Neural Network Based Decoding over molecular communication Channels

Peter Hartig

February 26, 2020

Abstract

Estimation of the probability distribution function characterizing a communication channel is investigated. In this work, pilot symbol sequences are used to train a neural network and mixture model to estimate unknown channel probability distribution functions. The estimated function is then evaluated using the resulting distribution as a metric for the Viterbi algorithm. In particular, the detection performance is investigated for non-linear channels relevant for the molecular communications domain. A proposal is then made to reduce the complexity of the resulting detection using a clustering of channel states for the Viterbi algorithm.

Contents

0.1 Notation

The following notation is used throughout this work. $p(x)$ is the probability of an event x. $p(x|y)$ is the conditional probability of x given y. $E[x]$ is the expected value of a random variable x.

Vectors are denoted by bold font lower-case letters ($\mathbf{x}$) and are assumed column vectors. The vector $\mathbf{x}_{\mathrm{i}}^{\mathrm{j}}$ denotes a vector containing the elements i through j of $\mathbf{x}$. $|\mathcal{A}|$ denotes the cardinality of the set $\mathcal{A}$. $\underset{x}{\operatorname{argmin}}\, f(x)$ is the value of variable x which minimizes the function $f(x)$.

0.2 Introduction

(TODO Introduce the MC channel completely in this section)

Characterizing the probability distribution governing communication channels is a fundamental barrier to communication. While optimal and sub-optimal strategies for overcoming this barrier have enabled vast and effective communication infrastructure, this barrier still limits communication in many contexts. One such context is the molecular communication channel which may be non-linear, time variant, and dependent on the specific transmitted information [**?**].

Communication contexts, such as wireless, have successfully used "pilot" symbol-streams, known to the receiver, to estimate channel information. The overhead of transmitting pilot sequences is, however, incompatible with the low symbol rate (and relative coherence time?) of molecular communication channels. An alternative data-driven method for characterizing molecular communication channels is based on neural networks. Neural networks have shown to be an effective tool in data-driven approximation of probability distributions (TODO cite).

The general communication channel is equivalent to a conditional probability $p(\mathbf{x}|\mathbf{y})$, with transmitted information $\mathbf{x}$ and received information $\mathbf{y}$ [**?**, Ch. 7]. $p(\mathbf{x}|\mathbf{y})$ takes into account the (potentially random) channel through which the information $\mathbf{x}$ passes in reaching the receiver. Detecting the most probable transmitted information from the received information is formalized as

$$\underset{x\in\mathcal{A}}{\operatorname{argmin}}\, p(\mathbf{x}|\mathbf{y}),$$

in which $\mathcal{A}$ is the set of all possible transmitted information $\mathbf{x}$. In general, sub-optimal solutions do not require perfect knowledge of the distribution $p(\mathbf{x}|\mathbf{y})$ and may be used when the true $p(\mathbf{x}|\mathbf{y})$ is unknown or impractical to

obtain (TODO CITE). This work investigates a neural network estimate of $p(\mathbf{x}|\mathbf{y})$.

(TODO Develop MAP vs MLSE here)

0.3 Background

The following section first introduces the optimization problem used in the remainder of this work. An efficient implementation for solving the optimization problem is then introduced. With this foundation in place, the data-driven distribution estimation is then incorporated into the algorithm.

0.3.1 MLSE and the Viterbi Algorithm

The optimization problem

$$\underset{x\in\mathcal{A}}{\operatorname{argmin}}\ p(\mathbf{y}|\mathbf{x}), \tag{1}$$

with transmitted information vector $\mathbf{x}$ and received information vector $\mathbf{y}$, is known as Maximum Likelihood Sequence Estimation (MLSE). The size of the search space $|\mathcal{A}|$ grows exponentially in the length of $\mathbf{x}$, the number of the transmitted symbols. Additional information about the communication channel can reduce the complexity of this search as illustrated in the following example.

Consider the communication channel for which each received symbol in the sequence $\mathbf{y}$ is a causal, linear, and time invariant (LTI) combination of a set of the transmitted symbol sequence $\mathbf{x}$ weighted by coefficients $\mathbf{a}$.

$$y[k] = \sum_{\mathrm{l}=1}^{\mathrm{L}} a[l]x[k-l].$$

For this channel,

$$\underset{\mathbf{x}\in\mathcal{A}}{\operatorname{argmin}}\ p(\mathbf{y}|\mathbf{x}) = \underset{\mathbf{x}\in\mathcal{A}}{\operatorname{argmin}}\ \sum_{\mathrm{i}=1}^{\mathrm{N}} \log(p(y_{\mathrm{i}}|\mathbf{x}_{\mathrm{i}-\mathrm{L}+1}^{\mathrm{i}})).$$

Noting that individual terms of the sum are statistically independent, and that the set of possible transmitted sequences $\mathbf{x}_{\mathrm{i}-\mathrm{L}+1}^{\mathrm{i}}$ is equivalent to a set of channel states, problem (1) is equivalent finding the shortest path through a trellis with edges weighted by $\log(p(y_{\mathrm{i}}|\mathbf{x}_{\mathrm{i}-\mathrm{L}+1}^{\mathrm{i}}))$, as in molecular communication 1. (TODO: Does this figure need further explanation?)

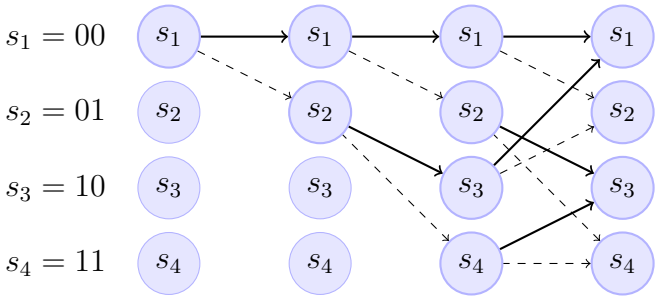

Figure 1: MLSE decoding for $\mathcal{A} = \{0, 1\}$ and $L = 2$. Each state $s_1 ... s_4$ represents a possible $\mathbf{x}_{\mathrm{i}-\mathrm{L}+1}$.

For finite state, causal channels, MLSE reduces to the Viterbi Algorithm over a trellis of states.

Viterbi Algorithm: (TODO: Decide if this is clear enough)

---

given $p(y_{\mathrm{i}}|x^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1})\ \forall\ i \in 1..N$.
for $i = 1...N$
  for each state $s \in \mathcal{A}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1}$
    1. Let $\mathrm{cost}^{\mathrm{i}}_s = -\log(p(y_{\mathrm{i}}|x^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1})) + \min_s\{\text{incoming costs}^{\mathrm{i-1}}_s\}$
return detected transmission $\hat{\mathbf{x}}$ corresponding to trellis path of $\operatorname{argmin}_s \mathrm{cost}^N_s$

Note that the Viterbi algorithm is *exponentially* complex in the number of channel states $|\mathcal{A}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1}|$, but *linearly* complex in the length of the transmitted sequence $\mathbf{x}$ (N in the algorithm above).

0.3.2 ViterbiNet

Despite the reduction in MLSE complexity using the Viterbi Algorithm, the metrics $p(y_{\mathrm{i}}|\mathbf{x}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1})$, used in each step of the algorithm, require knowledge of the channel probability distribution. This may be difficult or costly to obtain. By decomposing with Bayes' theorem, we can instead estimate the individual terms of

$$p(y_{\mathrm{i}}|\mathbf{x}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1}) = \frac{p(\mathbf{x}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1}|y_{\mathrm{i}})p(y_{\mathrm{i}})}{p(\mathbf{x}^{\mathrm{i}}_{\mathrm{i}-\mathrm{L}+1})}.$$

- $p(\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}}|y_{\mathrm{i}})$: The probability of being in channel state $\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}}$ given the corresponding received symbol y_{i}. If the number of states is finite, the probability mass function over $\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}} \in |\mathcal{A}^{\mathrm{i}}_{\mathrm{i-L+1}}|$ can be estimated using a neural network for classification as in Figure 2.

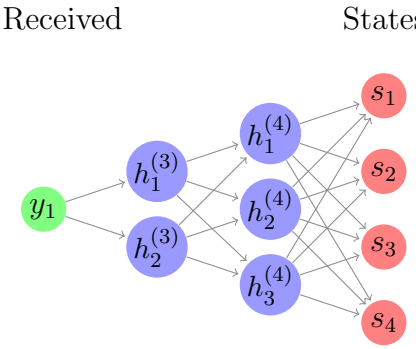

Figure 2: State classification Neural Network.

- $p(y_{\mathrm{i}}) = \sum_{s \in \mathcal{S}} p(s, y_{\mathrm{i}})$: The joint probability of channel state s and received signal y_{i} marginalized over all channel states $\mathcal{S}$. For the case of the LTI channel above, each state s corresponds to a possible symbol sequence $\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}}$. This probability distribution function can be estimated using a mixture-model (Figure 3) based on a model for channel noise and training data of received signals (in this case the same data used to train the neural network). The Expectation Maximization Algorithm [Ng00] is used with a chosen model for the channel states and noise. In the case of Gaussian noise, as used here, each state is modeled as a Gaussians. As $p(y_{\mathrm{i}})$ is constant over all states in a given step of the Viterbi algorithm, it may be interpreted as a weighting factor for the cost of a received symbol relative to other symbols in the path. (TODO: More detail on this or enough?)

communications/system$_m$odel/mm.pdfcommunications/system$_m$odel/mm.pngcommunications

Figure 3: A mixture of complex, Gaussian sources that can be estimate using the Expectation Maximization algorithm

- $p(\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}})$: The probability of a given transmitted sequence can be neglected for equiprobable transmit symbols.

In summary, the metrics $p(y_{\mathrm{i}}|\mathbf{x}^{\mathrm{i}}_{\mathrm{i-L+1}})$ required by the Viterbi algorithm can be estimated using a neural network and a mixture model.

0.3.3 A Reduced State ViterbiNet

While the Viterbi Algorithm complexity scales exponentially in the number of states possible in each step of the algorithm, in some cases this complexity can be reduced without significant degradation of performance. In particular, if the system is such that some states are redundant, these can be combined. The following, relevant example with state redundancy is posed and one potential method of reducing the number of states is considered.

Consider a received signal

$$y[k] = \sum_{\mathrm{l}=1}^{\mathrm{L}} a[l]x[k-l] + n[k],\ n[k] \sim \mathcal{N}(0,1)$$

with $x[k-l] \in \{-1,+1\}$ and $n[k] \sim \mathcal{N}(0,1)$.

In the case of a causal, LTI system with inter-symbol-interference characterized by $\mathbf{a} = [a[1]...a[5]] = [1, 0, .2, .2, .4]$ ($\|\mathbf{a}\|_2^2 = 1$), the received signal (Fig.**??**) has fewer than the potential $|\mathcal{A}^{\mathrm{i}}_{\mathrm{i-L+1}}| = 2^5$ states as defined above. As one of the channel taps is 0, this will have no impact and thus 16 of the potential 32 states are removed. Further, as there are two taps of value 0.2, these together represent only 3 states rather than 4.

One way to exploit state redundancy is to cluster the original states $|\mathcal{A}^{\mathrm{i}}_{\mathrm{i-L+1}}|$ states into a set of k clusters using training data. The K-Means algorithm is proposed as one such method.

K-Means Clustering Algorithm: A method for unsupervised classification

---

given initial location $L_c, \forall\, c \in \{1..K\}$ of K centroids.
for Number of Iterations.
  for each training data point x_{i}, $\mathrm{i} \in \{1..N\}$
    1. Label x_i with the index c of the closest centroid using $\|x_{\mathrm{i}} - \mathrm{L}_c\|_2^2$.
  for centroid $L_c, \forall\, c \in \{1..K\}$
    1. Move L_{c} to the average location of all training points with label .c
return Centroid locations.

Choosing an appropriate number of clusters will influence the performance of the resulting decoder (Have figure showing this).

An implementation point to note is that after clustering the training data into K states, the number of states must be increased to $(|\mathcal{A}_{\mathrm{i}}|)\mathrm{K}$. In the above example 2K. The Viterbi algorithm selects a symbol sequence corresponding to a path of state transitions through the trellis. In order

to make the correspondence between a unique path and a symbol sequence, each state transition must represent a transmitted symbol. *Each* resulting centroid from the k-means clustering is associated to *each* potential transmit symbol in order to create a "shorter" trellis for which each state transition still represents a transmitted symbol.

(TODO:Discuss minimum phase representations for channels and why this might be an advantage?)

0.4 Results

0.4.1 A supervised equalization benchmark

The Linear Minimum Mean-Squared Error (LMMSE) equalizer derived below is used provide a linear equalization reference to the non-linearity of the proposed neural network, mixture model equalization.

Given an estimate of channel memory length L, the convex and continuously differentiable optimization problem

$$\underset{\mathbf{h}\in\mathcal{C}^L}{\operatorname{argmin}}\ E[\|x-\mathbf{h}^T\mathbf{y}\|^2],$$

minimizes the squared error of the linearly detected symbol $\mathbf{h}^T\mathbf{y}$ and true symbol x. Using

$$\frac{\partial E[\|x-\mathbf{h}^T\mathbf{y}\|^2]}{\partial \mathbf{h}}=0$$

the optimal LMMSE equalizer is

$$\mathbf{h}=E[\mathbf{R}_{yy}]^{-1}E[\mathbf{r}_{yx}].$$

$E[\mathbf{R}_{yy}]$ and $E[\mathbf{r}_{yx}]$ are estimated by

$$E[\mathbf{R}_{yy}]=\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\mathbf{y}^{\mathrm{i}}_{\mathrm{i\text{-}L+1}}{\mathbf{y}^{\mathrm{i}}_{\mathrm{i\text{-}L+1}}}^{H}$$

and

$$E[\mathbf{r}_{yx}]=\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}\mathbf{y}^{\mathrm{i}}_{\mathrm{i\text{-}L+1}}x_{\mathrm{i}};$$

using the same training data as the neural network and mixture model.

0.4.2 Simulation Details

System Model

Verification and further testing of the implementation and integration of the algorithms described above is performed using the following system model.

$$y[\mathrm{k}] = f_{\mathrm{k}}(\mathbf{x}) + n[\mathrm{k}].$$

With $x[\mathrm{k}] \in \{-1, +1\}$, independent $n[\mathrm{k}] \sim \mathcal{N}(0, 1)$ and

$$\mathrm{SNR} = \frac{E\{|x[\mathrm{k}]|^2\}}{\sigma^2}.$$

Note that each received signal $y[\mathrm{k}]$ is potentially a function of all input symbols (assume causality?). Decoding performance for different values of the channel function $f_{\mathrm{k}}()$ are evaluated.

Algorithm Parameters

The parameters for the algorithms used in this work are detailed below.

- **Neural Network**
  - Architecture: 4 layers 1, 100 (Tanh activation), 50 (relu activation), M^L (Softmax $\rightarrow$ Negative Log-Likelihood)
  - Training Data Examples: 5000
  - Neural Network Updates: Admm [KB14] with step size 10^{-2}
  - Batch Size: 1000
  - Backpropogation Updates (Epochs): 900
  - Loss Function: Cross Entropy
- **Expectation Maximization Algorithm**
  - Training Data Examples: 5000
  - Mixture Model Source Types: Gaussian
  - Number of Sources: Number of channel states
- **K-Means Algorithm (For Reduced State ViterbiNet)**
  - Training Data Examples: 5000

0.4.3 Simulation Results

We first evaluate the detector performance using the Linear Time Invariant channel

$$y[\mathrm{k}] = \sum_{\mathrm{l}=1}^{\mathrm{L}} a[l]x[\mathrm{k} - l] + n[\mathrm{k}].$$

with impulse response $a[l] = ...$ Simulation results are shown in fig 4

communications/results/lti$_n$*ormal.pdfcommunications/results/lti*$_n$*ormal.pngcommunications/*

Figure 4: Detection performance over LTI channel with AWGN

LTI + Quantizer Channel

To evaluate decoder performance in the non-linear channel setting, we apply a quantizer, $Q()$, to the output of the previous LTI + AWGN channel.

$$y[\mathrm{k}] = Q(\sum_{\mathrm{l}=1}^{\mathrm{L}} a[l]x[\mathrm{k} - l]) + n[\mathrm{k}].$$

Here we evaluate performance under two levels of quantization severity: rounding channel output down to a given number of decimal places . Figure 5 depicts the quantizer and its impact on the received symbols from the LTI channel. The resulting performance is seen in Figure 6. Notice that this is essentially a reduction of the original channel states as there is now redundancy that was not previously present.

communications/system$_m$*odel/quantizer*$_o$*verlay.pdfcommunications/system*$_m$*odel/quantizer*$_o$*v*

Figure 5: Quantized output of LTI channel overlayed on the applied quantization function. TODO decide on axis titles

communications/results/quantized$_b$*efore*$_n$*oise.pdfcommunications/results/quantized*$_b$*efore*$_n$*oi*

Figure 6: Detection performance over LTI channel with AWGN and quantization

molecular communication Channel

Lastly, the performance of the neural network based detector is evaluated data collected from a molecular communication channel. ... Explain if there are any results to report.

0.5 Conclusion

The proposed detection method from [SEFG19] is evaluated in channels for which $p(y_{\mathrm{i}}|\mathbf{x}_{\mathrm{i-L+1}}^{\mathrm{i}})$, is unknown and non-linear. Simulation results indicate that this detector can successfully learn $p(y_{\mathrm{i}}|\mathbf{x}_{\mathrm{i-L+1}}^{\mathrm{i}})$ for non-linear channels. A method for reducing the complexity of the resulting Viterbi Algorithm is also proposed, however the method of state reduction for the channel needs to be further investigated to improve the poor performance observed.

In this work the MLSE is found using the Viterbi Algorithm. The same metrics can also be used in optimizing other performance metrics. In [SFEG20] these estimate metrics have been used for minimize bit error rate estimation using the BCJR algorithm.

Extensions of specific work: More on reduced state. Using other channels and repeat online type of updates for dynamic channels for these more challenging channels. Learning channel changes in cases for which coherence time is shorter than updates to network made by pilot symbols.

General method extensions:

Bibliography

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Ng00] Andrew Ng. Cs229 lecture notes. *CS229 Lecture notes*, 1(1), 2000.

[SEFG19] Nir Shlezinger, Yonina C Eldar, Nariman Farsad, and Andrea J Goldsmith. Viterbinet: Symbol detection using a deep learning based viterbi algorithm. In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2019.

[SFEG20] Nir Shlezinger, Nariman Farsad, Yonina C. Eldar, and Andrea J. Goldsmith. Data-driven factor graphs for deep symbol detection, 2020.