Neural Network Detection of Data Sequences in Communication Systems

Nariman Farsad, *Member, IEEE,* and Andrea Goldsmith, *Fellow, IEEE*

***Abstract*—We consider detection based on deep learning, and show it is possible to train detectors that perform well, without any knowledge of the underlying channel models. Moreover, when the channel model is known, we demonstrate that it is possible to train detectors that do not require channel state information (CSI). In particular, a technique we call sliding bidirectional recurrent neural network (SBRNN) is proposed for detection where, after training, the detector estimates the data in real-time as the signal stream arrives at the receiver. We evaluate this algorithm, as well as other neural network (NN) architectures, using the Poisson channel model, which is applicable to both optical and chemical communication systems. In addition, we also evaluate the performance of this detection method applied to data sent over a chemical communication platform, where the channel model is difficult to model analytically. We show that SBRNN is computationally efficient, and can perform detection under various channel conditions without knowing the underlying channel model. We also demonstrate that the bit error rate (BER) performance of the proposed SBRNN detector is better than that of a Viterbi detector with imperfect CSI as well as that of other NN detectors that have been previously proposed.**

***Index Terms*—Machine learning, deep learning, supervised learning, communication systems, detection, molecular communication.**

I. Introduction

One of the important modules in reliable recovery of data sent over a communication channel is the detection algorithm, where the transmitted signal is estimated from a noisy and corrupted version observed at the receiver. The design and analysis of this module has traditionally relied on mathematical models that describe the transmission process, signal propagation, receiver noise, and many other components of the system that affect the end-to-end signal transmission and reception. Most communication systems today convey data by embedding it into electromagnetic (EM) signals, which lend themselves to tractable channel models based on a simplification of Maxwell's equations. However, there are cases where tractable mathematical descriptions of the channel are elusive, either because the EM signal propagation is very complicated or when it is poorly understood. In addition, there are communication systems that do not use EM wave signalling and the corresponding communication channel models may be unknown or mathematically intractable. Some examples of the latter includes underwater communication using acoustic signals [1] as well as molecular communication, which relies on chemical signals to interconnect tiny devices with sub-millimeter dimensions in environments such as inside the human body [2]–[5].

Nariman Farsad and Andrea Goldsmith are with the Department of Electrical Engineering, Stanford University, Stanford, CA, 94305. Emails: nfarsad@stanford.edu, andrea@wsl.stanford.edu

Even when the underlying channel models are known, since the channel conditions may change with time, many model-based detection algorithms rely on the estimation of the instantaneous channel state information (CSI) (i.e., channel model parameters) for detection. Typically, this is achieved by transmitting and receiving a predesigned pilot sequence, which is known by the receiver, for estimating the CSI. However, this estimation process entails overhead that decreases the data transmission rate. Moreover, the accuracy of the estimation may also affect the performance of the detection algorithm.

In this paper, we investigate how different techniques from artificial intelligence and deep learning [6]–[8] can be used to design detection algorithms for communication systems that learn directly from data. We show that these algorithms are robust enough to perform detection under changing channel conditions, without knowing the underlying channel models or the CSI. This approach is particularly effective in emerging communication technologies, such as molecular communication, where accurate models may not exist or are difficult to derive analytically. For example, tractable analytical channel models for signal propagation in chemical communication channels with multiple reactive chemicals have been elusive [9]–[11].

Some examples of machine learning tools applied to design problems in communication systems include multiuser detection in code-division multiple-access (CDMA) systems [12]–[15], decoding of linear codes [16], design of new modulation and demodulation schemes [17], [18], detection and channel decoding [19]–[24], and estimating channel model parameters [25], [26]. A recent survey of machine learning techniques applied to communication systems can be found in [27]. The approach taken in most of these previous works was to use machine learning to improve one component of the communication system based on the knowledge of the underlying channel models.

Our approach is different from prior works since we assume that the mathematical models for the communication channel are completely unknown. This is motivated by the recent success in using deep neural networks (NNs) for end-to-end system design in applications such as image classification [28], [29], speech recognition [30]–[32], machine translation [33], [34], and bioinformatics [35]. For example, Figure 1 highlights some of the similarities between speech recognition, where deep NNs have been very successful at improving the detector's performance, and wireless as well as molecular communication systems. As indicated in the figure, for speech processing, the transmitter is the speaker, the transmission symbols are words, and the carrier signal is acoustic waves. At the receiver the goal of the detection algorithm is to

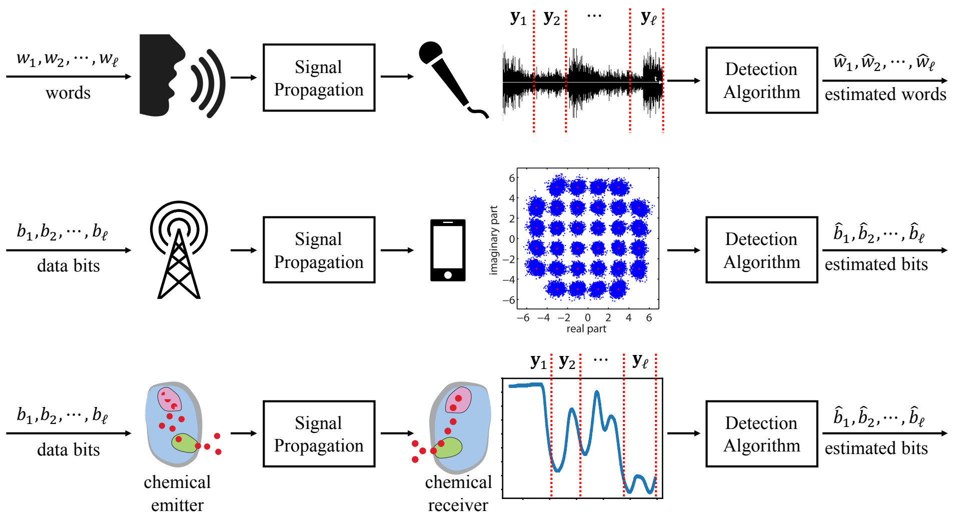

$w_1, w_2, \cdots, w_\ell$
words
Signal Propagation
$\mathbf{y}_1$ $\mathbf{y}_2$ $\cdots$ $\mathbf{y}_\ell$
Detection Algorithm
$\hat{w}_1, \hat{w}_2, \cdots, \hat{w}_\ell$
estimated words
$b_1, b_2, \cdots, b_\ell$
data bits
imaginary part
real part
$\hat{b}_1, \hat{b}_2, \cdots, \hat{b}_\ell$
estimated bits
chemical emitter
chemical receiver

Fig. 1: Similarities between speech recognition and digital communication systems.

recover the sequence of transmitted words from the acoustic signals that are received by the microphone. Similarly, in communication systems, such as wireless or chemical communications, the transmitted symbols are bits and the carrier signals are EM waves or chemical signals. At the receiver the goal of the detection algorithm is to detect the transmitted bits from the received signal. One important differences between communication systems and a speech recognition is the size of transmission symbol set, which is significantly larger for speech.

Motivated by this similarity, in this work we investigate how techniques from deep learning can be used to train a detection algorithm from samples of transmitted and received signals. We demonstrate that, using known NN architectures such as a recurrent neural network (RNN), it is possible to train a detector without any knowledge of the underlying system model. In this approach, the receiver goes through a training phase where a NN detector is trained using known transmission signals. We also propose a real-time NN sequence detector, which we call the *sliding bidirectional RNN (SBRNN) detector*, that detects the symbols corresponding to a data stream as they arrive at the destination. We demonstrate that if the SBRNN detector or the other NN detectors considered in this work are trained using a diverse dataset that contains transmission sequences in different channel conditions, the detectors will be robust to changing channel conditions, eliminating the need for instantaneous CSI estimation.

At first glance, the training phase in this approach may seem like an extra overhead. However, if the underlying channel models are known, then the models could be used off-line to generate training data under a diverse set of channel conditions. We demonstrate that using this approach, it is possible to train our SBRNN algorithm such that it would not require any instantaneous CSI. Another important benefit of using NN detectors in general is that they return likelihoods for each symbol. These likelihoods can be fed directly from the detector into a soft decoding algorithm such as the belief propagation algorithm without requiring a dedicated module to convert the detected symbols into likelihoods.

To evaluate the performance of NN detectors, we first use the Poisson channel model, a common model for optical channels and molecular communication channels [36]–[41]. We use this model to compare the performance of the NN detection to the Viterbi detector (VD). We show that for channels with long memories the SBRNN detection algorithm is computationally more efficient than the VD. Moreover, the VD requires CSI estimation, and its performance can degrade if this estimate is not accurate, while the SBRNN detector can perform detection without the CSI, even in a channel with changing conditions. We show that the bit error rate (BER) performance of the proposed SBRNN is better than the VD with CSI estimation error and it outperforms other well-known NN detector such as the RNN detector. As another performance measure, we use the experimental data collected by the molecular communication platform presented in [42]. The mathematical models underlying this experimental platform are currently unknown. We demonstrate that the proposed SBRNN algorithm can be used to train a sequence detector directly from limited measurement data. We also demonstrate that this approach perform significantly better than the detector used in previous experimental demonstrations [43], [44], as well as other NN detectors.

The rest of the paper is organized as follows. In Section II we present the problem statement. Then, in Section III, detection algorithms based on NN are introduced including the newly proposed SBRNN algorithm. The Poisson channel model and the VD are introduced in Section IV. The performance of the NN detection algorithms are evaluated using this channel model and are compared against the VD in Section V. In Section VI, the performance of NN detection algorithms are evaluated using a small data set that is collected via an experimental platform. Concluding remarks are provided in Section VII.

II. Problem Statement

In a digital communication system data is converted into a sequence of symbols for transmission over the channel. This

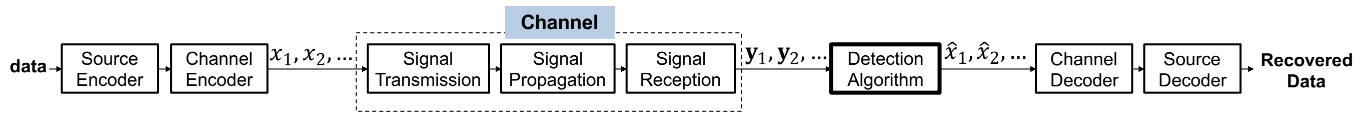


Fig. 2: Block diagram for digital communication systems.

process is typically carried out in two steps: in the first step, source coding is used to compress or represent the data using symbols or bits; in the second step, channel coding is used to introduce extra redundant symbols to mitigate the errors that may be introduced as part of the transmission and reception of the data [45]. Let $\mathcal{S} = \{s_1, s_2, \cdots, s_m\}$ be the finite set of symbols that could be sent by the transmitter, and $x_k \in \mathcal{S}$ be the k^{th} symbol that is transmitted. The channel coding can be designed such that the individual symbols in a long sequence are drawn according to the probability mass function (PMF) $P_X(x)$.

The signal that is observed at the destination is noisy and corrupted due to the perturbations introduced as part of transmission, propagation, and reception processes. We refer to these three processes collectively as the *communication channel* or simply the *channel*. Let the random vector $\mathbf{y}_k$ of length ℓ be the observed signal at the destination during the k^{th} transmission. Note that the observed signal $\mathbf{y}_k$ is typically a vector while the transmitted symbol x_k is typically a scalar. A *detection algorithm* is then used to estimate the transmitted symbols from the observed signal at the receiver. Let $\hat{x}_k$ be the symbol that is estimated for the k^{th} transmitted symbol x_k. After detection, the estimated symbols are passed to a channel decoder to correct some of the errors in detection, and then to a source decoder to recover the data. All the components of a communication system, shown in Figure 2, are designed to ensure reliable data transfer.

Typically, to design these modules, mathematical channel models are required, which describe the relationship between the transmitted symbols and the observed signal through

$$P_{\text{model}}(\mathbf{y}_1, \mathbf{y}_2, \cdots \mid x_1, x_2, \cdots; \mathbf{\Theta}), \tag{1}$$

where $\mathbf{\Theta}$ are the model parameters. Some of these parameters can be static (constants that do not change with channel conditions) and some of them can dynamically change with channel conditions over time. In this work, model parameters are considered to be the parameters that change with time. Hence, we use the terms model parameter and instantaneous CSI interchangeably. Using this model, the detection can be performed through symbol-by-symbol detection, where $\hat{x}_k$ is estimated from $\mathbf{y}_k$, or using sequence detection where the sequence $\hat{x}_k, \hat{x}_{k-1}, \cdots, \hat{x}_1$ is estimated from the sequence $\mathbf{y}_k, \mathbf{y}_{k-1}, \cdots, \mathbf{y}_1$[1]. As an example, for a simple channel with no intersymbol interference (ISI), given by the channel model $P_{\text{model}}(\mathbf{y}_k \mid x_k; \mathbf{\Theta})$, and a known PMF for the transmission symbols $P_X(x)$, a maximum a posteriori estimation (MAP) algorithm can be devised as

$$\hat{x}_k = \arg\max_{x \in \mathcal{S}} P_{\text{model}}(\mathbf{y}_k \mid x; \mathbf{\Theta}) P_X(x). \tag{2}$$

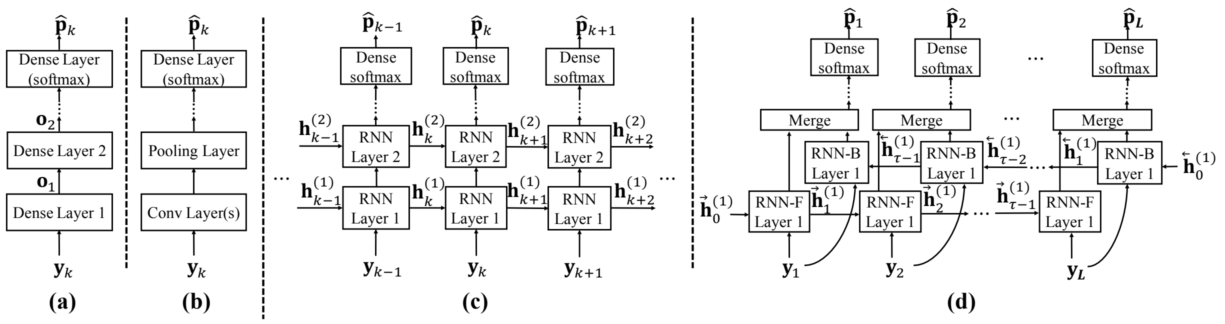
Therefore for detection, both the model and the parameters of the model $\mathbf{\Theta}$, which may change with time, are required. For this reason, many detection algorithms periodically estimate the model parameters (i.e., the CSI) by transmitting known symbols and then using the observed signals at the receiver for CSI estimation [46]. This extra overhead leads to a decrease in the data rate. One way to avoid CSI estimation is by using blind detectors. These detectors typically assume a particular probability distribution over $\mathbf{\Theta}$, and perform the detection without estimating the instantaneous CSI at the cost of higher probability of error. However, estimating the joint distribution over all model parameters $\mathbf{\Theta}$ can also be difficult, requiring a large amount of measurement data under various channel conditions. One of the problems we consider in this work is whether NN detectors can learn this distribution during training, or learn to simultaneously estimate the CSI and detect the symbols. This approach results in a robust detection algorithm that performs well under different and changing channel conditions without any knowledge of the channel models or their parameters.

When the underlying channel models do not lend themselves to computationally efficient detection algorithms, or are partly or completely unknown, the best approach to designing detection algorithms is unclear. For example, in communication channels with memory, the complexity of the optimal VD increases exponentially with memory length, and quickly becomes infeasible for systems with long memory. Note that the VD also relies on the knowledge of the channel model in terms of its input-output transition probability. As another example, tractable channel models for chemical communication channels with multiple reactive chemicals are unknown [9]–[11]. We propose that in these scenarios, a data driven approach using deep learning is an effective way to train detectors to determine the transmitted symbols directly using known transmission sequences.

III. Detection Using Deep Learning

Estimating the transmitted symbol from the received signals can be performed using NN architectures through supervised learning. This is achieved in two phases. First, a training dataset is used to train the NN offline. Once the network is trained, it can be deployed and used for detection. Note that the training phase is performed once offline, and therefore, it is not part of the detection process after deployment. We start this section by describing the training process.

[1]Note that the sequence of symbols $\hat{x}_k, \hat{x}_{k-1}, \cdots, \hat{x}_1$ can also be estimated from $\mathbf{y}_{k+\ell}, \mathbf{y}_{k+\ell-1}, \cdots, \mathbf{y}_1$ for some integer ℓ. However, to keep the notation simpler, without loss of generality we assume $\ell = 0$.



$\hat{\mathbf{p}}_k$
Dense Layer (softmax)
$\mathbf{o}_2$
Dense Layer 2
$\mathbf{o}_1$
Dense Layer 1
$\mathbf{y}_k$
(a)
Pooling Layer
Conv Layer(s)
(b)
Dense softmax
RNN Layer 2
RNN Layer 1
$\mathbf{h}_{k-1}^{(2)}$ $\mathbf{h}_{k}^{(2)}$ $\mathbf{h}_{k+1}^{(2)}$ $\mathbf{h}_{k+2}^{(2)}$
$\mathbf{h}_{k-1}^{(1)}$ $\mathbf{h}_{k}^{(1)}$ $\mathbf{h}_{k+1}^{(1)}$ $\mathbf{h}_{k+2}^{(1)}$
$\mathbf{y}_{k-1}$ $\mathbf{y}_k$ $\mathbf{y}_{k+1}$
(c)
$\hat{\mathbf{p}}_1$ $\hat{\mathbf{p}}_2$ $\hat{\mathbf{p}}_L$
Merge
RNN-B Layer 1
RNN-F Layer 1
$\vec{\mathbf{h}}_0^{(1)}$ $\vec{\mathbf{h}}_1^{(1)}$ $\vec{\mathbf{h}}_2^{(1)}$ $\vec{\mathbf{h}}_{\tau-1}^{(1)}$
$\overleftarrow{\mathbf{h}}_{\tau-1}^{(1)}$ $\overleftarrow{\mathbf{h}}_{\tau-2}^{(1)}$ $\overleftarrow{\mathbf{h}}_1^{(1)}$ $\overleftarrow{\mathbf{h}}_0^{(1)}$
$\mathbf{y}_1$ $\mathbf{y}_2$ $\mathbf{y}_L$
(d)
Symbol-by-Symbol Detec.
Sequence Detection

Fig. 3: Different neural network architectures for detection.

A. Training the Detector

Let $m = |\mathcal{S}|$ be the cardinality of the symbol set, and let $\mathbf{p}_k$ be the one-of-m representation of the symbol transmitted during the k^{th} transmission, given by

$$\mathbf{p}_k = \left[\mathbb{1}(x_k = s_1), \mathbb{1}(x_k = s_2), \cdots, \mathbb{1}(x_k = s_m)\right]^\intercal, \quad (3)$$

where $\mathbb{1}(.)$ is the indicator function. Therefore, the element corresponding to the symbol that is transmitted is 1, and all other elements of $\mathbf{p}_k$ are 0. Note that this is also the PMF of the transmitted symbol during the k^{th} transmission where, at the transmitter, with probability 1, one of the m symbols is transmitted. Also note that the length of the vector $\mathbf{p}_k$ is m, which may be different from the length of the vector of the observation signal $\mathbf{y}_k$ at the destination.

The detection algorithm goes through two phases. In the first phase, known sequences of symbols form $\mathcal{S}$ are transmitted repeatedly and received by the system to create a set of training data. The training data can be generated by selecting the transmitted symbols randomly according to a PMF, and generating the corresponding received signal using mathematical models, simulations, experimental measurements, or field measurements. Let $\mathbf{P}_K = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_K]$ be a sequence of K consecutively transmitted symbols (in the one-of-m encoded representation), and $\mathbf{Y}_K = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_K]$ the corresponding sequence of observed signals at the destination. Then, the training dataset is represented by

$$\{(\mathbf{P}_{K_1}^{(1)}, \mathbf{Y}_{K_1}^{(1)}), (\mathbf{P}_{K_2}^{(2)}, \mathbf{Y}_{K_2}^{(2)}), \cdots, (\mathbf{P}_{K_n}^{(n)}, \mathbf{Y}_{K_n}^{(n)})\}, \quad (4)$$

which consists of n training samples, where the i^{th} sample has K_i consecutive transmissions.

This dataset is then used to train a deep NN classifier that maps the received signal $\mathbf{y}_k$ to one of the transmission symbols in $\mathcal{S}$. The input to the NN can be the raw observed signals $\mathbf{y}_k$, or a set of features $\mathbf{r}_k$ extracted from the received signals. The NN output are the vectors $\hat{\mathbf{p}}_k = \text{NN}(\mathbf{y}_k; \mathcal{W})$, where $\mathcal{W}$ are the parameters of the NN. Using the above interpretation of $\mathbf{p}_k$ as a probability vector, $\hat{\mathbf{p}}_k$ are the estimations of the probability of x_k given the observations and the parameters of the NN. Note that this output is also useful for soft decision channel decoders (i.e., decoders where the decoder inputs are PMFs), which are typically the next module after detection as shown in Figure 2. If channel coding is not used, the symbol is estimated using $\hat{x}_k = \arg\max_{x_k \in \mathcal{S}} \hat{\mathbf{p}}_k$.

During the training, known transmission sequences of symbols are used to find the optimal set of parameters for the NN $\mathcal{W}^*$ such that

$$\mathcal{W}^* = \underset{\mathcal{W}}{\arg\min}\, \mathscr{L}(\mathbf{p}_k, \hat{\mathbf{p}}_k), \quad (5)$$

where $\mathscr{L}$ is the loss function. This optimization algorithm is typically solved using the training data, variants of stochastic gradient decent, and back propagation [7]. Since the output of the NN is a PMF, the cross-entropy loss function can be used for this optimization [7]:

$$\mathscr{L}_{\text{cross}} = H(\mathbf{p}_k, \hat{\mathbf{p}}_k) = H(\mathbf{p}_k) + D_{\text{KL}}\left(\mathbf{p}_k \parallel \hat{\mathbf{p}}_k\right), \quad (6)$$

where $H(\mathbf{p}_k, \hat{\mathbf{p}}_k)$ is the cross entropy between the correct PMF and the estimated PMF, and $D_{\text{KL}}(. \parallel .)$ is the Kullback-Leibler divergence [47]. Note that minimizing the loss is equivalent to minimizing the cross-entropy or the Kullback-Leibler divergence distance between the true PMF and the one estimated based on the NN. It is also equivalent to maximizing the log-likelihoods. Therefore, during the training, known transmission data are used to train a detector that *maximizes log-likelihoods*. Using Bayes' theorem, it is easy to show that minimizing the loss is equivalent to maximizing (2). We now discuss how several well-known NN architectures can be used for symbol-by-symbol detection and for sequence detection.

B. Symbol-by-Symbol Detectors

The most basic NN architecture that can be employed for detection uses several fully connected NN layers followed by a final softmax layer [6], [7]. The input to the first layer is the observed signal $\mathbf{y}_k$ or the feature vector $\mathbf{r}_k$, which is selectively extracted from the observed signal through preprocessing. The output of the final layer is of length m (i.e., the cardinality the symbol set), and the activation function for the final layer is the softmax activation. This ensures that the output of the layer $\hat{\mathbf{p}}_k$ is a PMF. Figure 3(a) shows the structure of this NN.

A more sophisticated class of NNs that is used in processing complex signals such as images is a convolution neural network (CNN) [6], [48], [49]. Essentially, the CNN is a set of filters that are trained to extract the most relevant features for detection from the received signal. The final layer in the CNN detector is a dense layer with output of length m, and a softmax activation function. This results in an estimated $\hat{\mathbf{p}}_k$

from the set of features that are extracted by the convolutional layers in the CNN. Figure 3(b) shows the structure of this NN.

For symbol-by-symbol detection the estimated PMF $\hat{\mathbf{p}}_k$ is given by

$$\hat{\mathbf{x}}_k = [P_{\text{NN}}(x_k = s_1|\mathbf{y}_k), P_{\text{NN}}(x_k = s_2|\mathbf{y}_k), \cdots, P_{\text{NN}}(x_k = s_m|\mathbf{y}_k)]^\intercal, \tag{7}$$

where P_{NN} is the probability of estimating each symbol based on the NN model used. The better the structure of the NN at capturing the physical channel characteristics based on P_{model} in (1), the better this estimate and the results.

C. Sequence Detectors

The symbol-by-symbol detector cannot take into account the effects of ISI between symbols. In this case, sequence detection can be performed using recurrent neural networks (RNN) [6], [7], which are well established for sequence estimation in different problems such as neural machine translation [33], speech recognition [30], or bioinformatics [35]. The estimated $\hat{\mathbf{p}}_k$ in this case is given by

$$\hat{\mathbf{p}}_k = \begin{bmatrix} P_{\text{RNN}}(x_k = s_1|\mathbf{y}_k, \mathbf{y}_{k-1}, \cdots, \mathbf{y}_1) \\ P_{\text{RNN}}(x_k = s_2|\mathbf{y}_k, \mathbf{y}_{k-1}, \cdots, \mathbf{y}_1) \\ \vdots \\ P_{\text{RNN}}(x_k = s_m|\mathbf{y}_k, \mathbf{y}_{k-1}, \cdots, \mathbf{y}_1) \end{bmatrix}, \tag{8}$$

where P_{RNN} is the probability of estimating each symbol based on the NN model used. In this work, we use long short-term memory (LSTM) networks [50], which have been extensively used in many applications.

Figure 3(c) shows the RNN structure. One of the main benefits of this detector is that after training, similar to a symbol-by-symbol detector, it can perform detection on any data stream as it arrives at the receiver. This is because the observations from previous symbols are summarized as the state of the RNN, which is represented by the vector $\mathbf{h}_k$. Note that the observed signal during the j^{th} transmission slot, $\mathbf{y}_j$ where $j > k$, may carry information about the k^{th} symbol x_k due to delays in signal arrival which results in ISI. However, since RNNs are feed-forward only, during the estimation of $\hat{\mathbf{p}}_k$, the observation signal $\mathbf{y}_j$ is not considered.

One way to overcome this limitation is by using bidirectional RNNs (BRNNs), where the sequence of received signals are once fed in the forward direction into one RNN cell and once fed in backwards into another RNN cell [51]. The two outputs are then concatenated and may be passed to more bidirectional layers. Figure 3(d) shows the BRNN structure. For a sequence of length L, the estimated $\hat{\mathbf{p}}_k$ for BRNN is given by

$$\hat{\mathbf{p}}_k = \begin{bmatrix} P_{\text{BRNN}}(x_k = s_1|\mathbf{y}_L, \mathbf{y}_{L-1}, \cdots, \mathbf{y}_1) \\ P_{\text{BRNN}}(x_k = s_2|\mathbf{y}_L, \mathbf{y}_{L-1}, \cdots, \mathbf{y}_1) \\ \vdots \\ P_{\text{BRNN}}(x_k = s_m|\mathbf{y}_L, \mathbf{y}_{L-1}, \cdots, \mathbf{y}_1) \end{bmatrix}, \tag{9}$$

where $k \leq L$. In this work we use the bidirectional LSTM (BLSTM) networks [52].

The BRNN architecture ensures that in the estimation of a symbol, future signal observations are taken into account, thereby overcoming the limitations of RNNs. The main trade-off is that as signals from a data stream arrive at the destination, the block length L increases, and the whole block needs to be re-estimated again for each new data symbol that is received. Therefore, this quickly becomes infeasible for long data streams. In the next section we present a new technique to solve this issue.

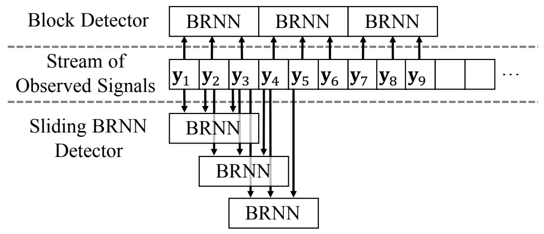

Block Detector
BRNN
BRNN
BRNN
Stream of Observed Signals
$\mathbf{y}_1$ $\mathbf{y}_2$ $\mathbf{y}_3$ $\mathbf{y}_4$ $\mathbf{y}_5$ $\mathbf{y}_6$ $\mathbf{y}_7$ $\mathbf{y}_8$ $\mathbf{y}_9$ ...
Sliding BRNN Detector
BRNN
BRNN
BRNN

Fig. 4: The sliding BRNN detector.

D. Sliding BRNN Detector

Since the data stream that arrives at the receiver can have any arbitrary length, it is not desirable to detect the whole sequence for each new symbol that arrives, as the sequence length could grow arbitrarily large. Therefore, we fix the maximum length of the BRNN. Ideally, the length must be at least the same size as the memory length of the channel. However, if this is not known in advance, the BRNN length can be treated as a hyperparameter to be tuned during training. Let L be the maximum length of the BRNN. Then during training, blocks of $\ell \leq L$ consecutive transmissions are used for training. Note that sequences of different length could be used during training as long as all sequence lengths are smaller than or equal to L. After training, the simplest scheme would be to detect the stream of incoming data in fixed blocks of length $\ell \leq L$ as shown in the top portion of Figure 4. The main drawback here is that the symbols at the end of each block may affect the symbols in the next block, and this relation is not captured in this scheme. Another issue is that ℓ consecutive symbols must be received before detection can be performed. The top portion of Figure 4, shows this scheme for $\ell = 3$.

To overcome these limitations, inspired by some of the techniques used in speech recognition [53], we propose a dynamic programing scheme we call the *sliding BRNN (SBRNN) detector*. In this scheme the first $\ell \leq L$ symbols are detected using the BRNN. Then as each new symbol arrives at the destination, the position of the BRNN slides ahead by one symbol. Let the set $\mathcal{J}_k = \{j \mid j \leq k \wedge j + L > k\}$ be the set of all valid starting positions for a BRNN detector of length L, such that the detector overlaps with the k^{th} symbol. For example, if $L = 3$ and $k = 4$, then $j = 1$ is not in the set $\mathcal{J}_k$ since the BRNN detector overlaps with symbol positions 1, 2, and 3, and not the symbol position 4. Let $\hat{\mathbf{p}}_k^{(j)}$ be the estimated PMF for the k^{th} symbol, when the start of the sliding BRNN is on $j \in \mathcal{J}_k$. The final PMF corresponding to the k^{th} symbol is given by the weighted sum of the estimated PMFs for each of the relevant windows:

$$\hat{\mathbf{p}}_k = \frac{1}{|\mathcal{J}_k|} \sum_{j \in \mathcal{J}_k} \hat{\mathbf{p}}_k^{(j)}. \tag{10}$$

One of the main benefits of this approach is that, after the first L symbols are received and detected, as the signal corresponding to a new symbol arrives at the destination, the detector immediately estimates that symbol. The detector also updates its estimate for the previous $L-1$ symbols dynamically. Therefore, this algorithm is similar to a dynamic programming algorithm.

The bottom portion of Figure 4 illustrates the sliding BRNN detector. In this example, after the first 3 symbols arrive, the PMF for the first three symbols, $i \in \{1,2,3\}$, is given by $\hat{\mathbf{p}}_i = \hat{\mathbf{p}}_i^{(1)}$. When the 4th symbol arrives, the estimate of the first symbol is unchanged, but for $i \in \{2,3\}$, the second and third symbol estimates are updated as $\hat{\mathbf{x}}_i = \frac{1}{2}(\hat{\mathbf{x}}_i^{(1)} + \hat{\mathbf{x}}_i^{(2)})$, and the 4th symbol is estimated by $\hat{\mathbf{p}}_4 = \hat{\mathbf{p}}_4^{(2)}$. Note that although in this paper we assume that the weights of all $\hat{\mathbf{p}}_k^{(j)}$ are the same (i.e., $\frac{1}{|\mathcal{J}_k|}$), the algorithm can use different weights. Moreover, the complexity of the SBRNN increases linearly with the length of the BRNN window, and hence with the memory length.

To evaluate the performance of all these NN detectors, we use both the Poisson channel model (a common model for optical and chemical communication systems) as well as an experimental platform for chemical communication where the underlying model is unknown [42]. The sequel discusses more details of the Poisson model and experimental platform, and how they were used for performance analysis of our proposed techniques.

IV. The Poisson Channel Model

The Poisson channel has been used extensively to model different communication systems in optical and molecular communication [36]–[41]. In these systems, information is encoded in the intensity of the photons or particles released by the transmitter and decoded from the intensity of photons or particles observed at the receiver. In the rest of this section, we refer to the photons, molecules, or particles simply as *particles*. We now describe this channel, and a VD for the channel.

In our model it is assumed that the transmitter uses on-off-keying (OOK) modulation, where the transmission symbol set is $\mathcal{S} = \{0,1\}$, and the transmitter either transmits a pulse with a fixed intensity to represent the 1-bit or no pulse to represent the 0-bit. Note that OOK modulation has been considered in many previous works on optical and molecular communication and has been shown to be the optimal input distribution for a large class of Poisson channels [54]–[56].

Let τ be the symbol interval, and $x_k \in \mathcal{S}$ the symbol corresponding to the k^{th} transmission. We assume that the transmitter can measure the number of particles that arrive at a sampling rate of ω samples per second. Then the number of samples in a given symbol duration is given by $a = \omega\tau$, where we assume that a is an integer. Let $\lambda(t)$ be the system response to a transmission of the pulse corresponding to the 1-bit. For optical channels, the system response is proportional

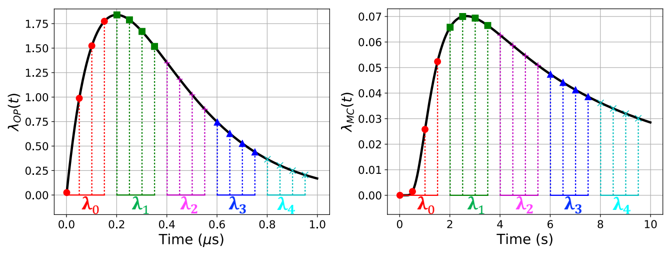


Fig. 5: A sample system response for optical and molecular channels. Left: Optical channel with $\lambda(t)$ for $N=1$, $\kappa_{\text{OP}}=1$, $\alpha=2$, $\beta=0.2$, $\tau=0.2$ μs, and $\omega=20$ MS/s. At $\tau=0.2$ μs, much of the intensity from the current transmission will arrive during future symbol intervals. Right: Molecular channel with $\kappa_{\text{MO}}=1$, $c=8$, $\mu=40$, $\tau=2$ s, and $\omega=2$ S/s. Molecular channel response has a loner tail than optical channel.

to the Gamma distribution, and given by [57]–[59]:

$$\lambda_{\text{OP}}(t) = \begin{cases} \kappa_{\text{OP}} \frac{\beta^{-\alpha} t^{\alpha-1}}{\Gamma(\alpha)} \exp(-t/\beta) & t > 0 \\ 0 & t \le 0 \end{cases}, \quad (11)$$

where κ_{OP} is the proportionality constant, and α and β are parameters of the channel, which can change over time. For molecular channels, the system response is proportional to the inverse Gaussian distribution [39], [40], [60], [61] given by:

$$\lambda_{\text{MO}}(t) = \begin{cases} \kappa_{\text{MO}} \sqrt{\frac{c}{2\pi t^3}} \exp\left[-\frac{c(t-\mu)^2}{2\mu^2 t}\right] & t > 0 \\ 0 & t \le 0 \end{cases}, \quad (12)$$

where κ_{MO} is the proportionality constant, and c and μ are parameters of the channel, which can change over time.

Since the receiver samples the data at a rate of ω, for $k \in \mathbb{N}$ and $j \in \{1,2,\cdots,a\}$, let

$$\boldsymbol{\lambda}_k[j] \triangleq \lambda\left(\frac{j+ka}{\omega}\right) \quad (13)$$

be the average intensity observed during the j^{th} sample of the k^{th} symbol in response to the transmission pulse corresponding to the 1-bit. Figure 5 shows the system response for both optical and molecular channels. Although for optical channels the symbol duration is many orders of magnitude smaller than molecular channel, the system responses are very similar in shape. Some notable differences are a faster rise time for the optical channel, and a longer tail for the molecular channel.

The system responses are used to formulate the Poisson channel model. Particularly, the intensity that is observed during the j^{th} sample of the k^{th} symbol is distributed according to

$$\mathbf{y}_k[j] \sim \mathscr{P}\left(\sum_{i=0}^{k} x_{k-i}\boldsymbol{\lambda}_i[j] + \eta\right), \quad (14)$$

where η is the mean of an independent additive Poisson noise due to background interference and/or the receiver noise. Using this model, the signal that is observed by the receiver, for any sequence of bit transmissions, can be generated as illustrated in Figure 6. This signal has a similar structure to the signal observed using the experimental platform in [43, see

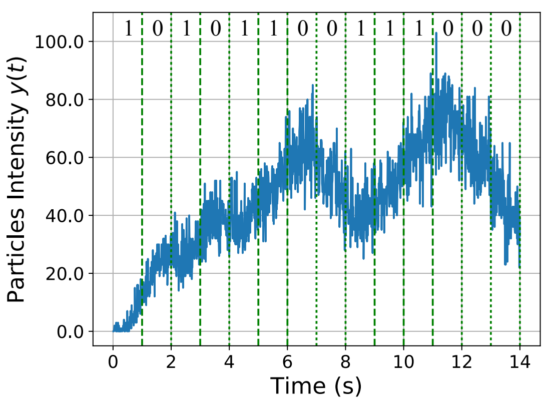


Fig. 6: The observed signal for the transmission of the bit sequence 10101100111000 for $\kappa_{\text{MO}} = 100$, $c = 8$, $\mu = 40$, $\tau = 1$, $\omega = 100$ Hz, and $\eta = 1$.

Figure 13], although this analytically-modeled signal exhibits more noise.

The model parameters (i.e., the CSI) for the Poisson channel model are $\mathbf{\Theta}_{\text{OP}} = [\alpha, \beta, \eta]$ and $\mathbf{\Theta}_{\text{MO}} = [c, \mu, \eta]$, respectively for optical and molecular channels. In this work, we assume that the sampling rate ω, and the proportionality constants κ_{OP} and κ_{MO} are fixed and are not part of the model parameters. Note that α and β can change over time due to atmospheric turbulence or mobility. Similarly, c and μ are functions of the distance between the transmitter and the receiver, flow velocity, and the Diffusion coefficient, which may change over time, e.g., due to variations in temperature and pressure [5]. The background noise η may also change with time. Note that although the symbol interval τ may be changed to increase or decrease the data rate, both the transmitter and receiver must agree on the value of τ. Thus, we assume that the value of τ is always known at the receiver, and therefore, it is not part of the CSI. In the next subsection, we present the optimal VD, assuming that the receiver knows all the model parameters $\mathbf{\Theta}_{\text{OP}}$ and $\mathbf{\Theta}_{\text{MO}}$ perfectly.

A. The Viterbi Detector

The VD assumes a certain memory length M where the current observed signal is affected only by the past M transmitted symbols. In this case, (14) becomes

$$\mathbf{y}_k[j] \sim \mathscr{P}\left(x_k \boldsymbol{\lambda}_0[j] + \sum_{l=1}^{M} x_{k-l}\boldsymbol{\lambda}_l[j] + \eta\right). \tag{15}$$

Since the marginal distribution of the j^{th} sample of the k^{th} symbol is Poisson distributed according to (15), given the model parameters $\mathbf{\Theta}_{\text{pois}}$, we have

$$P(\mathbf{y}_k \mid x_{k-M}, x_{k-M+1}, \cdots, x_k, \mathbf{\Theta}_{\text{pois}}) = \prod_{j=1}^{a} P(\mathbf{y}_k[j] \mid x_{k-M}, x_{k-M+1}, \cdots, x_k, \mathbf{\Theta}_{\text{pois}}). \tag{16}$$

This is because, given the model parameters as well as the current symbol and the previous M symbols, the samples within the current bit interval are generated independently and distributed according to (15). Note that (16) holds only if the memory length M is known perfectly. If the estimate of M is inaccurate, then (16) is also inaccurate.

Let $\mathcal{V} = \{v_0, v_1, \cdots, v_{2^M-1}\}$ be the set of states in the trellis of the VD, where the state v_u corresponds to the previous M transmitted bits $[x_{-M}, x_{-M+1}, \cdots, x_{-1}]$ forming the binary representation of u. Let $\hat{x}_k$, $1 \leq k \leq K$ be the information bits to be estimated. Let $V_{k,u}$ be the state corresponding to the k^{th} symbol interval, where u is the binary representation of $[\hat{x}_{k-M}, \hat{x}_{k-M+1}, \cdots, \hat{x}_{k-1}]$. Let $\mathcal{L}(V_{k,u})$ denote the log-likelihood of the state $V_{k,u}$. For a state $V_{k+1,u} = [\hat{x}_{k-M+1}, \hat{x}_{k-M+2}, \cdots, \hat{x}_k]$, there are two states in the set $\{V_{k,i}\}_{i=0}^{2^M-1}$ that can transition to $V_{k+1,u}$:

$$u_0 = \lfloor \tfrac{u}{2} \rfloor, \tag{17}$$

$$u_1 = \lfloor \tfrac{u}{2} \rfloor + 2^{M-1}, \tag{18}$$

where $\lfloor . \rfloor$ is the floor function. Let the binary vector $\mathbf{b}_{u_0} = [0, \hat{x}_{k-M+1}, \hat{x}_{k-M+2}, \cdots, \hat{x}_{k-1}]$ be the binary representation of u_0 and similarly $\mathbf{b}_{u_1}$ the binary representation of u_1. The log-likelihoods of each state in the next symbol slot are updated according to

$$\begin{aligned}\mathcal{L}(V_{k+1,u}) = \max[&\mathcal{L}(V_{k,u_0}) + \mathcal{L}(V_{k,u_0}, V_{k+1,u}),\\ &\mathcal{L}(V_{k,u_1}) + \mathcal{L}(V_{k,u_1}, V_{k+1,u})],\end{aligned} \tag{19}$$

where $\mathcal{L}(V_{k,u_i}, V_{k+1,u})$, $i \in \{0, 1\}$, is the log-likelihood increment of transitioning from state V_{k,u_i} to $V_{k+1,u}$. Let

$$\boldsymbol{\Lambda}_{u_i,u}[j] = (u \mod 2)\boldsymbol{\lambda}_0[j] + \sum_{l=1}^{M} \mathbf{b}_{u_i}[M-l+1]\boldsymbol{\lambda}_l[j] + \eta. \tag{20}$$

Using the PMF of the Poisson distribution, (15), (16), and (20) we have

$$\mathcal{L}(V_{k,u_i}, V_{k+1,u}) = -\sum_{j=1}^{a} \boldsymbol{\Lambda}_{u_i,u}[j] + \sum_{j=1}^{a} \log(\boldsymbol{\Lambda}_{u_i,u}[j])\mathbf{y}_k[j], \tag{21}$$

where the extra term $-\sum_{j=1}^{a} \log(\mathbf{y}_k[j]!)$ is dropped since it will be the same for both transitions from u_0 and u_1. Using these transition probabilities and setting the $\mathcal{L}(V_{0,0}) = 0$ and $\mathcal{L}(V_{0,u}) = -\infty$, for $u \neq 0$, the most likely sequence $\hat{x}_k$, $1 \leq k \leq K$, can be estimated using the Viterbi algorithm [62]. When the memory length is long, it is not computationally feasible to consider all the states in the trellis as they grow exponentially with memory length. Therefore, in this work we implement the Viterbi beam search algorithm [63]. In this scheme, at each time slot, only the transition from the previous N states with the largest log-likelihoods are considered. When $N = 2^M$, the Viterbi beam search algorithm reduces to the traditional Viterbi algorithm.

We now evaluate the performance of NN detectors using the Poisson channel model.

V. Evaluation Based on Poisson Channel

In this section we evaluate the performance of the proposed SBRNN detector. First, we use the Poisson channel model to

TABLE I: Performance of the VD beam search as function of N. The optical channel results is obtained using $\mathbf{\Theta}_{\text{OP}} = [\beta = 0.2, \eta = 1]$ and $\tau = 0.025$ μs and the molecular channel results using $\mathbf{\Theta}_{\text{MO}} = [c = 8, \mu = 40, \eta = 100]$ and $\tau = 0.5$ s.

N	10	100	200	500	1000
Opti. VD 0.0% error	0.0466	0.03937	0.03972	0.03906	0.03972
Opti. VD 2.5% error	0.226	0.175	0.17561	0.15889	0.1509
Opti. VD 5.0% error	0.4036	0.385	0.38519	0.39538	0.36
Mole. VD 0.0% error	0.00466	0.00398	0.00464	0.00448	0.00432
Mole. VD 2.5% error	0.0066	0.0055	0.00524	0.0056	0.00582
Mole. VD 5.0% error	0.41792	0.34667	0.30424	0.29314	0.30588

compare the performance of the SBRNN to an RNN detector, the VD detector with perfect CSI estimates, and the VD with imperfect CSI estimates. We demonstrate that unlike the RNN detector, the SBRNN detector is robust to changing channel conditions and can outperform the VD with CSI estimation error. Second, we use the experimental platform developed in [42] to demonstrate that the SBRNN detector, which does not require any underlying analytical channel model, can be implemented in practice to perform real-time detection. In particular, we use the experimental data collected on the platform to train and implement the SBRNN as part of a chemical communication system for real-time text messaging. We show that SBRNN outperforms all the other NN detectors we consider in terms of BER performance.

For evaluating the performance of the SBRNN on the Poisson channel, we consider both the optical channel and the molecular channel. For the optical channel, we assume that the channel parameters are $\mathbf{\Theta}_{\text{OP}} = [\beta, \eta]$, and assume $\alpha = 2$ and $\kappa_{\text{OP}} = 10$. We use these values for α and κ_{OP} since they resulted in system responses that resembled the ones presented in [57]–[59]. For the molecular channel the model parameters are $\mathbf{\Theta}_{\text{MO}} = [c, \mu, \eta]$, and $\kappa_{\text{MO}} = 10^4$. The value of κ_{MO} was selected to resemble the system response in [43]. For the optical channel we use $\omega = 2$ GS/s and for the molecular channel we use $\omega = 100$ S/s.

For the VD algorithm we consider Viterbi with beam search, where only the top $N = 100$ states with the largest log-likelihoods are kept in the trellis during each time slot. We also consider two different scenarios for CSI estimation. In the first scenario we assume that the detector estimates the CSI perfectly, i.e., the values of the model parameters $\mathbf{\Theta}_{\text{OP}}$ and $\mathbf{\Theta}_{\text{MO}}$ are known perfectly at the receiver. In practice, it may not be possible to achieve perfect CSI estimation. In the second scenario we consider the VD with CSI estimation error. Let ζ be a parameter in $\mathbf{\Theta}_{\text{OP}}$ or $\mathbf{\Theta}_{\text{MO}}$. Then the estimate of this parameter is simulated by $\hat{\zeta} = \zeta + Z$, where Z is a zero-mean Gaussian noise with a standard divination that is 2.5% or 5% of ζ. In the rest of this section, we refer to these cases as the VD with 2.5% and 5% error, and the case with perfect CSI as the VD with 0% error. Table I shows the BER performance of the VD for different values of N. It can be seen that $N = 100$, which is used in the rest of this section, is sufficient to achieve good performance with the VD.

Both the RNN and the SBRNN detectors use LSTM cells [50]. For the SBRNN, the size of the output is 80. For the RNN, since the SBRNN uses two RNNs, one for the forward direction and one for the backward direction, the size of the output is 160. This ensures that the SBRNN detector and the RNN detector have roughly the same number of parameters. The number of layers used for both detectors in this section is 3. The input to the NNs are a set of normalized features $\mathbf{r}_k$ extracted from the received signal $\mathbf{y}_k$. The feature extraction algorithm is described in appendix. This feature extraction step normalizes the input, which assists the NNs to learn faster from the data [7].

To train the RNN and SBRNN detectors, transmitted bits are generated at random and the corresponding received signal is generated using the Poisson model in (14). In particular, the training data consists of many samples of sequences of 100 consecutively transmitted bits and the corresponding received signal. Since in this work we focus on uncoded communication, we assume the occurrence of both bits in the transmitted sequence are equiprobable. For each sequence, the CSI are selected at random. Particularly, for the optical channel, for each 100-bit sequence,

$$\beta \sim \mathscr{U}(\{0.15, 0.16, 0.17, \cdots, 0.3\}),$$
$$\eta \sim \mathscr{U}(\{1, 10, 20, 50, 100, 200, 500\}),$$
$$\tau \sim \mathscr{U}(\{0.025, 0.05, 0.075, 0.1\})(\text{all in } \mu\text{s}),$$

where $\mathscr{U}(\mathcal{A})$ indicates uniform distribution over the set $\mathcal{A}$. Similarly, for molecular channel,

$$c \sim \mathscr{U}(\{1, 2, \cdots, 30\}),$$
$$\mu \sim \mathscr{U}(\{5, 10, 15, \cdots, 65\}),$$
$$\eta \sim \mathscr{U}(\{1, 50, 100, 500, 1\text{k}, 5\text{k}, 10\text{k}, 20\text{k}, 30\text{k}, 40\text{k}, 50\text{k}\}),$$
$$\tau \sim \mathscr{U}(\{0.5, 1, 1.5, 2\})(\text{all in s}).$$

For the SBRNN training, each 100-bit sequence is randomly broken into subsequences of length $L \sim \mathscr{U}(\{2, 3, 4, \cdots, 50\})$. For all training, the Adam optimization algorithm [64] is used with the learning rate 10^{-3}, and the batch size is 500. We train on 500k sequences of 100 bits.

First, we evaluate the BER performance with respect to the memory length M used in the VD, and the sequence length L used in the SBRNN. For all the BER performance plots in this section, to calculate the BER, 1000 sequences of 100 random bits are used. Figure 7(a) shows the results for the optical (top plots) and the molecular (bottom plots) channels with the parameters described above. From the results it is clear that the performance of the VD relies heavily on estimating the memory length of the system correctly. If this estimate is inaccurate, the SBRNN algorithm outperforms the VD with perfect CSI. We also observe that the SBRNN achieves a better BER when there is a CSI estimation error of 2.5% or more. Note that the RNN detector does not have a parameter that depends on the memory length and has a significantly larger BER compared to the SBRNN. For the optical channel, the RNN detector outperforms the VD with 5% error in CSI estimation. Moreover, it can seen that the optical channel has a shorter memory length compared to the molecular channel.

Remark 1: When the VD has the perfect CSI, it can estimate the memory length correctly by using the system response. However, if there is CSI estimation error, the memory length may not be estimated correctly, and as can be seen in Figure 7(a), this can have degrading effects on the performance

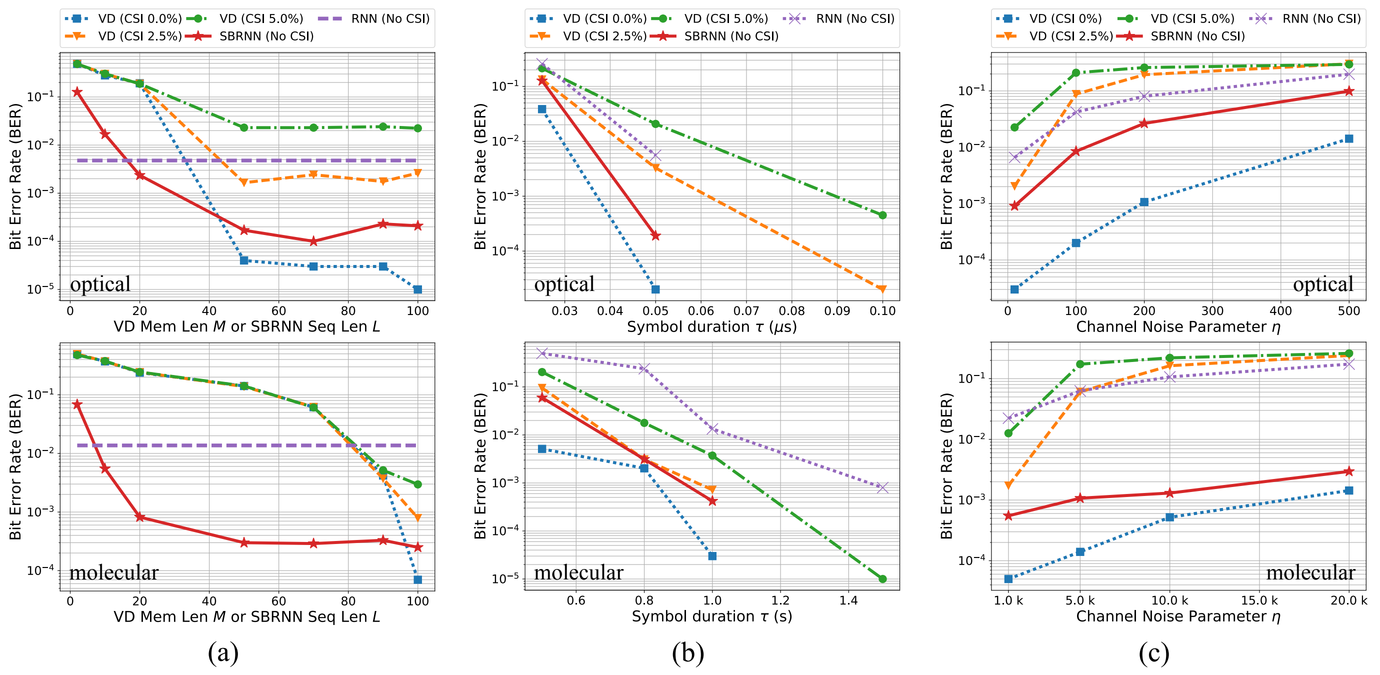


Fig. 7: The BER performance comparison of the SBRNN detector, the RNN detector, and the VD. The top plots present the optical channel and the bottom plots present the molecular channel. (a) The BER at various memory lengths M and SBRNN sequence lengths L. Top: $\mathbf{\Theta}_{\text{OP}} = [\beta = 0.2, \eta = 1]$ and $\tau = 0.05$ μs. Bottom: $\mathbf{\Theta}_{\text{MO}} = [c = 10, \mu = 40, \eta = 100]$ and $\tau = 1$ s. (b) The BER at various symbol durations for $L = 50$ and $M = 99$. The $\mathbf{\Theta}_{\text{OP}}$ (top) and $\mathbf{\Theta}_{\text{MO}}$ (bottom) are the same as (a). (c) The BER at various noise rates for $L = 50$ and $M = 99$. Except η, all the parameters are the same as those in (a).

of the VD. However, in the rest of this section, for all the other VD plots, we use the memory length of 99, i.e., the largest possible memory length in sequences of 100 bits. Although this does not capture the performance degradation that may result from the error in estimating the memory length, as we will show, the SBRNN still achieves a BER performance that is as good or better than the VD plots with CSI estimation error under various channel conditions.

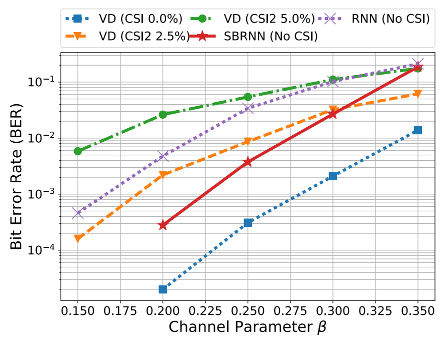
Next we evaluate the BER for different symbol durations in Figure 7(b). Again we observe that the SBRNN achieves a better BER when there is a CSI estimation error of 2.5% or more. The RNN detector outperforms the VD with 5% CSI estimation error for the optical channel, but does not perform well for the molecular channel. All detectors achieve zero-error in decoding the 1000 sequences of 100 random bits used to calculate the BER for the optical channel with $\tau = 0.1$ μs. Similarly, for the molecular channel at $\tau = 1.5$ s, all detectors except the RNN detector achieve zero error.

Figure 7(c) evaluates the BER performance at various noise rates. The SBRNN achieves a BER performance close to the VD with perfect CSI across a wide range of values. For larger values of η, i.e., low signal-to-noise ratio (SNR), both the RNN detector and the SBRNN detector outperform the VD with CSI estimation error.

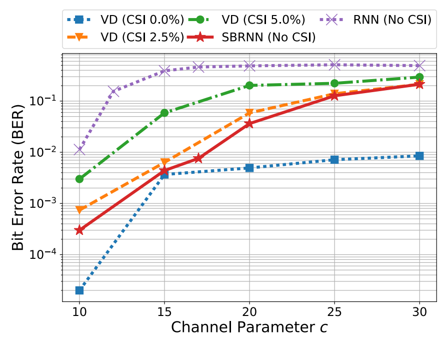
In Figure 8, we evaluate the performance of the detection algorithms with respect to the channel parameters that affect the system response. Note that in optical and molecular communication these parameters can change rapidly due to atmospheric turbulence, changes in temperature, or changes in the distance between the transmitter and the receiver. Therefore, estimating these parameters accurately can be challenging. Furthermore, since these parameters change the shape of the system responses they change the memory length as well. From the plot, it can be seen that the SBRNN performs as well or better than the VD with an estimation error of 2.5%. Moreover, for the optical channel, the RNN detector performs better than the VD with 5% estimation error.

We conclude this section by comparing the computational complexity of the SBRNN detector, the RNN detector and the VD. Let n be the length of the sequence to be decoded. Recall that L is the length of the sliding BRNN, M is the memory length of the channel, and N is the number of states with highest log-likelihood values among the 2^M states of the trellis that are kept at each time instance in the beam search Viterbi algorithm. Note that for the traditional Viterbi algorithm $N = 2^M$. The computational complexity of the SBRNN is given by $O(L(n - L + 1))$, while the computational complexity of the VD is given by $O(Nn)$. Therefore, for the traditional VD, the computational complexity grows exponentially with memory length M. However, this is not the case for the SBRNN detector. The computational complexity of the RNN detector is $O(n)$. Therefore, the RNN detector is the most efficient in terms of computational complexity, while the SBRNN detector and the beam search VD algorithm can have similar computational complexity. Finally, the traditional VD algorithm is impractical for the channels considered due to its exponential computational complexity in the memory length M.

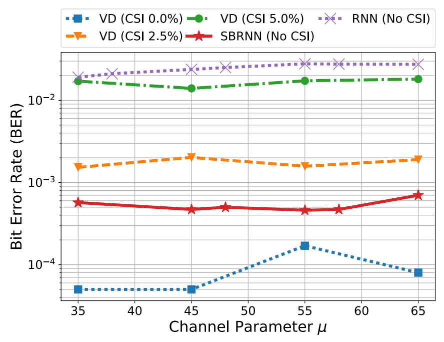
We now evaluate the NN detectors on a data generated from a molecular communication experimental platform presented



(a)



(b)



(c)

Fig. 8: The BER performance comparison of the SBRNN detector ($L = 50$), the RNN detector, and the VD ($M = 99$). (a) The BER at various β for the optical channel with $\eta = 1$ and $\tau = 0.05$ μs. (b) The BER at various c for the molecular channel with $\mu = 40$, $\eta = 1$, and $\tau = 1$ s. (c) The BER at various μ for the molecular channel with $c = 10$, $\eta = 1000$, and $\tau = 1$ s.

in [42]. We demonstrate that NN detectors can be implemented in practice without any channel models, and that they can be trained directly on a limited experimental or field measurements. We also demonstrate that they can be implemented for real-time text detection by building a real-time messaging service using the experimental platform in [42].

VI. Evaluation Based on Experimental Platform

We use the experimental platform in [42] to collect measurement data and create the dataset that is used for training and testing the detection algorithms. In the platform, time-slotted communication is employed where the transmitter modulates information on acid and base signals by injecting these chemicals into the channel during each symbol duration. The receiver then uses a pH probe for detection. A binary modulation scheme is used in the platform where the 0-bit is transmitted by pumping acid into the environment for 30 ms at the beginning of the symbol interval, and the 1-bit is represented by pumping base into the environment for 30 ms at the beginning of the symbol interval. The symbol interval consists of this 30 ms injection interval followed by a period of silence, which can also be considered as a guard band between symbols. In particular, four different silence durations (guard bands) of 220 ms, 304 ms, 350 ms, and 470 ms are used in this work to represent bit rates of 4, 3, 2.6, and 2 bps. This is similar to the OOK modulation used in the previous section for the Poisson channel model, except that chemicals of different types are released for both the 1-bit and the 0-bit.

The training and test data sets are generated as follows. For each symbol duration, random bit sequences of length 120 are transmitted 100 times, where each of the 100 transmissions are separated in time. Since we assume no channel coding is used, the bits are i.i.d. and equiprobable. This results in 12k bits per symbol duration that is used for training and testing. From the data, 84 transmissions per symbol duration (10,080 bits) are used for training and 16 transmissions are used for testing (1,920 bits). Therefore, the total number of training bits are 40,320, and the total number of bits used for testing is 7,680.

Although we expect from the physics of the chemical propagation and chemical reaction that the channel should have memory, since the channel model for this experimental platform is currently unknown, we implement both symbol-by-symbol and sequence detectors based on NNs. Note that due to the lack of a channel model, we cannot use the VD for comparison since it cannot be implemented without an underlying channel model. Instead, as a baseline detection algorithm, we use the slope detector that was used in previous work [42]–[44]. For all training of the NN detectors, the Adam optimization algorithm [64] is used with the learning rate 10^{-3}. Unless specified otherwise, the number of epochs used during training is 200 and the batch size is 10. All the hyperparameters are tuned using grid search.

We consider two symbol-by-symbol NN detectors. The first detector uses three fully connected layers with 80 hidden nodes and a final softmax layer for detection. Each fully connected layer uses the rectified linear unit (ReLU) activation function. The input to the network is a set of features extracted from the received signal, which are chosen based on performance and the characteristics of the physical channel as explained in appendix. We refer to this network as *Base-Net*. A second symbol-by-symbol detector uses 1-dimensional CNNs. The best network architecture that we found has the following layers. 1) 16 filters of length 2 with ReLU activation; 2) 16 filters of length 4 with ReLU activation; 3) max pooling layer with pool size 2; 4) 16 filters of length 6 with ReLU activation; 5) 16 filters of length 8 with ReLU activation; 6) max pooling layer with pool size 2; 7) flatten and a softmax layer. The stride size for the filters is 1 in all layers. We refer to this network as *CNN-Net*.

For the sequence detection, we use three networks, two based on RNNs and one based on the SBRNN. The first network has 3 LSTM layers and a final softmax layer, where the length of the output of each LSTM layer is 40. Two different inputs are used with this network. In the first, the input is the same set of features as the Base-Net above. We refer to this network as *LSTM3-Net*. In the second, the input is the pretrained CNN-Net described above without the top softmax layer. In this network, the CNN-Net chooses the features directly from the received signal. We refer to this network as *CNN-LSTM3-Net*. Finally, we consider three layers

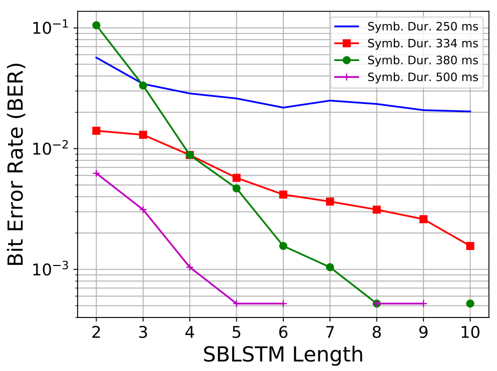


Fig. 9: The BER as a function of SBLSTM length.

of bidirectional LSTM cells, where each cell's output length is 40, and a final softmax layer. The input to this network is the same set of features used for Base-Net and the LSTM3-Net. When this network is used, during testing we use the SBRNN algorithm. We refer to this network as *SBLSTM3-Net*. For all the sequence detection algorithms, during testing, sample data sequences of the 120 bits are treated as an incoming data stream, and the detector estimates the bits one-by-one, simulating a real communication scenario. This demonstrates that these algorithms can work on any length data stream and can perform detection in real-time as data arrives at the receiver.

A. *System's Memory and ISI*

We first demonstrate that this communication system has a long memory. We use the RNN based detection techniques for this, and train the LSTM3-Net on sequences of 120 consecutive bits. The trained model is referred to as LSTM3-Net120. We run the trained model on the test data, once resetting the input state of the LSTM cell after each bit detection, and once passing the state as the input state for the next bit. Therefore, the former ignores the memory of the system and the ISI, while the latter considers the memory. The bit error rate (BER) performance for the memoryless LSTM3-Net120 detector is 0.1010 for 4 bps, and 0.0167 for 2 bps, while for the LSTM3-Net120 detector with memory, they are 0.0333 and 0.0005, respectively. This clearly demonstrates that the system has memory.

To evaluate the memory length, we train a length-10 SBLSTM3-Net on all sequences of 10 consecutive bits in the training data. Then, on the test data, we evaluate the BER performance for SBLSTM of length 2 to 10. Figure 9 shows the results for each symbol duration. The BER reduces as the length of the SBLSTM increases, again confirming that the system has memory. For example, for the 500 ms symbol duration, from the plot, we conclude that the memory is longer than 4. Note that some of the missing points for the 500 ms and 380 ms symbol durations are because there were zero errors in the test data. Moreover, BER values below 5×10^{-3} are not very accurate since the number of errors in the test dataset are less than 10 (in a typical BER plot the number of errors should be about 100). However, given enough test data, it would be possible to estimate the channel memory using the SBLSTM detector by finding the minimum length after which BER does not improve.

TABLE II: Bit Error Rate Performance

Symb. Dur.	250 ms	334 ms	380 ms	500 ms
Baseline	0.1297	0.0755	0.0797	0.0516
Base-Net	0.1057	0.0245	0.0380	0.0115
CNN-Net	0.1068	0.0750	0.0589	0.0063
CNN-LSTM3-Net120	0.0677	0.0271	0.0026	0.0021
LSTM3-Net120	**0.0333**	0.0417	0.0083	0.0005
SBLSTM3-Net10	0.0406	**0.0141**	**0.0005**	**0.0000**

B. *Performance and Resiliency*

Table II summarizes the best BER performance we obtain for all detection algorithms, including the baseline algorithm, by tuning all the hyperparameters using grid search. The number in front of the sequence detectors, indicates the sequence length. For example, LSTM3-Net120 is an LSTM3-Net that is trained on 120 bit sequences. In general, algorithms that use sequence detection perform significantly better than any symbol-by-symbol detection algorithm including the baseline algorithm. This is partly due to significant ISI present in the chemical communication platform. Overall, the proposed SBLSTM algorithm performs better than all other NN detectors considered.

Another important issue for detection algorithms are changing channel conditions and resiliency. As the channel conditions worsen, the received signal is further degraded, which increases the BER. Although we assume no channel coding is used in this work, one way to mitigate this problem is by using stronger channel codes that can correct some of the errors. However, given that the NN detectors rely on training data to tune the detector parameters, overfitting may be an issue. To evaluate the susceptibility of NN detectors to this effect, we collect data with a pH probe that has a degraded response due to normal wear and tear.

We collect 20 samples of 120 bit sequence transmissions for each of the 250 ms and 500 ms symbol durations using this degraded pH probe. First, to demonstrate that the response of the probe is indeed degraded, we evaluate it using the baseline slope-based detection algorithm. The best BERs obtained using the baseline detector are 0.1583 and 0.0741 for symbol durations of 250 ms and 500 ms, respectively. These values are significantly larger than those in Table II, because of the degraded pH probe. We then use the SBLSTM3-Net10 and the LSTM3-Net120 trained on the data from the good pH, on the test data from the degraded pH. For the SBLSTM3-Net10, the BERs obtained are 0.0883 and 0.0142, and for the LSTM3-Net120, the BERs are 0.1254 and 0.0504. These results confirm again that the proposed SBRNN algorithm is more resilient to changing channel conditions than the RNN.

Finally, to demonstrate that the proposed SBRNN algorithm can be implemented as part of a real-time communication system, we use it as part of a text messaging service built on the experimental platform. We demonstrate that using the SBRNN for detection at the receiver, we are able to reliably transmit and receive messages at 2 bps. This data rate is an order of magnitude higher than previous systems [43], [44].

VII. Conclusions

This work considered a machine learning approach to the detection problem in communication systems. In this scheme,

TABLE III: Set of features that are extracted from the received signal and are used as input to different NN detectors in this paper. These values have been selected such that the trained network achieves the best result on a small validation set.

Feature/Parameter	B	γ	$\mathbf{b}$	$\mathbf{d}$	$\hat{b}_0$ & $\hat{b}_{B-1}$	mean & var $\hat{\mathbf{b}}$	τ
Sec. V: Optical Channel	10	1	No	Yes	Yes	Yes	Yes
Sec. V: Molecular Channel	10	1000	No	Yes	Yes	Yes	Yes
Sec. VI: Base-Net	9	1	No	Yes	Yes	Yes	Yes
Sec. VI: CNN-Net	30	1	Yes	No	No	No	No
Sec. VI: CNN-LSTM3-Net120	30	1	Yes	No	No	No	No
Sec. VI: LSTM3-Net120	9	1	No	Yes	Yes	Yes	Yes
Sec. VI: SBLSTM3-Net10	9	1	No	Yes	Yes	Yes	Yes

a neural network detector is directly trained using measurement data from experiments, data collected in the field, or data generated from channel models. Different NN architectures were considered for symbol-by-symbol and sequence detection. For channels with memory, which rely on sequence detection, the SBRNN detection algorithm was presented for real-time symbol detection in data streams. To evaluate the performance of the proposed algorithm, the Poisson channel model for molecular communication was considered as well as the VD for this channel. It was shown that the proposed SBRNN algorithm can achieve a performance close to the VD with perfect CSI, and better than the RNN detector and the VD with CSI estimation error. Moreover, it was demonstrated that using a rich training dataset that contains sample transmission data under various channel conditions, the SBRNN detector can be trained to be resilient to the changes in the channel, and achieves a good BER performance for a wide range of channel conditions. Finally, to demonstrate that this algorithm can be implemented in practice, a molecular communication platform that uses multiple chemicals for signaling was used. Although the underlying channel model for this platform is unknown, it was demonstrated that NN detectors can be trained directly from experimental data. The SBRNN algorithm was shown to achieve the best BER performance among all other considered algorithms based on NNs as well as a slope detector considered in previous work. Finally, a text messaging application was implemented on the experimental platform for demonstration where it was shown that reliable communication at rates of 2 bps is possible, which is an order of magnitude faster than the data rate reported in previous work for molecular communication channels.

As part of future work we plan to investigate how techniques from reinforcement learning could be used to better respond to changing channel conditions. We would also like to study if the evolution of the internal state of the SBRNN detector could help in developing channel models for systems where the underlying models are unknown.

APPENDIX
FEATURE EXTRACTION

In this appendix we describe the set of features that are extracted from the received signal and are used as the input to the different NN detectors considered in this work. The set of features $\mathbf{r}_k$, extracted from the received signal during the k^{th}channel use $\mathbf{y}_k$, must preserve and summarize the important information-bearing components of the received signal. For the Poisson channel, since the information is encoded in the intensity of the signal, much of the information is contained in the rate of change of intensity. In particular, intensity increases in response to the transmission of the 1-bit, while intensity decreases or remains the same in response to transmission of the 0-bit. Note that this is also true for the pH signal in the experimental platform used in Section VI. First the symbol interval (i.e., the time between the green lines in Figure 6) is divided into a number of equal subintervals or bins. Then the values inside each bin are averaged to represent the value for the corresponding bin. Let B be the number of bins, and $\mathbf{b} = [b_0, b_1, \cdots, b_{B-1}]$ the corresponding values of each bin. We then extract the rate of change during a symbol duration by differentiating the bin vector to obtain the vector $\mathbf{d} = [d_0, d_1, \cdots, d_{B-2}]$, where $d_{i-1} = b_i - b_{i-1}$. We refer to this vector as the *slope vector* and use it as part of the feature set $\mathbf{r}_k$ extracted from the received signal.

Other values that can be used to infer the rate of change are b_0 and b_{B-1}, the value of the first and the last bins, and the mean and the variance of the $\mathbf{b}$. Since the intensity can grow large due to ISI, $\mathbf{b}$ may be normalized with the parameter γ as $\hat{\mathbf{b}} = \mathbf{b}/\gamma$. Therefore, instead of b_0 and b_{B-1}, $\hat{b}_0$ and $\hat{b}_{B-1}$, and the mean and the variance of the $\hat{\mathbf{b}}$ may be used as part of the feature set $\mathbf{r}_k$. Finally, since the transmitter and the receiver have to agree on the symbol duration, the receiver knows the symbol duration, which can be part of the feature set. Table III summarizes the set of features that are used as input to the each of the NN detection algorithms in this paper.

ACKNOWLEDGMENT

The authors like to thank Chan-Byoung Chae, Yonathan Morin, Milind Rao, and Morteza Mardani Korani for helpful discussions that resulted in improvements in this manuscript.

REFERENCES

[1] M. Stojanovic and J. Preisig, "Underwater acoustic communication channels: Propagation models and statistical characterization," *IEEE Communications Magazine*, vol. 47, no. 1, pp. 84–89, 2009.

[2] Y. Moritani, S. Hiyama, and T. Suda, "Molecular communication for health care applications," in *Proc. of 4th Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, Pisa, Italy, 2006, p. 5.

[3] I. F. Akyildiz, F. Brunetti, and C. Blazquez, "Nanonetworks: A new communication paradigm," *Computer Networks*, vol. 52, no. 12, pp. 2260–2279, August 2008.

[4] T. Nakano, A. W. Eckford, and T. Haraguchi, *Molecular communication*. Cambridge University Press, 2013.

[5] N. Farsad, H. B. Yilmaz, A. Eckford, C. B. Chae, and W. Guo, "A comprehensive survey of recent advancements in molecular communication," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1887–1919, thirdquarter 2016.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14539

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, Nov. 2016.

[8] M. Ibnkahla, "Applications of neural networks to digital communications a survey," *Signal Processing*, vol. 80, no. 7, pp. 1185–1215, 2000.

[9] N. Farsad and A. Goldsmith, "A molecular communication system using acids, bases and hydrogen ions," in *2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2016, pp. 1–6.

[10] B. Grzybowski, *Chemistry in Motion: Reaction-Diffusion Systems for Micro- and Nanotechnology*. Wiley, 2009.

[11] L. Debnath, *Nonlinear partial differential equations for scientists and engineers*. Springer Science & Business Media, 2011.

[12] B. Aazhang, B. P. Paris, and G. C. Orsak, "Neural networks for multiuser detection in code-division multiple-access communications," *IEEE Transactions on Communications*, vol. 40, no. 7, pp. 1212–1222, Jul 1992.

[13] U. Mitra and H. V. Poor, "Neural network techniques for adaptive multiuser demodulation," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 9, pp. 1460–1470, Dec 1994.

[14] J. J. Murillo-fuentes, S. Caro, and F. Pérez-Cruz, "Gaussian processes for multiuser detection in cdma receivers," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, P. B. Schölkopf, and J. C. Platt, Eds. MIT Press, 2006, pp. 939–946.

[15] Y. Işık and N. Taşpınar, "Multiuser detection with neural network and pic in cdma systems for awgn and rayleigh fading asynchronous channels," *Wireless Personal Communications*, vol. 43, no. 4, pp. 1185–1194, 2007.

[16] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sept 2016.

[17] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning-based communication over the air," *arXiv preprint arXiv:1707.03384*, 2017.

[18] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," in *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Dec 2016, pp. 223–228.

[19] E. Nachmani, E. Marciano, D. Burshtein, and Y. Be'ery, "RNN decoding of linear block codes," *arXiv preprint arXiv:1702.07560*, 2017.

[20] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE Journal of Selected Topics in Signal Processing*, 2018.

[21] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE Journal of Selected Topics in Signal Processing*, 2018.

[22] S. Cammerer, T. Gruber, J. Hoydis, and S. t. Brink, "Scaling deep learning-based decoding of polar codes via partitioning," *arXiv preprint arXiv:1702.06901*, 2017.

[23] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning-based communication over the air," *IEEE Journal of Selected Topics in Signal Processing*, 2017.

[24] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," *arXiv preprint arXiv:1706.01151*, 2017.

[25] C. Lee, H. B. Yilmaz, C.-B. Chae, N. Farsad, and A. Goldsmith, "Machine learning based channel modeling for molecular MIMO communications," in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017.

[26] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning approximate neural estimators for wireless channel state information," *arXiv preprint arXiv:1707.06260*, 2017.

[27] T. J. O'Shea and J. Hoydis, "An introduction to machine learning communications systems," *arXiv preprint arXiv:1702.00832*, 2017.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[30] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[31] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1764–1772.

[32] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2016, pp. 173–182.

[33] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv:1409.0473 [cs, stat]*, Sep. 2014.

[34] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[35] Z. Li and Y. Yu, "Protein Secondary Structure Prediction Using Cascaded Convolutional and Recurrent Neural Networks," *arXiv:1604.07176*, 2016.

[36] S. R. Z. Ghassemlooy, W. Popoola, *Optical Wireless Communications: System and Channel Modelling with MATLAB*, 1st ed. CRC Press, 2012.

[37] C. Gong and Z. Xu, "Channel estimation and signal detection for optical wireless scattering communication with inter-symbol interference," *IEEE Transactions on Wireless Communications*, vol. 14, no. 10, pp. 5326–5337, Oct 2015.

[38] G. Aminian, H. Arjmandi, A. Gohari, M. Nasiri-Kenari, and U. Mitra, "Capacity of diffusion-based molecular communication networks over lti-poisson channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 1, no. 2, pp. 188–201, June 2015.

[39] V. Jamali, A. Ahmadzadeh, C. Jardin, H. Sticht, and R. Schober, "Channel estimation for diffusive molecular communications," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 423—4252, Oct 2016.

[40] V. Jamali, A. Ahmadzadeh, N. Farsad, and R. Schober, "Scw codes for optimal csi-free detection in diffusive molecular communications," in *IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 3190–3194.

[41] V. Jamali, N. Farsad, R. Schober, and A. Goldsmith, "Non-coherent detection for diffusive molecular communications," *arXiv preprint arXiv:1707.08926*, 2017.

[42] D. P. N. Farsad and A. Goldsmith, "A novel experimental platform for in-vessel multi-chemical molecular communications," in *IEEE Global Communications Conference (GLOBECOM)*, 2017.

[43] N. Farsad, W. Guo, and A. W. Eckford, "Tabletop molecular communication: Text messages through chemical signals," *PLOS ONE*, vol. 8, no. 12, p. e82935, Dec 2013.

[44] B. H. Koo, C. Lee, H. B. Yilmaz, N. Farsad, A. Eckford, and C. B. Chae, "Molecular MIMO: From theory to prototype," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 600–614, March 2016.

[45] A. J. Viterbi and J. K. Omura, *Principles of digital communication and coding*. Courier Corporation, 2013.

[46] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.

[47] T. M. Cover and J. A. Thomas, *Elements of Information Theory 2nd Edition*, 2nd ed. Wiley-Interscience, 2006.

[48] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[49] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[51] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[52] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[53] A. Graves, S. Fernndez, and F. Gomez, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *In Proceedings of the International Conference on Machine Learning, ICML 2006*, 2006, pp. 369–376.

[54] S. Shamai, "Capacity of a pulse amplitude modulated direct detection photon channel," *IEE Proceedings I - Communications, Speech and Vision*, vol. 137, no. 6, pp. 424–430, Dec 1990.

[55] J. Cao, S. Hranilovic, and J. Chen, "Capacity-achieving distributions for the discrete-time poisson channel–Part I: General properties and numerical techniques," *IEEE Transactions on Communications*, vol. 62, no. 1, pp. 194–202, 2014.

[56] N. Farsad, C. Rose, M. Mdard, and A. Goldsmith, "Capacity of molecular channels with imperfect particle-intensity modulation and detection," in *IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2468–2472.

[57] N. Hayasaka and T. Ito, "Channel modeling of nondirected wireless infrared indoor diffuse link," *Electronics and Communications in Japan (Part I: Communications)*, vol. 90, no. 6, pp. 9–19, 2007.

[58] A. K. Majumdar, C. E. Luna, and P. S. Idell, "Reconstruction of probability density function of intensity fluctuations relevant to free-space laser communications through atmospheric turbulence," in *Proc. SPIE*, vol. 6709, 2007, p. 67090.

[59] H. Ding, G. Chen, A. K. Majumdar, B. M. Sadler, and Z. Xu, "Modeling of non-line-of-sight ultraviolet scattering channels for communication," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 9, 2009.

[60] K. V. Srinivas, A. Eckford, and R. Adve, "Molecular communication in fluid media: The additive inverse gaussian noise channel," *IEEE Transactions on Information Theory*, vol. 58, no. 7, pp. 4678–4692, 2012.

[61] A. Noel, K. C. Cheung, and R. Schober, "Optimal receiver design for diffusive molecular communication with flow and additive noise," *IEEE Transactions on NanoBioscience*, vol. 13, no. 3, pp. 350–362, Sept 2014.

[62] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, March 1973.

[63] X. Lingyun and D. Limin, “Efficient viterbi beam search algorithm using dynamic pruning,” in *Proceedings. of 7th International Conference on Signal Processing*, vol. 1, Aug 2004, pp. 699–702.

[64] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.