Deep Reinforcement Learning Autoencoder with Noisy Feedback

Mathieu Goutay, Fayçal Ait Aoudia, and Jakob Hoydis
Nokia Bell Labs, Paris-Saclay, 91620 Nozay, France
mathieu.goutay@insa-lyon.fr, {faycal.ait_aoudia, jakob.hoydis}@nokia-bell-labs.com

***Abstract*—End-to-end learning of communication systems enables joint optimization of transmitter and receiver, implemented as deep neural network (NN)-based autoencoders, over any type of channel and for an arbitrary performance metric. Recently, an *alternating training* procedure was proposed which eliminates the need for an explicit channel model. However, this approach requires feedback of real-valued losses from the receiver to the transmitter during training. In this paper, we first show that alternating training works even with a noisy feedback channel. Then, we design a system that learns to transmit real numbers over an unknown channel without a preexisting feedback link. Once trained, this *feedback system* can be used to communicate losses during alternating training of autoencoders. Evaluations over additive white Gaussian noise (AWGN) and Rayleigh block-fading (RBF) channels show that end-to-end communication systems trained using the proposed feedback system achieve the same performance as when trained with a perfect feedback link.**

I. Introduction

For the last decades, engineers have built communication systems by splitting the transmitter and receiver into multiple components, each optimized for a specific task. This modular block-structure is the basis for the robust and versatile communication systems we have today. Since the early days, machine learning (ML) techniques were considered for specific problems in communication [1]. For various reasons, however, they have never become the de facto solution and were rarely implemented in commercial products.

Recently, it was proposed [2] to interpret end-to-end communication systems as *autoencoders* [3], where the transmitter and receiver are implemented as deep neural networks (NNs). The main practical drawback of this theoretically very appealing idea is the need for a mathematical channel model to perform the training. This makes its application to real channels of practical interest challenging. A first approach to circumvent this problem [4] consists in training the system using a channel model, and then fine-tuning the receiver with measured data. A shortcoming of this approach is sub-optimal training of the transmitter which makes it not fully satisfactory. Another solution studied by multiple authors [5], [6] consist in learning a differentiable channel model in the form of a generative adversarial network (GAN), which can then be used for supervised autoencoder training. However, it still needs to be shown that this approach works for practical channels. Lastly, the authors of [7] proposed a third approach called *alternating training*, where the autoencoder is trained by alternating supervised learning of the receiver, and reinforcement learning (RL) of the transmitter. A similar method is used in [8], but the receiver is not actually trained and simply detects symbols through clustering. Another model-free approach is developed in [9] based on simultaneous perturbation methods. Although all of these methods do not require any knowledge of the channel and can be performed directly with real hardware, a reliable feedback link is needed during training, from the receiver to the transmitter, as illustrated in Fig. 1.

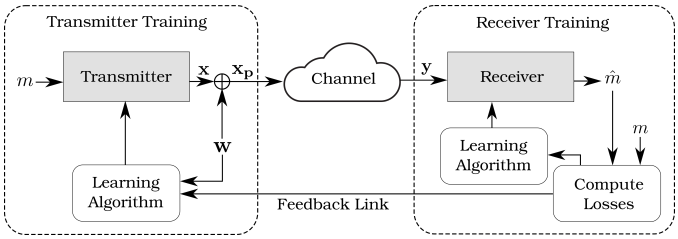


Fig. 1. Training of a communication system using a feedback link

In this work, we show that alternating training of end-to-end communication systems can be performed with noisy feedback, up to a certain point, without performance loss. Based on this observation, we design a system which learns to communicate real values over an unknown channel and without the need for any reliable link. Once trained, this *feedback system* can be used in lieu of the perfect feedback link assumed in [7]. Therefore, the contribution of this paper, combined with the alternating training algorithm of [7], enables training of end-to-end communication systems without channel model and without an extra feedback link. Evaluation of the proposed scheme on additive white Gaussian noise (AWGN) and Rayleigh block-fading (RBF) channels shows that the proposed approach achieves the same performance as training with perfect feedback. Moreover, the learned system outperforms a coherent quadrature phase-shift keying (QPSK) baseline as well as a highly optimized higher-order sphere packing scheme [10] on the RBF channel.

Notations : Matrices and column vectors are denoted by bold upper- and lower-case letters, respectively. $x^{(i)}$ is the i^{th} element of $\mathbf{x}$. $||\mathbf{X}||_F$ denotes the Frobenius norm of $\mathbf{X}$. $[\mathbf{x}]_{\top a}^{\perp b}$ is the restriction of the elements of $\mathbf{x}$ in the interval $[a, b]$ by clipping. $\Re(\mathbf{x})$ and $\Im(\mathbf{x})$ denote the real and the imaginary part of $\mathbf{x}$, respectively. $\mathbb{1}_N$ is the N-dimensional all-one vector and $\mathbf{I}_N$ the N–by–N identity matrix. The gradient and Jacobian operators with respect to a vector $\boldsymbol{\theta}$ are both denoted $\nabla_{\boldsymbol{\theta}}$.

II. Learning of a Communication System

In an ML-based end-to-end communication system, the transmitter and the receiver are both implemented as deep NNs, with parameters (weights, bias) $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_R$, respectively. The transmitter $f^T_{\boldsymbol{\theta}_T} : \mathbb{M} \to \mathbb{C}^{N_c}$ maps a message m uniformly drawn from a finite discrete set $\mathbb{M} = \{1, \dots, M\}$ to a vector of complex symbols $\mathbf{x} \in \mathbb{C}^{N_c}$, which are normalized and sent over the channel. N_c is the number of (complex) channel uses. The receiver $f^R_{\boldsymbol{\theta}_R} : \mathbb{C}^{N_c} \to \left\{\mathbf{p} \in \mathbb{R}^M_+ \mid \sum_{i=1}^M p^{(i)} = 1\right\}$ generates a probability distribution over $\mathbb{M}$ from the received signal $\mathbf{y}$, and hard decoding is achieved by taking the message $\hat{m}$ with highest probability. The channel acts as a stochastic black box with unknown transfer function which alters the sent signal $\mathbf{x}$ according to the conditional distribution $P(\mathbf{y}|\mathbf{x})$.

The key idea of alternating training [7] is to find the best sets of parameters by alternating between supervised learning of the receiver and RL-based learning of the transmitter (Fig. 1). The aim of receiver training is to learn an estimate of the conditional probability $P(m|\mathbf{y})$ and is performed by stochastic gradient descent (SGD) [3] on the cross-entropy (CE)

$$L(\boldsymbol{\theta}_R) = \frac{1}{S_c}\sum_{i=1}^{S_c} \underbrace{-\log\left(\left[f^R_{\boldsymbol{\theta}_R}(\mathbf{y}^{(i)})\right]^{(m^{(i)})}\right)}_{l^{(i)}} \tag{1}$$

where S_c is the *minibatch* size, i.e., the number of examples $m^{(i)}$ used to estimate the loss L, $l^{(i)}$ is the *per example loss*, $\mathbf{y}^{(i)}$ the signal received when $m^{(i)}$ was sent, and $\left[f^R_{\boldsymbol{\theta}_R}(\mathbf{y}^{(i)})\right]$ the probability distribution vector over all possible messages outputted by the receiver. It is assumed that the examples are independent and identically distributed (i.i.d.).

Transmitter training is achieved by adding a zero-mean stochastic perturbation $\mathbf{w}$ to the encoded message, i.e., $\mathbf{x}_p = \mathbf{x}+\mathbf{w}$ denotes the resulting signal transmitted over the channel. The perturbation is chosen to be Gaussian i.i.d. with a variance σ_c, $\mathbf{w} \sim \mathcal{CN}(\mathbf{0}, \sigma_c^2\mathbf{I}_{N_c})$, so the conditional distribution of $\mathbf{x}_p$ is denoted by $\pi_{\boldsymbol{\theta}_T}(\mathbf{x}_p|m) = \mathcal{CN}(f^T_{\boldsymbol{\theta}_T}(m), \sigma_c^2\mathbf{I}_{N_c})$. The receiver computes the per-example losses $l^{(i)}$ for all transmitted messages $m^{(i)}$, and sends them back to the transmitter, which optimizes its parameters $\boldsymbol{\theta}_T$ by SGD. The gradient of the aggregate loss $J(\mathbf{m}, \mathbf{l}, \mathbf{X}_p)$ under the *policy* $\pi_{\boldsymbol{\theta}_T}$ is estimated using the policy gradient theorem [11] from the field of RL :

$$\nabla_{\boldsymbol{\theta}_T} J(\mathbf{m}, \mathbf{l}, \mathbf{X}_p) = \frac{1}{S_c}\sum_{i=1}^{S_c} l^{(i)} \underbrace{\nabla_{\boldsymbol{\theta}_T} \log\left(\pi_{\boldsymbol{\theta}_T}(\mathbf{x}_p^{(i)}|m^{(i)})\right)}_{\triangleq\, D^{(i)}} \tag{2}$$

$$= \frac{1}{S_c}\sum_{i=1}^{S_c} \frac{2l^{(i)}}{\sigma_c^2}\left(\nabla_{\boldsymbol{\theta}_T} f^T_{\boldsymbol{\theta}_T}\left(m^{(i)}\right)\right)^{\mathsf{T}}\left(\mathbf{x}_p^{(i)} - f^T_{\boldsymbol{\theta}_T}\left(m^{(i)}\right)\right)$$

where $\mathbf{l}$ is the vector of per-example losses $l^{(i)}$ and $\mathbf{X}_p$ the matrix of perturbed vectors $\mathbf{x}_p^{(i)}$.

In [7], the authors assumed that a perfectly reliable link is available to feed back the losses computed by the receiver to the transmitter, which is hardly the case in practice. In the next section, we study the effect of *noisy* feedback on the end-to-end system training process.

III. Effect of Noisy Losses on Training

Assuming a noisy scalar feedback channel, we denote by $\boldsymbol{\varepsilon}$ the error introduced by the channel, such that $\tilde{\mathbf{l}} = \mathbf{l} + \boldsymbol{\varepsilon}$, where $\mathbf{l}$ denotes the true loss vector, and $\tilde{\mathbf{l}}$ the noisy loss. We model the feedback channel as being AWGN, i.e., $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_l^2\mathbf{I}_{S_c})$. Note that σ_l^2 is the mean squared error (MSE) between $\mathbf{l}$ and $\tilde{\mathbf{l}}$. We denote by $\nabla_{\boldsymbol{\theta}_T}\tilde{J}$ the loss function gradient estimator obtained by replacing the true losses by noisy ones in (2). All the numerical evaluations of this section were performed with $S_c = 10^5$, $M = 256$, $N_c = 4$, and $\sigma_c^2 = 0.02$.

A. Effect of the Noise Variance on the Loss Function Gradient

As the error $\boldsymbol{\varepsilon}$ is assumed to have zero-mean, one can easily see that the loss function gradient estimator is not biased, i.e., $\mathbb{E}\left[\nabla_{\boldsymbol{\theta}_T}\tilde{J}\right] = \mathbb{E}\left[\nabla_{\boldsymbol{\theta}_T} J\right]$. However, noisy feedback increases the estimator variance, interfering with the communication system training. To get insight into this issue, one can compute the cumulative element-wise variance of $\nabla_{\boldsymbol{\theta}_T}\tilde{J}$, denoted by V:

$$\begin{aligned} V &\triangleq \mathbb{E}\left[\left\|\nabla_{\boldsymbol{\theta}_T}\tilde{J} - \mathbb{E}\left[\nabla_{\boldsymbol{\theta}_T} J\right]\right\|_2^2\right] \\ &= \mathbb{E}\left[\left\|\frac{1}{S_c}\sum_{i=0}^{S_c}\tilde{l}^{(i)}D^{(i)} - \mathbb{E}\left[\frac{1}{S_c}\sum_{i=0}^{S_c} l^{(i)}D^{(i)}\right]\right\|_2^2\right] \\ &= \frac{1}{S_c}\left(\underbrace{\mathbb{E}\left[||lD - \mathbb{E}\left[lD\right]||_2^2\right]}_{\triangleq A} + \underbrace{\sigma_l^2\mathbb{E}\left[||D||_2^2\right]}_{\triangleq B}\right) \end{aligned} \tag{3}$$

where l and D refer to the random variables generating the samples $l^{(i)}$ and $D^{(i)}$ (as defined in (2)), respectively. It can be noticed that A is the cumulative element-wise variance of the loss function gradient estimator without noise $\nabla_{\boldsymbol{\theta}_T} J$. Having noisy feedback therefore increases V by the term B which can be written as:

$$B = \frac{4\sigma_l^2}{\sigma_c^4}\mathbb{E}\left[||\nabla_{\theta_T} f^T_{\theta_T}(m)||_F^2\right]. \tag{4}$$

Fig. 2 shows a numerical evaluation of V as a function of σ_l^2 for an untrained system, after 1000 training iterations, and for a system trained until no significant progress could be observed. In the three considered cases, V is not significantly impacted by increasing values of σ_l^2 up to a certain point, from which V increases linearly with σ_l^2. From the above analysis, this phenomenon can be explained by the minor impact of B on V compared to A for low values of σ_l^2. However, as σ_l^2 increases, the value of B grows up to the point that it becomes the predominant term. Another remarkable results of this numerical evaluations is the decrease of V with the number of training iterations. This can be explained by the fact that the losses are positive and bounded, and that the training process aims to reduce their expected value, leading simultaneously to a decreased variance.

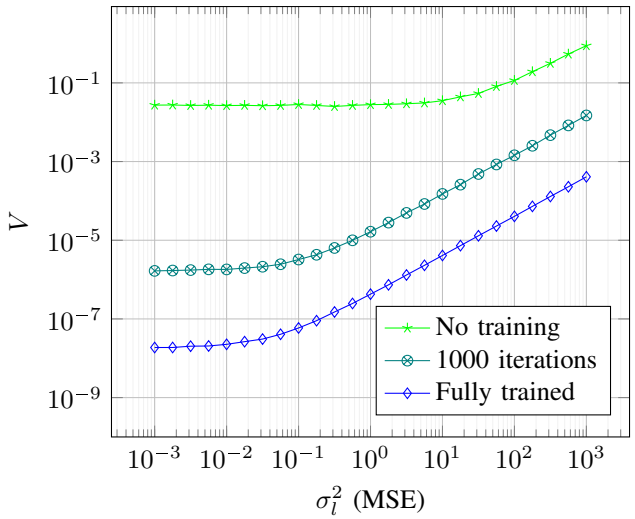


Fig. 2. Numerical evaluation of the cumulative element-wise variance of the loss function gradient estimator V for different values of σ_l^2 (MSE)

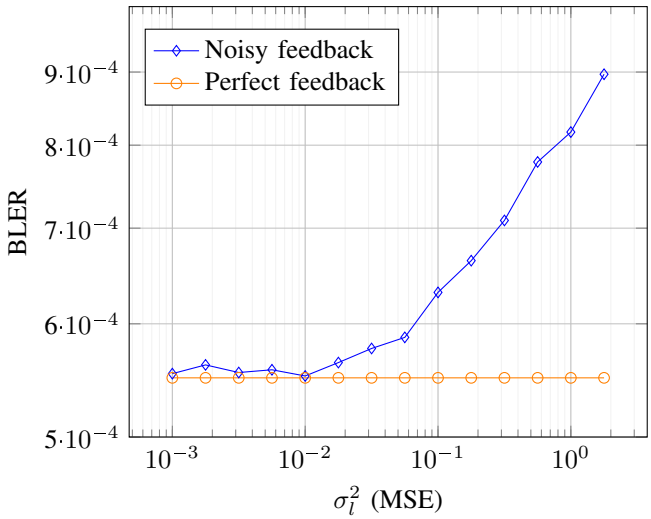


Fig. 3. Comparison of the BLER achieved by communication systems trained with noisy and perfect feedback

B. Impact of the Noise Variance on the BLER

Fig. 3 compares the block error rate (BLER), i.e., $\Pr(\hat{m} \neq m)$, achieved by a fully trained autoencoder using alternative training with perfect feedback against that achieved with noisy feedback. Both training and evaluation were done on an AWGN channel with a signal-to-noise ratio (SNR) of 10 dB as defined in (5) in Section V. Up to values of $\sigma_l^2 \approx 10^{-2}$, the system with noisy feedback achieves the same performance as if it was trained with a perfect feedback link. For higher values of σ_l^2, the achieved BLER increases quickly with σ_l^2. These results can be explained by the high variance of the gradient loss estimator for values of σ_l^2 higher than 10^{-2}, leading to poor training of the transmitter.

These results suggest that training an autoencoder with noisy feedback can lead to the same BLER performance as with perfect feedback, as long as the MSE of the noisy feedback link stays below a threshold, estimated here as being approximately 10^{-2}. Motivated by this observation, we present in the next section a system which learns to transmit real scalars. This system achieves a low enough MSE so that it can be used, once trained, to feed back the losses in the alternating training scheme, as shown in Fig. 4, and enables the same BLER performance as in [7] with a perfect feedback.

IV. Learning the Communication of Real Numbers

This section presents an ML-based end-to-end system for the transmission of real numbers, referred to as *feedback system*, in opposition to the *communication system* presented in Section II which sends messages. The corresponding training algorithm is also introduced, and does not require any channel model or preexisting feedback link. It is assumed that the real numbers to be transmitted take values in $[0, 1]$. Two *devices*, denoted by A and B, are each assumed to be equipped with a transmitter and a receiver, allowing data transmission and reception. Transmitter A aims to communicate real numbers to receiver B, and similarly transmitter B aims to communicate real numbers to receiver A. This is illustrated in Fig. 5.

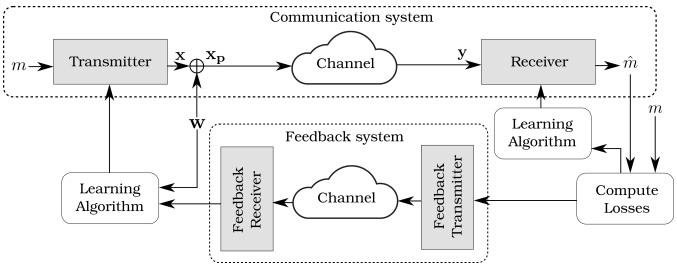


Fig. 4. Training of a communication system using a feedback system

Transmitter $X \in \{\mathrm{A}, \mathrm{B}\}$ implements the mapping: $f^{T_X}_{\boldsymbol{\theta}_{T_X}} : \mathbb{R} \to \mathbb{C}^{N_f}$, where $\boldsymbol{\theta}_{T_X}$ is the parameter vector of transmitter X, and N_f the number of channel uses required to send a single real number. The final layer normalizes the average energy per channel symbol, such that $\mathbb{E}\left[\frac{1}{N_f}\|\mathbf{x}\|_2^2\right] = 1$. Receiver $X \in \{\mathrm{A}, \mathrm{B}\}$ implements the mapping $f^{R_X}_{\boldsymbol{\theta}_{R_X}} : \mathbb{C}^{N_f} \to \mathbb{R}$, $\boldsymbol{\theta}_{R_X}$ being the parameter vector of receiver X. The proposed training algorithm, which can be seen as an extension of the one proposed in [7], is presented in the following.

A. Receiver Training

Due to the symmetry between the devices A and B, only training of receiver B is described. First, transmitter A generates a minibatch of real numbers of size S_f containing realizations $\mathbf{r}$ distributed in $[0, 1]$. Each of these realizations is encoded by the transmitter into N_f complex symbols, creating an S_f–by–N_f matrix $\mathbf{X}$ that is sent over the channel (row-by-row). The receiver receives the perturbed symbols $\mathbf{Y}$ and decodes them into real numbers $\hat{\mathbf{r}}$. Finally, SGD is performed on $f^{R_B}_{\boldsymbol{\theta}_{R_B}}$ to minimize the MSE between $\mathbf{r}$ and $\hat{\mathbf{r}}$, the CE only being used when transmitting messages. It is assumed that device B is aware of the true real numbers sent by the device A. This can easily be done in practice, e.g., using pseudo-random number generators initialized with the same seed. Algorithm 1 shows training of a receiver.

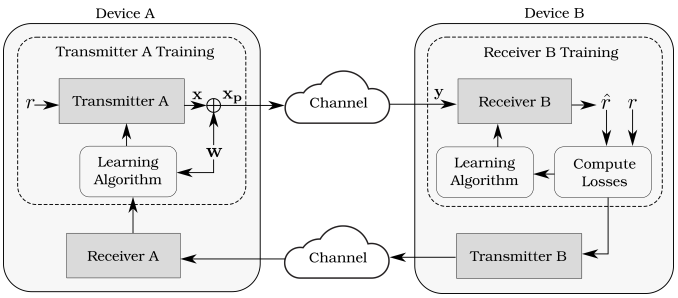


Fig. 5. Training of a feedback system

Algorithm 1 : function TRAINRECEIVERB

repeat
  $\mathbf{r} \leftarrow \text{TRAININGSOURCE}(S_f)$
  $\mathbf{X} \leftarrow f^{(T_A)}_{\boldsymbol{\theta}_{T_A}}(\mathbf{r})$ } Transmitter A
  $\triangleright$ Channel : $\mathbf{Y} \sim P(\mathbf{Y}|\mathbf{X})$
  $\hat{\mathbf{r}} \leftarrow f^{(R_B)}_{\boldsymbol{\theta}_{R_B}}(\mathbf{Y})$
  $\mathbf{r} \leftarrow \text{TRAININGSOURCE}(S_f)$ } Receiver B
  $\text{SGD}(\boldsymbol{\theta}_{R_B}, \mathbf{r}, \hat{\mathbf{r}})$
until Stop criterion is met

B. Transmitter Training

Similarly, only training of transmitter A is presented due to symmetry. Transmitter A starts by creating a minibatch of real numbers of size S_f and encodes each example into N_f complex symbols to form the matrix $\mathbf{X}$ of size S_f–by–N_f. Note that the batch size of the receiver and transmitter training are chosen equal for simplicity but could be different. To enable exploration, a stochastic perturbation is added to the output of the transmitter. More precisely, a perturbation matrix $\mathbf{W}$ is generated by sampling a circular complex Gaussian distribution with variance σ_f^2. In order to achieve unit energy channel symbols, the output of the transmitters are scaled before the perturbation is added, and the matrix containing the sent symbols is given by $\mathbf{X}_p = \sqrt{1-\sigma_f^2}\mathbf{X} + \mathbf{W}$.

The receiver receives the matrix $\mathbf{Y}$ and decodes the symbols into real numbers $\hat{\mathbf{r}}$. The per-examples losses are computed as being the per-symbol squared error. Because the system is trained to communicate real numbers in both ways, it is possible to transmit the losses back to the device A using the transmitter B, and thus avoiding the need of a preexisting reliable feedback link. Finally, SGD is performed on the transmitter weights using an estimation of the gradient obtained by the policy gradient theorem [11]. Algorithm 2 summarizes the training procedure of a transmitter.

C. Main Loop

Algorithm 3 shows the main loop of the proposed algorithm. Each iteration is divided into two steps. In the first step, transmitter A and receiver B are trained iteratively until a certain criterion is met. In the second step, transmitter B and receiver A are trained in the same way. The intuition is that, by alternating between optimization of a transmitter and a receiver, communication errors will decrease. A key idea of this algorithm is to use one transmitter-receiver pair to transmit the losses which are needed for training of the other pair. We expect that as the training progresses, each pair will get better at reliably communicating real numbers (and hence losses), enabling more accurate optimization of the other pair.

Algorithm 2 : function TRAINTRANSMITTERA

repeat
  $\mathbf{r} \leftarrow \text{TRAININGSOURCE}(S_f)$
  $\mathbf{X} \leftarrow f^{(T_A)}_{\boldsymbol{\theta}_{T_A}}(\mathbf{r})$
  $\mathbf{W} \leftarrow \text{PERTURBATION}(\sigma_f)$ } Transmitter A
  $\mathbf{X}_p \leftarrow \sqrt{1-\sigma_f^2}\mathbf{X} + \mathbf{W}$
  $\triangleright$ Channel : $\mathbf{Y} \sim P(\mathbf{Y}|\mathbf{X}_p)$
  $\hat{\mathbf{r}} \leftarrow f^{(R_B)}_{\boldsymbol{\theta}_{R_B}}(\mathbf{Y})$
  $\mathbf{r} \leftarrow \text{TRAININGSOURCE}(S_f)$
  $\mathbf{l} \leftarrow \text{PEREXAMPLELOSSES}(\mathbf{r}, \hat{\mathbf{r}})$ } Receiver B
  $\text{SENDWITHTRANSMITTERB}(\mathbf{l})$
  $\hat{\mathbf{l}} \leftarrow \text{RECEIVEWITHRECEIVERA}$
  $\text{SGD}(\boldsymbol{\theta}_{T_A}, \hat{\mathbf{l}}, \mathbf{W})$ } Transmitter A
until Stop criterion is met

Algorithm 3 : Main loop

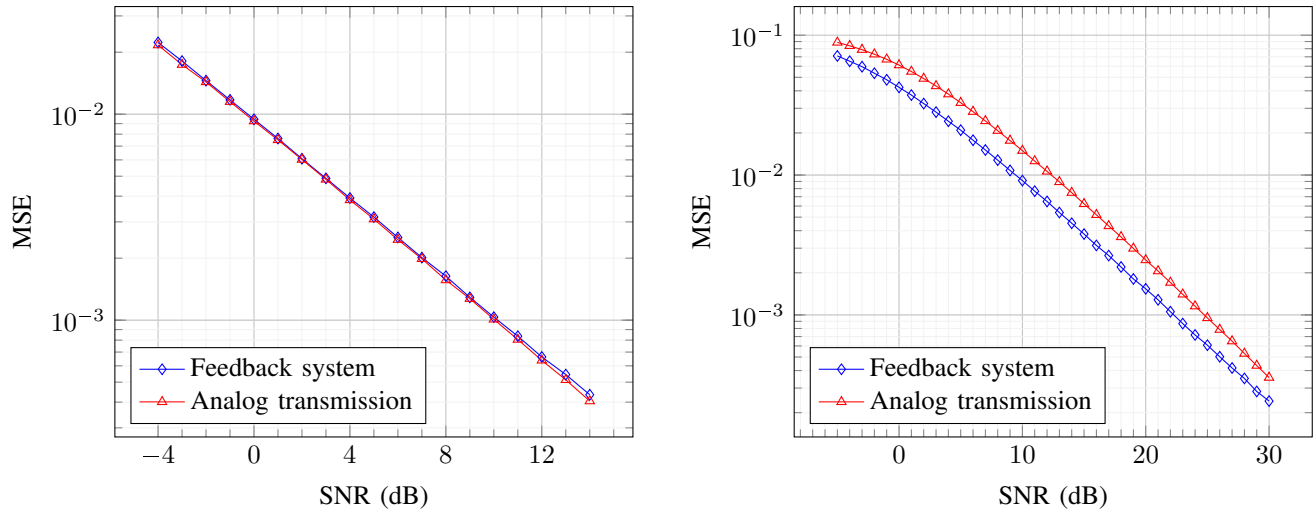
while Stop criterion not met **do**
  while Stop criterion not met **do**
    TRAINTRANSMITTERA
    TRAINRECEIVERB
  end while
  while Stop criterion not met **do**
    TRAINTRANSMITTERB
    TRAINRECEIVERA
  end while
end while

V. EVALUATION RESULTS

We first evaluate numerically the performance of the feedback system described above. Then, we compare the BLERs of the communication system of Section II trained using the alternating scheme with (i) perfect feedback [7] and (ii) the proposed feedback system. The SNR is defined as

$$\text{SNR} = \frac{\mathbb{E}\left[\frac{1}{N_c}\|\mathbf{x}\|_2^2\right]}{\sigma^2} = \frac{1}{\sigma^2} \tag{5}$$

where the last equality is due to normalization of channel symbols, and σ^2 is the channel noise variance. Training was performed using the Adam [12] optimizer. Evaluation was done on AWGN and RBF channels. For the feedback system, only the results for one transmitter-receiver pair are presented, as they are similar for the other one.



(a) AWGN channel

(b) RBF channel

Fig. 6. MSE achieved by the proposed feedback system and an analog transmission system over AWGN and RBF channels

A. Feedback System Transmitters and Receivers Architectures

Both transmitters of the feedback system are made of two dense layers, a first one of $10N_f$ units with ELU activation functions [3], and a second one of $2N_f$ units with linear activations. The output of the second layer form the real and imaginary parts of the N_f complex channel symbols used to transmit a real number. A last layer normalizes the average energy per symbol.

Regarding the receivers, their first layers split the N_f received complex symbols into $2N_f$ real numbers. Then, different architectures are used for the AWGN and RBF channel. In the AWGN channel case, a single hidden dense layer with $10N_f$ units and ReLu activations [3], followed by an output layer consisting of a single-neuron dense layer with linear activation was found to be enough. In the RBF channel case, the receiver adopts a radio transformer network (RTN) [2] architecture, similar to the one used in [7], the only differences being that the second to last layer has $10N_f$ units, and that the last layer has a single unit with linear activation.

B. Evaluation of the Feedback System

In Fig. 6, the MSE achieved by the proposed feedback system and by an analog scheme are presented for AWGN and RBF channels. Considering the AWGN and RBF case, N_f was set to 4 and 5 respectively, and training was performed at an SNR of 10 dB and 20 dB respectively. σ_f^2 was set to 0.02, S_f to 10^5, and real numbers r to be sent are drawn from a uniform distribution on [0,1], i.e., $r \sim \mathcal{U}(0,1)$.

In the AWGN channel case, each real number was transmitted using $N_f = 4$ channel uses. Before transmitting real numbers with the analog system, they are centered and scaled to have zero-mean and unit variance. The transmitted channel symbols vector corresponding to a real number r is therefore

$$\mathbf{x} = \frac{r - \mathbb{E}[r] + j\,(r - \mathbb{E}[r])}{\sqrt{2\text{var}(r)}} 1_{N_f}. \tag{6}$$

Decoding is done by averaging over the repetitions and proper scaling:

$$\hat{r} = \left[\frac{\sqrt{2\text{var}(r)}}{2N_f} \sum_{j=1}^{N_f} \left(\Re\left(y^{(j)}\right) + \Im\left(y^{(j)}\right) \right) + \mathbb{E}[r] \right]_{\top 0}^{\perp 1}. \tag{7}$$

Fig. 6a shows that over an AWGN channel, the feedback system and the analog system achieve similar performance. Moreover, when the SNR is greater than 0 dB, the MSE is lower than 10^{-2}, meaning from our previous analysis (see Section III) that either the feedback system or an analog system with repetition can be used in lieu of a perfect feedback link to perform alternating training.

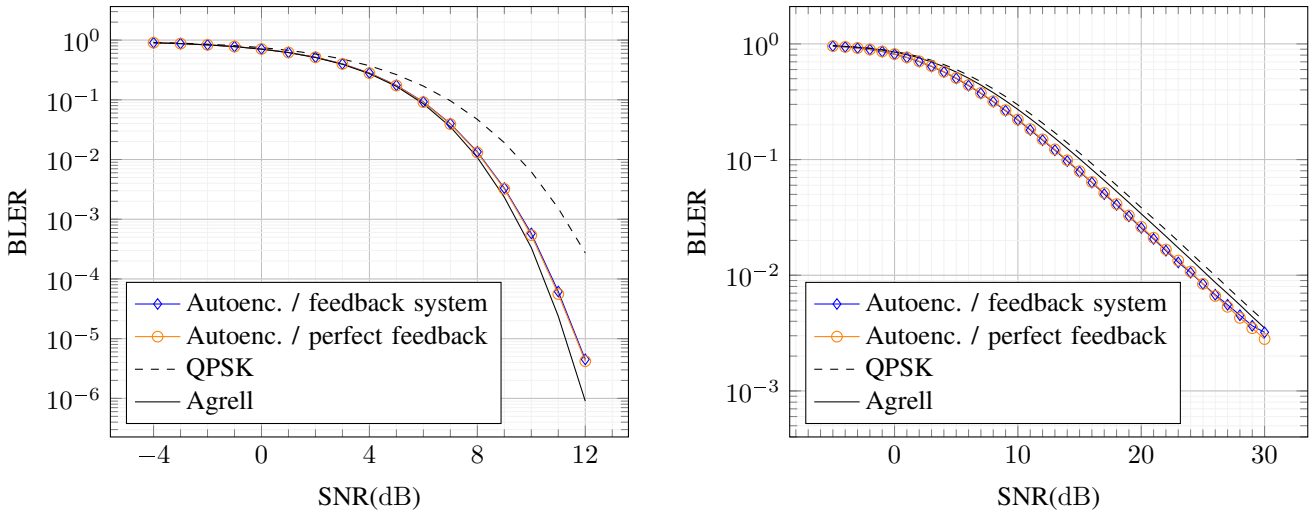
Regarding the RBF channel, comparison was done to an analog scheme with repetition over four channel uses in Fig. 6b. A pilot was added to perform equalization. Similarly to the AWGN case, real numbers are centered and scaled to have zero-mean and unit variance before sending. Detection is done by first computing an estimate of the channel gain $\hat{h}$ using the pilot, before averaging over the equalized received symbols and scaling:

