learn. Theory*, 1998, pp. 12–23.

[21] Y. M. Shtar'kov, "Universal sequential coding of single messages," *Probl. Peredachi Inf.*, vol. 23, no. 3, pp. 3–17, 1987.

[22] A. Orlitsky, N. P. Santhanam, and J. Zhang, "Universal compression of memoryless sources over unknown alphabets," *IEEE Trans. Inf. Theory*, vol. 50, no. 7, pp. 1469–1481, Jul. 2004.

[23] I. J. Myung, "Tutorial on maximum likelihood estimation," *J. Math. Psychol.*, vol. 47, no. 1, pp. 90–100, Feb. 2003.

[24] L. Bottou, "Online learning and stochastic approximations," *On-line Learn. Neural Netw.*, vol. 17, no. 9, p. 142, 1998.

[25] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Netw.*, vol. 12, no. 1, pp. 145–151, 1999.

[26] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Math. Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

[27] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Feb. 2011.

[28] M. D. Zeiler. (2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701

[29] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[30] G. Hebrail and A. Berard. (2012). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml