# Sequential Density Estimation via Nonlinear Continuous Weighted Finite Automata

**Tianyu Li** TIANYU.LI@MAIL.MCGILL.CA
*McGill University, MILA*

**Bogdan Mazoure** BOGDAN.MAZOURE@MAIL.MCGILL.CA
*McGill University, MILA*

**Guillaume Rabusseau** GUILLAUME.RABUSSEAU@UMONTREAL.CA
*Université de Montréal, DIRO, MILA, CIFAR AI Chair*

## Abstract

Weighted finite automata (WFAs) have been widely applied in many fields. One of the classic problems for WFAs is probability distribution estimation over sequences of discrete symbols. Although WFAs have been extended to deal with continuous input data, namely continuous WFAs (CWFAs) (Li et al., 2022), it is still unclear how to approximate density functions over sequences of continuous random variables using WFA-based models, due to the limitation on the expressiveness of the model as well as the tractability of approximating density functions via CWFAs. In this paper, we propose a nonlinear extension to the CWFA model to first improve its expressiveness, we refer to it as the nonlinear continuous WFAs (NCWFAs). Then we leverage the so-called RNADE method, which is a well-known density estimator based on neural networks, and propose the RNADE-NCWFA model. The RNADE-NCWFA model computes a density function by design. We show that this model is strictly more expressive than the Gaussian HMM model, which CWFA cannot approximate. Empirically, we conduct a synthetic experiment using Gaussian HMM generated data. We focus on evaluating the model's ability to estimate densities for sequences of varying lengths (longer length than the training data). We observe that our model performs the best among the compared baseline methods.

**Keywords:** Weighted finite automata, sequential density estimation, neural density estimation.

## 1. Introduction

Many tasks in natural language processing, computational biology, reinforcement learning, and time series analysis rely on learning with sequential data, i.e. , estimating functions defined over sequences of observations from training data. Weighted finite automata (WFAs) are a powerful and flexible class of models which can efficiently represent such functions. WFAs are tractable, they encompass a wide range of machine learning models (they can for example compute any probability distribution defined by a hidden Markov model (HMM) (Denis and Esposito, 2008) and can model the transition and observation behavior of partially observable Markov decision processes (Thon and Jaeger, 2015)) and they offer appealing theoretical guarantees. In particular, the so-called *spectral methods* for learning HMMs (Hsu et al., 2009), WFAs (Bailly et al., 2009; Balle et al., 2014a) and related models (Glaude and Pietquin, 2016; Boots et al., 2011), provide an alternative to Expectation-Maximization (EM) based learning algorithms that is both computationally efficient and consistent.

One of the major applications of WFA is to approximate probability distribution over sequences of discrete symbols. Although the WFA model has been extended to the continuous domain (Li et al., 2022; Rabusseau et al., 2019) as the so-called linear 2-RNN model (or continuous WFA model), approximating density functions for sequential data under continuous domain using this model is not straight-forward, as the model does not guarantee to compute a density function by construction. Moreover, due to the linearity of the model, the continuous WFA model (CWFA) is not expressive enough to estimate some of the common density functions over sequences of continuous random variables such as a Gaussian hidden Markov model.

In recent years, neural networks have been widely applied in density estimation and have been proved to be particularly successful. To estimate a density function via neural networks, the neural density estimator need to be flexible enough to represent complex densities but have tractable inference functions and learning algorithms. One particular example of such models is the class of autoregressive models (Uria et al., 2016, 2013), where the joint density is decomposed into a product of conditionals and each conditional is approximated by a neural network. One other type of methods are the so-called flow-based methods (normalizing flows) (Dinh et al., 2014, 2016; Rezende and Mohamed, 2015). Flow-based methods transform a base density (e.g. a standard Gaussian) into the target density by an invertible transformation with tractable Jacobian. Although these methods have been used to estimate sequential densities, the sequences often come as fixed length. It is often unclear how to generalize these methods to account for varying length of the sequences in the testing phase, which can be important for some sequential task, such as language modeling for NLP task. Weighted finite automata, on the other hand, are designed to carry out such task under the discrete setting. The question is, how to generalize WFA to approximate density functions over continuous domains.

In this paper, by extending the classic CWFA model with a (nonlinear) feature mapping and a (nonlinear) termination function, we first propose our nonlinear continuous weighted finite automata (NCWFA) model. Combining this model with the RNADE framework (Uria et al., 2013), we propose RNADE-NCWFA to approximate sequential density functions. The model is flexible as it naturally generalizes to sequences of varying lengths. Moreover, we show that the RNADE-NCWFA model is strictly more expressive than the Gaussian HMM model. In addition, we propose a spectral learning based algorithm for efficiently learning the parameters of a RNADE-NCWFA. For the empirical study, we conduct synthetic experiments using data generated from a Gaussian HMM model. We compare our proposed spectral learning of RNADE-NCWFA with HMM learned with the EM algorithm, RNADE with LSTM and RNADE-NCWFA learned with stochastic gradient descent. We evaluate the models' performance through their log likelihood over sequences of unseen length, meaning the testing sequences are longer than the training sequences, to observe the models' generalization ability. We show that our model outperforms all the baseline models on this metric, especially for long testing sequences. Moreover, the advantage of our model is more significant when dealing with small training sizes and noisy data.

## 2. Background

In this section, we first introduce basic tensor algebra. Then we introduce the continuous weighted finite automata model as well as the RNADE model for density estimation.

### 2.1. Tensor algebra

We first recall basic definitions of tensor algebra; more details can be found in (Kolda and Bader, 2009). A tensor $\mathcal{T} \in \mathbb{R}^{d_1 \times \cdots \times d_p}$ can simply be seen as a multidimensional array $(\mathcal{T}_{i_1,\cdots,i_p} : i_n \in [d_n], n \in [p])$. The mode-$n$ fibers of $\mathcal{T}$ are the vectors obtained by fixing all indices except the $n$th one, e.g. $\mathcal{T}_{:,i_2,\cdots,i_p} \in \mathbb{R}^{d_1}$. The $n$th mode matricization of $\mathcal{T}$ is the matrix having the mode-$n$ fibers of $\mathcal{T}$ for columns and is denoted by $\mathcal{T}_{(n)} \in \mathbb{R}^{d_n \times d_1 \cdots d_{n-1} d_{n+1} \cdots d_p}$. The vectorization of a tensor is defined by $\mathrm{vec}(\mathcal{T}) = \mathrm{vec}(\mathcal{T}_{(1)})$. In the following $\mathcal{T}$ always denotes a tensor of size $d_1 \times \cdots \times d_p$.