$$\hat{r} = \left[\frac{\sqrt{2\text{var}(r)}}{2(N_f - 1)} \sum_{j=1}^{N_f} \left(\Re\left(\frac{y^{(j)}}{\hat{h}}\right) + \Im\left(\frac{y^{(j)}}{\hat{h}}\right) \right) + \mathbb{E}[r] \right]_{\top 0}^{\perp 1}. \tag{8}$$

For fairness, the number of channel uses for the feedback system was set to $N_f = 5$. Fig. 6b reveals that the feedback system outperforms the analog approach, with a maximum gain of approximately 3dB being achieved around an MSE of 10^{-2}. On an RBF channel, these results show that the feedback system is capable of providing an MSE of 10^{-2} at SNR = 10 dB, which should be sufficient for the alternative training.

C. Alternating Training using the Feedback System

Next, the performance of communication systems as introduced in Section II are evaluated. We are interested in BLERs achieved by communication systems, as introduced in Section II, when trained using feedback systems instead of a perfect



(a) AWGN channel

(b) RBF channel

Fig. 7. BLER achieved by the compared approaches over AWGN and RBF channels

feedback link. This is illustrated in Fig. 4. The communication systems transmit messages from a set of $M = 256$ messages, uses the same architecture as in [7], and losses are clipped in $[0, 1]$ before being sent back to the transmitter to match feedback systems' training conditions.

Comparison is done to a system trained with perfect link, as well as to QPSK and Agrell [10] modulation schemes. Agrell is a subset of the E8 lattice, designed by numerical optimization to solve the sphere packing problem for $M = 256$. To enable QPSK and Agrell on an RBF channel, an additional pilot symbol was used to perform equalization. For the AWGN and RBF channel, the communication and feedback systems were trained at 10dB and 20dB respectively in both directions, with $\sigma_c^2 = \sigma_f^2 = 0.02$ and $S_c = S_f = 10^5$. The number of channel uses was set to $N_c = N_f = 4$ for the AWGN channel, and $Nc = N_f = 5$ for fairness for the RBF channel.

Fig. 7 shows the BLER of trained communication systems over AWGN and RBF channels. It can be seen that the communication systems trained using the feedback system achieve the same performance as those trained with perfect feedback. This demonstrates the effectiveness of the proposed approach. In the AWGN channel case, our communication system outperforms QPSK but has a slightly higher BLER than Agrell. This is not surprising as Agrell is a close-to-optimal solution to the sphere packing problem, leading to high performance on the AWGN channel. However, our communication system outperforms both QPSK and Agrell over the RBF channel. This shows that it is able to discover a more robust scheme, even without any channel model or reliable feedback link.

VI. Conclusion

We have shown that alternating training [7] of ML-based communication systems can be performed with noisy feedback, up to a certain MSE, without performance loss. We then proposed a feedback system able to learn the transmission of real numbers without channel model or preexisting feedback link. This feedback system can be used in lieu of the perfect feedback link to perform alternating training. Finally, evaluations show that the feedback system leads to identical performance compared to that achieved with a perfect feedback link, conditioned on a sufficiently high but realistic training SNR. Moreover, our communication system outperforms both QPSK and a highly-optimized higher-order modulation scheme on an RBF channel.

References

[1] B. Widrow and M. E. Hoff, "Adaptive switching circuits," Stanford Univ Ca Stanford Electronics Labs, Tech. Rep., 1960.

[2] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[4] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning-based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[5] T. J. O'Shea, T. Roy, and N. West, "Approximating the void: Learning stochastic channel models from observation with variational generative adversarial networks," *arXiv preprint arXiv:1805.06350*, 2018.

[6] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional gan," *arXiv preprint arXiv:1807.00447*, 2018.

[7] F. Ait Aoudia and J. Hoydis, "End-to-end learning of communications systems without a channel model," *CoRR*, vol. abs/1804.02276, 2018.

[8] C. de Vrieze, S. Barratt, D. Tsai, and A. Sahai, "Cooperative multi-agent reinforcement learning for low-level wireless communication," *arXiv preprint arXiv:1801.04541*, Jan. 2018.

[9] V. Raj and S. Kalyani, "Backpropagating through the air: Deep learning at physical layer without channel models," *IEEE Commun. Lett.*, Aug. 2018.

[10] E. Agrell, "Database of sphere packing," https://codes.se/packings/8.htm, accessed: 2018-07-27.

[11] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Adv. Neural Inf. Process. Syst.*, vol. 12, 2000.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.