The mode-$n$ matrix product of the tensor $\mathcal{T}$ and a matrix $\mathbf{X} \in \mathbb{R}^{m \times d_n}$ is a tensor denoted by $\mathcal{T} \times_n \mathbf{X}$. It is of size $d_1 \times \cdots \times d_{n-1} \times m \times d_{n+1} \times \cdots \times d_p$ and is defined by the relation $\mathcal{Y} = \mathcal{T} \times_n \mathbf{X} \Leftrightarrow \mathcal{Y}_{(n)} = \mathbf{X}\mathcal{T}_{(n)}$. The mode-$n$ vector product of the tensor $\mathcal{T}$ and a vector $\boldsymbol{v} \in \mathbb{R}^{d_n}$ is a tensor defined by $\mathcal{T} \bullet_n \boldsymbol{v} = \mathcal{T} \times_n \boldsymbol{v}^\top \in \mathbb{R}^{d_1 \times \cdots \times d_{n-1} \times d_{n+1} \times \cdots \times d_p}$. It is easy to check that the $n$-mode product satisfies $(\mathcal{T} \times_n \mathbf{A}) \times_n \mathbf{B} = \mathcal{T} \times_n \mathbf{B}\mathbf{A}$ where we assume compatible dimensions of the tensor $\mathcal{T}$ and the matrices $\mathbf{A}$ and $\mathbf{B}$. Given strictly positive integers $n_1, \cdots, n_k$ satisfying $\sum_i n_i = p$, we use the notation $(\mathcal{T})_{\langle\langle n_1, n_2, \cdots, n_k \rangle\rangle}$ to denote the $k$th order tensor obtained by reshaping $\mathcal{T} \in \mathbb{R}^{d_1 \times \cdots \times d_p}$ into a tensor* of size $(\prod_{i_1=1}^{n_1} d_{i_1}) \times (\prod_{i_2=1}^{n_2} d_{n_1+i_2}) \times \cdots \times (\prod_{i_k=1}^{n_k} d_{n_1+\cdots+n_{k-1}+i_k})$.

A rank $R$ tensor train (TT) decomposition (Oseledets, 2011) of a tensor $\mathcal{T} \in \mathbb{R}^{d_1 \times \cdots \times d_p}$ factorizes $\mathcal{T}$ into the product of $p$ core tensors $\mathcal{G}_1 \in \mathbb{R}^{d_1 \times R}, \mathcal{G}_2 \in \mathbb{R}^{R \times d_2 \times R}, \cdots, \mathcal{G}_{p-1} \in \mathbb{R}^{R \times d_{p-1} \times R}, \mathcal{G}_p \in \mathbb{R}^{R \times d_p}$, and is defined† by $\mathcal{T}_{i_1,\cdots,i_p} = (\mathcal{G}_1)_{i_1,:}(\mathcal{G}_2)_{:,i_2,:} \cdots (\mathcal{G}_{p-1})_{:,i_{p-1},:}(\mathcal{G}_p)_{:,i_p}$ for all indices $i_1 \in [d_1], \cdots, i_p \in [d_p]$ (here $(\mathcal{G}_1)_{i_1,:}$ is a row vector, where $[d] = \{1, 2, \cdots, d\}$, $(\mathcal{G}_2)_{:,i_2,:}$ is an $R \times R$ matrix, etc.). We will use the notation $\mathcal{T} = [\![\mathcal{G}_1, \cdots, \mathcal{G}_p]\!]$ to denote this product. The name of this decomposition comes from the fact that the tensor $\mathcal{T}$ is decomposed into a train of lower-order tensors.

### 2.2. Continuous weighted finite automata (CWFAs)

The concept of continuous weighted finite automata (CWFAs) is a generalization of the classic weighted finite automata model to its continuous input case and is also shown to be equivalent to the linear second-order RNN model (Li et al., 2022; Rabusseau et al., 2019).

**Definition 1** *A continuous weighted finite automaton with $k$ states (CWFA) is defined by a tuple $A = \langle \boldsymbol{\alpha}, \mathcal{A}, \boldsymbol{\Omega} \rangle$, where $\boldsymbol{\alpha} \in \mathbb{R}^k$ is the initial weight, $\mathcal{A} \in \mathbb{R}^{k \times d \times k}$ is the transition tensor, and $\boldsymbol{\Omega} \in \mathbb{R}^{k \times p}$ is the termination matrix. Let $(\mathbb{R}^d)^*$ denote the set of sequences of size $d$ real-valued vectors. A CWFA computes the following function $f : (\mathbb{R}^d)^* \to \mathbb{R}^p$:*

$$f(\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n) = (\mathcal{A} \bullet_1 \boldsymbol{\alpha} \bullet_2 \boldsymbol{x}_1)^\top (\mathcal{A} \bullet_2 \boldsymbol{x}_2) \cdots (\mathcal{A} \bullet_2 \boldsymbol{x}_n) \boldsymbol{\Omega}. \tag{1}$$

*Note that the specific ordering used to perform matricization, vectorization and such a reshaping is not relevant as long as it is consistent across all operations.

†The classical definition of the TT-decomposition allows the rank $R$ to be different for each mode, but this definition is sufficient for the purpose of this paper.

To learn the CWFA model, (Li et al., 2022) extend the spectral learning algorithm for the classic WFA model (Mohri et al., 2012; Balle et al., 2014b) to its continuous case. The algorithm relies on the Hankel tensor, which is a generalization of the Hankel matrix.

**Definition 2** *For a function $f : (\mathbb{R}^d)^* \to \mathbb{R}^p$, its Hankel tensor of length $l$, $\mathcal{H}_f^l \in \mathbb{R}^{[d]^l \times p}$ is defined by $(\mathcal{H}_f)^l_{i_1,\cdots,i_l,:} = f(\boldsymbol{e}_{i_1}, \cdots, \boldsymbol{e}_{i_l})$, where $\boldsymbol{e}_1, \cdots, \boldsymbol{e}_d$ denotes the canonical basis of $\mathbb{R}^d$.*

In practice, to learn the Hankel tensor, one can use gradient descent to minimize the loss function $\mathcal{L}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l, \boldsymbol{y}, \mathcal{H}) = \|(\boldsymbol{x}^{\otimes})^\top (\mathcal{H})_{\langle\langle l,1 \rangle\rangle} - \boldsymbol{y}\|_F^2$, over a training dataset. Here $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l$ is an input sequence and $\boldsymbol{y}$ the corresponding output, and $\boldsymbol{x}^{\otimes} = \boldsymbol{x}_1 \otimes \boldsymbol{x}_2 \otimes \cdots \otimes \boldsymbol{x}_l$. It is is shown in (Li et al., 2022) that the Hankel tensor of a CWFA with finite states can be parameterized by its tensor train form, i.e. $\mathcal{H}^{(l)} = [\![\mathcal{G}_1, \cdots, \mathcal{G}_{l+1}]\!]$. The spectral learning algorithm for CWFA relies on the following theorem (Li et al., 2022) showing how to recover the parameters of CWFA from Hankel tensors of the function it computes.

**Theorem 3** *Let $f : (\mathbb{R}^d)^* \to \mathbb{R}^p$ be a function computed by a minimal linear CWFA with $n$ hidden units and let $L$ be an integer such that*[‡] $\text{rank}((\mathcal{H}_f^{(2L)})_{\langle\langle L, L+1 \rangle\rangle}) = n$. *Then, for any $\mathbf{P} \in \mathbb{R}^{d^L \times n}$ and $\mathbf{S} \in \mathbb{R}^{n \times d^L p}$ such that $(\mathcal{H}_f^{(2L)})_{\langle\langle L, L+1 \rangle\rangle} = \mathbf{PS}$, the minimal CWFA computing $f$ is defined by:*

$$\boldsymbol{\alpha} = (\mathbf{S}^\dagger)^\top (\mathcal{H}_f^{(L)})_{\langle\langle L+1 \rangle\rangle}, \ \boldsymbol{\Omega}^\top = \mathbf{P}^\dagger (\mathcal{H}_f^{(L)})_{\langle\langle L,1 \rangle\rangle}, \ \mathcal{A} = ((\mathcal{H}_f^{(2L+1)})_{\langle\langle L,1,L+1 \rangle\rangle}) \times_1 \mathbf{P}^\dagger \times_3 (\mathbf{S}^\dagger)^\top$$

### 2.3. Real-valued neural autoregressive density estimator (RNADE)

The real-valued neural autoregressive density estimator (RNADE) (Uria et al., 2013) is a generalization of the original neural autoregressive density estimator (NADE) (Uria et al., 2016) to continuous variables. The core idea of RNADE is to estimate the joint density using the chain rule and approximate each conditional density via neural networks, i.e.

$$p(x_1, \cdots, x_n) = \prod_{i=1}^n p(x_i | x_{<i}) \quad \text{with} \quad p(x_i | x_{<i}) = p_M(x_i | \theta_i), \tag{2}$$

where $x_{<i}$ denotes all attributes preceding $x_i \in \mathbb{R}$ in a fixed ordering, $p_M$ is a mixture of $m$ Gaussians with parameters $\theta_i = \{\boldsymbol{\beta}_i \in \mathbb{R}^m, \boldsymbol{\mu}_i \in \mathbb{R}^m, \boldsymbol{\sigma}_i \in \mathbb{R}^m\}$. Moreover, we have: $p_M(x_i|\theta_i) = \sum_{j=1}^m \boldsymbol{\beta}_i^j \mathcal{N}(x_i | \boldsymbol{\mu}_i^j, \boldsymbol{\sigma}_i^j)$, where $\boldsymbol{\beta}_i^j$ denotes the $j$th element of $\boldsymbol{\beta}_i$, same for $\boldsymbol{\mu}_i^j$ and $\boldsymbol{\sigma}_i^j$ and $\mathcal{N}(x_i | \boldsymbol{\mu}_i^j, \boldsymbol{\sigma}_i^j)$ denotes the Gaussian density with mean $\boldsymbol{\mu}_i^j$ and standard deviation $\boldsymbol{\sigma}_i^j$ evaluated at $x_i$. Note that $\boldsymbol{\beta}_i, \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i$ are functions of $x_{<i}$. These functions are often chosen to be various forms of neural networks. In the classic setting, RNADE with $m$ mixing components and $k$ hidden states has the following update rules:

$$\boldsymbol{h}_i = g_i(\boldsymbol{h}_{i-1}), \quad \boldsymbol{\beta}_i = \text{softmax}(\mathbf{V}_i^\beta \boldsymbol{h}_{i-1} + \boldsymbol{b}_i^\beta) \tag{3}$$

$$\boldsymbol{\mu}_i = \mathbf{V}_i^\mu \boldsymbol{h}_{i-1} + \boldsymbol{b}_i^\mu, \quad \boldsymbol{\sigma}_i = \exp(\mathbf{V}_i^\sigma \boldsymbol{h}_{i-1} + \boldsymbol{b}_i^\sigma), \tag{4}$$

where $\mathbf{V}_i^\beta, \mathbf{V}_i^\mu$ and $\mathbf{V}_i^\sigma$ are $m \times k$ matrices, $\boldsymbol{b}_i^\beta, \boldsymbol{b}_i^\mu$ and $\boldsymbol{b}_i^\sigma$ are vectors of size $m$, and $g_i$ is an update function for the hidden state which is time step dependent (see (Uria et al., 2013)

[‡]It is worth mentioning that such an integer does not always exist. See (Li et al., 2022) for more details.

for more details on the specific update functions used in the original RNADE formulation). The softmax function (Bridle, 1990) ensures the mixing weights $\boldsymbol{\beta}$ are positive and sum to one and the exponential ensures the variances are positive. RNADE is trained to minimize the negative log likelihood: $\mathcal{L}(x_1 \cdots x_n, \theta_i) = -\sum_{i=1}^{n} \log(p_M(x_i|\theta_i))$ via gradient descent.

## 3. Methodology

To approximate density functions with CWFA, we need to improve the expressivity of the model and constrain it to compute a valid density function. In this section, we first introduce nonlinear continuous weighted finite automata. Then, we present RNADE-NCWFA for sequential density approximation, which combines CWFA with the RNADE framework. In the end, we show that RNADE-NCWFA is strictly more expressive than Gaussian HMM and present our spectral learning based algorithm for learning RNADE-NCWFA.

### 3.1. Nonlinear Continuous Weighted Finite Automata (NCWFAs)

To leverage CWFAs to estimate density functions, we first need to improve the expressivity of the model. We will do so by introducing a nonlinear feature map as well as a nonlinear termination function. We hence propose the nonlinear continuous weighted finite automata (NCWFA) model as the following:

**Definition 4** *A nonlinear continuous weighted finite automaton (NCWFA) is defined by a tuple $\tilde{A} = \langle \boldsymbol{\alpha}, \xi, \phi, \mathcal{A} \rangle$, where $\boldsymbol{\alpha} \in \mathbb{R}^k$ is the initial weight, $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ is the feature map, $\xi : \mathbb{R}^k \to \mathbb{R}^p$ is the termination function and $\mathcal{A} \in \mathbb{R}^{k \times d \times k}$ is the transition tensor. Given a sequence $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l$, the function that the NCWFA $\tilde{A}$ computes is:*

$$\boldsymbol{h_0} = \boldsymbol{\alpha}, \qquad \boldsymbol{h_t} = \mathcal{A} \bullet_1 \boldsymbol{h}_{t-1} \bullet_2 \phi(\boldsymbol{x}_t), \qquad f_{\tilde{A}}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l) = \xi(\boldsymbol{h}_l). \tag{5}$$

One immediate observation is that we can exactly recover the definition of a CWFA by letting $\phi(\boldsymbol{x}_i) = \boldsymbol{x}_i$ and $\xi(\boldsymbol{h}) = \boldsymbol{h}^\top \boldsymbol{\Omega}$.

### 3.2. Density Estimation with NCWFAs

The second problem to tackle is that we need to constrain the NCWFA so that it can tractably compute a density function. In this section, we will leverage the RNADE method to propose the RNADE-NCWFAs model. The proposed model is flexible and can compute sequential densities of arbitrary sequence length. Moreover, we will show that this model is strictly more expressive than the classic Gaussian HMM model.

Recall the core idea of RNADE is to estimate the joint density using the chain rule as in Equation 2. Instead of approximating the conditionals via the classic RNADE treatment as in Equations 3, we use an NCWFA $\tilde{A} = \langle \boldsymbol{\alpha}, \xi, \phi, \boldsymbol{\mathcal{A}} \rangle$, i.e., $p(\boldsymbol{x}_i|\boldsymbol{x}_{<i}) = f_{\tilde{A}}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_i)$. One key difference with the classic RNADE model is that the state update function is independent of the time step, allowing the model to generalize to sequences of arbitrary lengths. However, an NCWFA does not readily compute a density function, as the function does not necessarily integrates to one and the output is non-negative. To overcome this issue, we adopt the approach used in RNADE by constraining the output of the NCWFA to be a mixture of Gaussians with diagonal covariance matrices:

$$\phi(\boldsymbol{x}_i) = \tanh(\boldsymbol{x}_i^\top \mathbf{W}), \quad \boldsymbol{h}_i = \boldsymbol{\mathcal{A}} \bullet_1 \boldsymbol{h}_{i-1} \bullet_2 \phi(\boldsymbol{x}_i), \quad \boldsymbol{\beta}_i = \text{softmax}(\mathbf{V}_i^\beta \boldsymbol{h}_{i-1} + \boldsymbol{b}_i^\beta) \tag{6}$$

$$\mathbf{M}_i = \boldsymbol{\mathcal{V}}^\mu \bullet_1 \boldsymbol{h}_{i-1} + \mathbf{B}^\mu, \quad \boldsymbol{\Sigma}_i = \exp(\boldsymbol{\mathcal{V}}^\sigma \bullet_1 \boldsymbol{h}_{i-1} + \boldsymbol{\mathcal{B}}^\sigma) \tag{7}$$

$$\xi(\boldsymbol{x}_i, \boldsymbol{h}_{i-1}) = \sum_{j=1}^{m} \boldsymbol{\beta}_i^j \mathcal{N}(\boldsymbol{x}_i | \mathbf{M}_i^j, \text{diag}(\boldsymbol{\Sigma}_i^j)), \qquad f_{\tilde{A}}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l) = \xi(\boldsymbol{x}_l, \boldsymbol{h}_{l-1}) \tag{8}$$

where $\boldsymbol{h}_0 = \boldsymbol{\alpha}, \boldsymbol{\mathcal{V}}^\mu \in \mathbb{R}^{k \times m \times d}, \boldsymbol{\mathcal{V}}^\sigma \in \mathbb{R}^{k \times m \times d}, \mathbf{B}^\mu \in \mathbb{R}^{m \times d}, \boldsymbol{\mathcal{B}}^\sigma \in \mathbb{R}^{m \times d}$, $\boldsymbol{\mu}_i^j = (\mathbf{M}_i)_{:,j} \in \mathbb{R}^d, \boldsymbol{\Sigma}_i^j = (\boldsymbol{\Sigma}_i)_{:,j} \in \mathbb{R}^d$. diag is defined to be $\text{diag}(\boldsymbol{\Sigma}_i^j) = (\boldsymbol{\Sigma}_i^j \otimes \mathbf{1}) \circ \mathbf{I}$, where $\circ$ denotes the Hadamard product, $\mathbf{1} \in \mathbb{R}^d$ is an all one vector and $\mathbf{I} \in \mathbb{R}^{d \times d}$ is an identity matrix. For simplicity, we let $d' = d$ and approximate each conditional via a mixture of Gaussian with diagonal covariance matrix. This can be changed to a full covariance matrix, should the corresponding assumption (positive semi-definite) of the matrix is satisfied. Note this simplification does not affect the expressiveness of the model, as a GMM with a diagonal covariance matrix is also an universal approximator for densities and can approximate a GMM with a full covariance matrix (Benesty et al., 2008), given enough states. Under this definition, it is easy to show that $\prod_{i=1}^{l} f_{\tilde{A}}(\boldsymbol{x}_{\leq i})$ computes the density of the sequence $\boldsymbol{x}_{\leq l}$, where $\boldsymbol{x}_{\leq l}$ denotes $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l$. We will refer to this NCWFA model with RNADE structure as RNADE-NCWFA of $k$ states and $m$ mixtures. Note although the definitions of $\boldsymbol{\beta}_i$, $\mathbf{M}_i$ and $\boldsymbol{\Sigma}_i$ takes specific forms, in practice, one can use any differentiable function of $\boldsymbol{h}_i$ to compute $\boldsymbol{\beta}_i$, $\mathbf{M}_i$ and $\boldsymbol{\Sigma}_i$, so long as $\boldsymbol{\beta}_i$ sums to one and $\boldsymbol{\Sigma}_i$ is positive.

One natural question to ask is how expressive this model is. We show in the following theorem (proof in Appendix A) that RNADE-NCWFA is strictly more expressive than Gaussian HMMs, which is well known for sequential modeling (Bilmes et al., 1998).

**Theorem 5** *Given a Gaussian HMM with $k$ states $\eta = \langle \boldsymbol{\mu}, \mathbf{T}, O \rangle$, where $O : \mathbb{R}^k \times \mathbb{R}^d \to \mathbb{R}^+$ is the Gaussian emission function, $\mu \in \mathbb{R}^k$ is the initial state distribution and $\mathbf{T} \in [0,1]^k$ is the transition matrix, there exists a $k$ states $k$ mixtures RNADE-NCWFA $\tilde{A} = \langle \boldsymbol{\alpha}, \xi, \phi, \boldsymbol{\mathcal{A}} \rangle$ with full covariance matrices such that the density function over all possible trajectories generated by $\eta$ can be computed by $\tilde{A}$: $p^\eta(\boldsymbol{o}_1 \cdots \boldsymbol{o}_n) = \prod_{i=1}^{n} f_{\tilde{A}}(\boldsymbol{o}_{\leq n})$ for any trajectory $\boldsymbol{o}_1 \cdots \boldsymbol{o}_n$. Moreover, there exists a RNADE-NCWFA $\tilde{A}$ such that no Gaussian HMM model can compute its density.*

Note that a CWFA cannot compute the density function of a Gaussian mixture. Indeed, the function computed by a CWFA on a sequence of length 1 is linear in its input, whereas a RNADE-NCWFA associate such an input to a Gaussian density.

To learn RNADE-NCWFA, we want to maximize the likelihood given some training set $D_l = \{\boldsymbol{x}_{\leq l}^1, \cdots, \boldsymbol{x}_{\leq l}^N\}$ of length-$l$ sequences of $d$ dimensional vectors, i.e., $\boldsymbol{x}_{\leq l}^n = \boldsymbol{x}_1^n, \cdots, \boldsymbol{x}_l^n$, where each $\boldsymbol{x}_i^n \in \mathbb{R}^d$. More specifically, we want to minimize the negative log likelihood function: $\mathcal{L}(\tilde{A}, D) = -\sum_{i=1}^{N} \sum_{j=1}^{l} \log(f_{\tilde{A}}(\boldsymbol{x}_{\leq j}^i))$. One straight-forward solution is to use gradient descent to optimize this loss function. However, as pointed out in (Bengio et al., 1994), due to repeated multiplication by the same transition tensor, gradient descent is prone to suffer from the vanishing gradient problem and to fail in capturing long term dependencies. One alternative is the classic spectral learning algorithm for WFAs. Recall that the spectral learning method for CWFA requires to first learn Hankel tensors of length $L$, $2L$ and $2L+1$ and then perform a rank factorization on the learnt Hankel tensor to recover the CWFA parameters (see (Li et al., 2022)). However, due to the nonlinearity

added to the model, namely the feature map $\phi$ and the termination function $\xi$, spectral learning alone will not be enough. To circumvent this issue, we present an algorithm jointly leveraging gradient descent and spectral learning. The idea is to first learn the Hankel tensors of various length and the function $\phi$ and $\xi$ using gradient descent. Then we use the spectral learning algorithm to recover the transition tensor and the initial weights.

Let $\delta$ and $\omega$ denote the parameters of the mappings $\phi$ and $\xi$, respectively (see Eq. 6-8), and let $\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(l)} = [\![\boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_l^{(l)}]\!]$ be the TT form of the Hankel tensor, where $\boldsymbol{\mathcal{G}}_1^{(l)} \in \mathbb{R}^{d \times k}$ and $\boldsymbol{\mathcal{G}}_i^{(l)} \in \mathbb{R}^{k \times d \times k}$ for $i = 2, \cdots, l$. The spectral learning method for RNADE-NCWFAs first involves an approximation of the Hankel tensor via minimizing the following loss function:

$$\mathcal{L}(\delta, \omega, \boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_l^{(l)}, D_l) = -\sum_{i=1}^{N} \sum_{j=1}^{l} \log \left[ \xi \left( \psi(\boldsymbol{x}_{\leq j}^i)^\top ([\![\boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_j^{(l)}]\!])_{\langle\langle j,1 \rangle\rangle} \right) \right] \tag{9}$$

where $\psi(\boldsymbol{x}_{\leq j}) = \phi(\boldsymbol{x}_1) \otimes \cdots \otimes \phi(\boldsymbol{x}_j)$. In this process, we have obtained the Hankel tensors and the parameters of the termination function and the feature map. Then, one can perform a rank factorization on the learned Hankel tensor and recover the rest of the parameters for the RNADE-NCWFA, namely $\boldsymbol{\alpha}, \mathcal{A}$. The detailed algorithm is presented in Algorithm 1.

**Algorithm 1** `NCWFA-SL`: Spectral Learning of RNADE-NCWFA

**Input:** Three training datasets $D_L, D_{2L}, D_{2L+1}$ with input sequences of length $L$, $2L$ and $2L+1$ respectively, an encoder $\phi_\delta$, a termination function $\xi_\omega$ and TT-parameterized Hankel tensors $\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(L)}$, $\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(2L)}$ and $\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(2L+1)}$, learning rate $\gamma$, desired rank $R$

**while** Model not converging **do**
  **for** $l \in \{L, 2L, 2L+1\}$ **do**
    Update $\delta, \omega, \boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_l^{(l)}$ via gradient descent by minimize the loss function 9:
    **for** $\theta \in \{\delta, \omega, \boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_l^{(l)}\}$ **do**

$$\theta \leftarrow \theta - \gamma \nabla_\theta \mathcal{L}(\delta, \omega, \boldsymbol{\mathcal{G}}_1^{(l)}, \cdots, \boldsymbol{\mathcal{G}}_l^{(l)}, D_l)$$

Let $(\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(2L)})_{\langle\langle L, L+1 \rangle\rangle} = \mathbf{PS}$ be a rank $R$ factorization.

Return the RNADE-NCWFA $\tilde{A} = \langle \boldsymbol{\alpha}, \xi_\omega, \phi_\delta, \boldsymbol{\mathcal{A}} \rangle$ where

$$\boldsymbol{\alpha} = (\mathbf{S}^\dagger)^\top (\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(L)})_{\langle\langle L+1 \rangle\rangle}, \qquad \boldsymbol{\mathcal{A}} = ((\boldsymbol{\mathcal{H}}_{\tilde{A}}^{(2L+1)})_{\langle\langle L,1,L+1 \rangle\rangle}) \times_1 \mathbf{P}^\dagger \times_3 (\mathbf{S}^\dagger)^\top$$

## 4. Experiments

For the experiments, we conduct a synthetic experiment based on data generated from a random 10-states Gaussian HMM. We sample sequences of length 3, 6 and 7 from the HMM. To evaluate the model's performance on its generalization ability to unseen length of sequence, we sample 1,000 sequences from length 8 to length 400 from the same HMM for the test data. To test the model's resistance to noise, we inject the training samples with Gaussian noise of different standard deviations (0.1 and 1.0) with 0 mean.

For the baseline models, we have HMM learned with expectation maximization (EM) method, as it can computes density of sequences of any length by design. We also modified

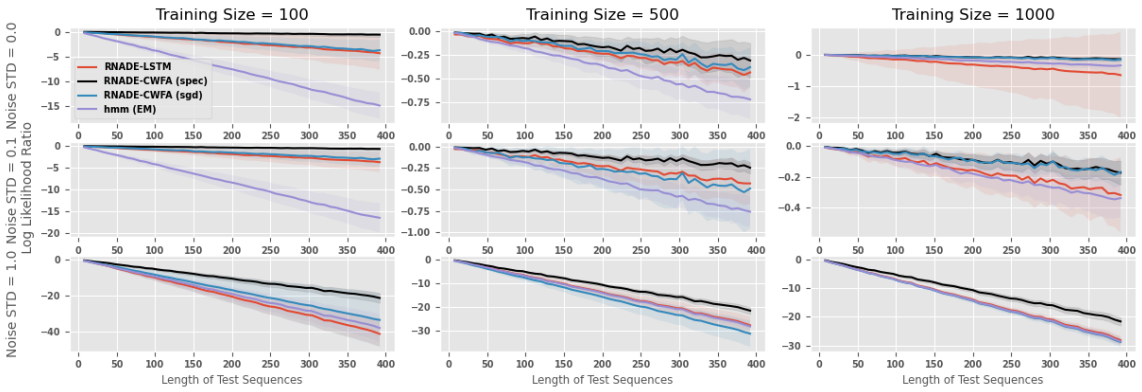


Figure 1: Log likelihood ratios between the tested models and the ground truth likelihood. We show the trend w.r.t. the length of the testing sequences under different sample sizes (columns) and standard deviations of the injected noise (rows).

the RNADE model by replacing the hidden states update rule 3 with an LSTM structure to give the RNADE model the ability to generalize to sequences of arbitrary length, regardless of the length of the training sequences. We refer to this model as RNADE-LSTM. For our model, by following Algorithm 1, we have the method RNADE-NCWFA (spec). Alternatively, although we have mentioned that training the (RNADE-)NCWFA model through pure gradient descent method can have many issues, we also list this approach of training RNADE-NCWFA as one of the baselines, referred to as RNADE-NCWFA (sgd). For all the training processes, if gradient descent is involved, we always use Adam optimizer (Kingma and Ba, 2014) with 0.001 learning rate with early stopping. For HMM as well as RNADE-NCWFA models, we set the size of the model to be 10 (ground truth of the randomly HMM). For RNADE-LSTM, we set the size of the hidden states to be 10. We present the trend of the averaged log likelihood ratio with the ground truth likelihood, i.e. $\log(\frac{\text{predicted likelihood}}{\text{ground truth}})$ w.r.t. length of the sequences over 10 seeds in Figure 1 and a snapshot of the log likelihood for each model of 400 length testing sequences in Appendix B.

From the experiment results, we can see that RNADE-CWFA (spec) consistently has the best performance across all training sizes and levels of noise injected. More precisely, this advantage is more significant when given small training sizes and (or) the data is injected with high noise. Moreover, the spectral learning algorithm shows stable training results as the standard deviation of the log likelihood (ratio) is the lowest among all methods. This is especially the case when not enough training samples are provided. In addition, one can see that this advantage is consistent with all test sequence lengths we have experimented.

## 5. Conclusion and Future Work

In this paper, we propose the RNADE-NCWFA model, an expressive and tractable WFA-based density estimator over sequences of continuous vectors. We extend the notion of continuous WFA to its nonlinear case by introducing a nonlinear feature mapping function as well as a nonlinear termination function. We then combine the idea from RNADE to propose our density estimation model RNADE-NCWFA and its spectral learning based learning algorithm. In addition, we show that theoretically, RNADE-NCWFA is strictly more expressive than the Gaussian HMM model. We show that, empirically, our method

has great capability of generalize to sequences of varying length, which is potentially not the same as the training sequences. For future work, we are looking into more experiments on real dataset and compare with more baselines. Moreover, we did not add nonlinear transition for the NCWFA model as it would imply that the Hankel tensor will be of infinite tensor train rank, hence making the spectral learning algorithm intractable. We will be looking into possibilities of adding this nonlinearity into the NCWFA model and have a working algorithm for it. In addition, we would like to examine more closely in terms of the expressivity of the RNADE-NCWFA.

## References

Raphaël Bailly, François Denis, and Liva Ralaivola. Grammatical inference as a principal component analysis problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 33–40. ACM, 2009.

Borja Balle, Xavier Carreras, Franco M Luque, and Ariadna Quattoni. Spectral learning of weighted automata. *Machine learning*, 96(1-2):33–63, 2014a.

Borja Balle, William L. Hamilton, and Joelle Pineau. Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In *Proceedings of ICML*, pages 1386–1394, 2014b.

Jacob Benesty, M Mohan Sondhi, Yiteng Huang, et al. *Springer handbook of speech processing*, volume 1. Springer, 2008.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Jeff A Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

Byron Boots, Sajid M Siddiqi, and Geoffrey J Gordon. Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research*, 30 (7):954–966, 2011.

John S Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer, 1990.

François Denis and Yann Esposito. On rational stochastic languages. *Fundamenta Informaticae*, 86(1, 2):41–77, 2008.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Hadrien Glaude and Olivier Pietquin. PAC learning of probabilistic automaton based on the method of moments. In *Proceedings of ICML*, pages 820–829, 2016.

Daniel J. Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. In *Proceedings of COLT*, 2009.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Tianyu Li, Doina Precup, and Guillaume Rabusseau. Connecting weighted automata, tensor networks and recurrent neural networks through spectral learning. *Machine Learning*, 2022.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33 (5):2295–2317, 2011.

Guillaume Rabusseau, Tianyu Li, and Doina Precup. Connecting weighted automata and recurrent neural networks through spectral learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1630–1639. PMLR, 2019.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Michael Thon and Herbert Jaeger. Links between multiplicity automata, observable operator models and predictive state representations: a unified learning framework. *The Journal of Machine Learning Research*, 16(1):103–147, 2015.

Benigno Uria, Iain Murray, and Hugo Larochelle. Rnade: The real-valued neural autoregressive density-estimator. *Advances in Neural Information Processing Systems*, 26, 2013.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *The Journal of Machine Learning Research*, 17(1):7184–7220, 2016.

## Appendix A. Proof of Theorem

**Theorem 6** *Given a Gaussian HMM with $k$ states $\eta = \langle \boldsymbol{\mu}, \mathbf{T}, O \rangle$, where $O : \mathbb{R}^k \times \mathbb{R}^d \to \mathbb{R}^+$ is the Gaussian emission function, $\mu \in \mathbb{R}^k$ is the initial state distribution and $\mathbf{T} \in [0,1]^k$ is the transition matrix, there exists a $k$ states $k$ mixtures RNADE-NCWFA $\tilde{A} = \langle \boldsymbol{\alpha}, \xi, \phi, \boldsymbol{\mathcal{A}} \rangle$ with full covariance matrices such that the density function over all possible trajectories generated by $\eta$ can be computed by $\tilde{A}$:*

$$p^{\eta}(\boldsymbol{o}_1 \cdots \boldsymbol{o}_n) = \prod_{i=1}^{n} f_{\tilde{A}}(\boldsymbol{o}_{\leq n})$$

*for any trajectory $\boldsymbol{o}_1 \cdots \boldsymbol{o}_n$. Moreover, there exists a RNADE-NCWFA $\tilde{A}$ such that no Gaussian HMM model can compute its density.*

**Proof** For the Gaussian HMM $\eta$, given an observation sequences $\boldsymbol{o}_1 \cdots \boldsymbol{o}_n$, its density under $\eta$ is:

$$p^{\eta}(\boldsymbol{o}_1 \cdots \boldsymbol{o}_n) = O(\boldsymbol{m}^\top, \boldsymbol{o}_1) O(\boldsymbol{m}^\top \mathbf{T}, \boldsymbol{o}_2) \cdots O(\boldsymbol{m}^\top \mathbf{T}^{n-1}, \boldsymbol{o}_n),$$

where $O(\boldsymbol{h}, \boldsymbol{o}) = \sum_{i=1}^{k} \boldsymbol{h}_i \mathcal{N}(\boldsymbol{o} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$ for some mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Let $\boldsymbol{\alpha} = \boldsymbol{m}$, $\mathcal{A}_{:,i,:} = \mathbf{T}$ for $i \in [k]$, $\phi(\boldsymbol{x}) = [\frac{1}{k}, \frac{1}{k}, \cdots, \frac{1}{k}]^\top$ and $\xi = O$. Note it reasonable to let $\xi = O$, since as long as we let $\boldsymbol{\beta}_i = \boldsymbol{\alpha}^\top \mathbf{T}^{i-1}$, $\boldsymbol{\beta}_0 = \boldsymbol{\alpha}^\top$, $\boldsymbol{\mu}_i = \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}$, following equations 7, then for any $\boldsymbol{h} \in \mathbb{R}^k, \boldsymbol{o} \in \mathbb{R}^d$, we have $\xi(\boldsymbol{h}, \boldsymbol{o}) = O(\boldsymbol{h}, \boldsymbol{o})$. Then under this parameterization, we have $\mathcal{A} \bullet_2 \phi(\boldsymbol{o}_j) = \mathbf{T}$. Then the RNADE-NCWFA computes the following function:

$$\begin{aligned} f_{\tilde{A}}(\boldsymbol{o}_1, \cdots, \boldsymbol{o}_i) &= \xi((\mathcal{A} \bullet_1 \boldsymbol{\alpha}^\top \bullet_2 \phi(\boldsymbol{o}_1))^\top (\mathcal{A} \bullet_2 \phi(\boldsymbol{o}_2)) \cdots (\mathcal{A} \bullet_2 \phi(\boldsymbol{o}_{i-1})), \boldsymbol{o}_i) \\ &= \xi(\boldsymbol{\alpha}^\top \mathbf{T}^{i-1}, \boldsymbol{o}_i) = O(\boldsymbol{m}^\top \mathbf{T}^{i-1}, \boldsymbol{o}_i) \end{aligned}$$

Therefore, we have:

$$\begin{aligned} p^{\eta}(\boldsymbol{o}_1 \cdots \boldsymbol{o}_n) &= O(\boldsymbol{m}^\top, \boldsymbol{o}_1) O(\boldsymbol{m}^\top \mathbf{T}, \boldsymbol{o}_2) \cdots O(\boldsymbol{m}^\top \mathbf{T}^{n-1}, \boldsymbol{o}_n) \\ &= f_{\tilde{A}}(\boldsymbol{o}_1) f_{\tilde{A}}(\boldsymbol{o}_1, \boldsymbol{o}_2) \cdots f_{\tilde{A}}(\boldsymbol{o}_1, \cdots, \boldsymbol{o}_n) = \prod_{i=1}^{n} f_{\tilde{A}}(\boldsymbol{o}_{\leq n}) \end{aligned}$$

For the proof of the second half of the theorem, consider a shifting Gaussian HMM, where the mean vector of the Gaussian emission is a function of the time steps, i.e., $\boldsymbol{\mu} = q(i)$, where $i = 1, 2, \cdots$. For simplicity, assume the shifting Gaussian HMM is for one dimensional sequences and has one mixture. In addition, let $q(i) = i$ and assume the variance is 1. Then the emission function can be written as $O^t(o) = \mathcal{N}(o | t, 1)$. Then the density of a sequence $o_1, \cdots, o_n$ under this shifting Gaussian HMM $\eta_s$ is:

$$p^{\eta_s}(o_1, \cdots, o_n) = O^1(o_1) O^2(o_2) \cdots O^n(o_n).$$

We show that this density cannot be computed by a Gaussian HMM of finite states. If $p^{\eta_s}$ can be computed by a Gaussian HMM, then for the mean vector $\boldsymbol{\mu}$ there exists an initial

weight vector $\boldsymbol{m}$, a transition matrix $\mathbf{T}$ satisfying the following linear system:

$$\begin{cases} \boldsymbol{m}^\top \boldsymbol{\mu} & = 1 \\ \boldsymbol{m}^\top \mathbf{T} \boldsymbol{\mu} & = 2 \\ & \vdots \\ \boldsymbol{m}^\top \mathbf{T}^{n-1} \boldsymbol{\mu} & = n \\ & \vdots \end{cases}$$

This linear system is, however, overdetermined, as $\boldsymbol{\mu}$ is a vector of finite size, while there are infinite linearly independent equations to satisfy. Therefore, a Gaussian HMM of finite states cannot compute the density function of a shifting Gaussian HMM.

We now show such density can be computed by a RNADE-NCWFA. Let $\boldsymbol{\alpha}^\top = [1, 1]$, and $\mathcal{A}_{:,i,:} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $\boldsymbol{\mu}_i = \langle \boldsymbol{h}_{i-1}, [0, 1] \rangle$, $\phi(o)^\top = [0.5, 0.5]$, $\boldsymbol{\Sigma}_i = 1$. Then we have:

$$\begin{aligned} f_{\tilde{A}}(o_1, \cdots, o_i) &= \xi((\mathcal{A} \bullet_1 \boldsymbol{\alpha}^\top \bullet_2 \phi(o_1))^\top (\mathcal{A} \bullet_2 \phi(o_2)) \cdots (\mathcal{A} \bullet_2 \phi(o_{i-1})), o_i) \\ &= \xi(\boldsymbol{\alpha}^\top \mathbf{T}^{i-1}, o_i) = \xi([1, i], o_i) = \mathcal{N}(o_i | i, 1) \end{aligned}$$

Therefore:

$$\begin{aligned} p^{\eta_s}(o_1, \cdots, o_n) &= \mathcal{N}(o|1,1)\mathcal{N}(o|1,2) \cdots \mathcal{N}(o|1,n) \\ &= f_{\tilde{A}}(o_1) f_{\tilde{A}}(o_1, o_2) \cdots f_{\tilde{A}}(o_1, \cdots, o_n) = \prod_{i=1}^{n} f_{\tilde{A}}(\boldsymbol{o}_{\leq n}) \end{aligned}$$

Therefore, for the given shifting Gaussian HMM density, it can be computed by a RNADE-NCWFA, but cannot be computed by a Gaussian HMM with finite states. ■

## Appendix B. Experiment Results in Table

In this section, we show a snapshot of the results we show in the previous experiment results in Figure 1. The result is listed in Table 1. The reported likelihood is evaluated on test sequence of length 400. From this table, we can see more clearly the advantage of our RNADE-NCWFA model when trained with spectral learning algorithm.

Table 1: Comparison of model performance in term of average log likelihood (in NAT). Different models are compared under different training sizes and level of noise injected. The reported likelihood (mean (standard deviation)) is evaluated on test sequence of length 400.

| Training Size | 100 | | | 500 | | | 1000 | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise Std | 0 | 0.1 | 1 | 0 | 0.1 | 1 | 0 | 0.1 | 1 |
| HMM (EM) | -615.26 (2.57) | -616.88 (3.40) | -638.44 (7.50) | -601.15 (0.20) | -601.18 (0.17) | -628.75 (1.51) | -600.70 (0.12) | -600.69 (0.12) | -628.91 (1.13) |
| RNADE-LSTM | -604.71 (3.36) | -604.10 (2.29) | -641.72 (7.17) | -600.86 (0.15) | -600.85 (0.23) | -628.28 (3.56) | -601.01 (1.34) | -600.67 (0.24) | -628.45 (1.56) |
| RNADE-NCWFA (spec) | **-600.96 (0.42)** | **-601.06 (0.24)** | **-621.75 (2.71)** | **-600.73 (0.18)** | **-600.67 (0.067)** | **-622.10 (1.44)** | **-600.50 (0.49)** | -600.53 (0.07) | **-621.91 (1.13)** |
| RNADE-NCWFA (sgd) | -604.11 (2.13) | -603.29 (1.69) | -633.96 (14.6) | -600.81 (0.20) | -600.91 (0.46) | -631.80 (5.53) | -600.51 (0.06) | **-600.52 (0.08)** | -629.11 (1.65) |
| Ground Truth | -600.40 | -600.40 | -600.40 | -600.40 | -600.40 | -600.40 | -600.40 | -600.40 | -600.40 |