Model-free Reinforcement Learning of Semantic Communication by Stochastic Policy Gradient

Edgar Beck, Carsten Bockelmann and Armin Dekorsy
Department of Communications Engineering
University of Bremen, Bremen, Germany
Email: {beck, bockelmann, dekorsy}@ant.uni-bremen.de

***Abstract*—Motivated by the recent success of Machine Learning tools in wireless communications, the idea of semantic communication by Weaver from 1949 has gained attention. It breaks with Shannon's classic design paradigm by aiming to transmit the meaning, i.e., semantics, of a message instead of its exact version, allowing for information rate savings. In this work, we apply the Stochastic Policy Gradient (SPG) to design a semantic communication system by reinforcement learning, not requiring a known or differentiable channel model - a crucial step towards deployment in practice. Further, we motivate the use of SPG for both classic and semantic communication from the maximization of the mutual information between received and target variables. Numerical results show that our approach achieves comparable performance to a model-aware approach based on the reparametrization trick, albeit with a decreased convergence rate.**

***Index Terms*—Semantic communication, wireless networks, infomax, information bottleneck, machine learning, reinforcement learning, stochastic policy gradient, task-oriented.**

I. Introduction

To meet the unprecedented needs of 6G communication efficiency in terms of data rate, latency, and power, attention has been drawn to semantic communication [1]–[4]. It aims to transmit the meaning of a message rather than its exact version, which has been the main focus of digital error-free system design so far [1]. Bao, Basu et al. [5] were the first to define semantic information sources and channels to tackle the semantic design by conventional approaches arguing for the generality of Shannon's theory not only for the technical level but for semantic level design as Weaver [1].

Recently, inspired by [1], [5] and the rise of Machine Learning (ML) in communications research, transformer-based Deep Neural Networks (DNNs), have been introduced to Auto Encoders (AEs) for text transmission to learn compressed hidden representations of semantic content, aiming to improve communication efficiency [6]. In [7], the authors suggest using semantic similarity as the objective function: As most semantic metrics are non-differentiable, they propose a self-critic Reinforcement Learning (RL) solution. Both [6], [7] improve performance especially at low SNR compared to classical digital transmissions with [7] being slightly superior.

This paper builds on our idea from [4]: There, we define semantic communication as the data-reduced, reliable transmission of semantic sources and cast its design as an Information Bottleneck (IB) problem extending [5]. We apply our ML-based design SINFONY to a distributed multipoint scenario, communicating meaning from multiple image sources to a single receiver for semantic recovery. Numerical results show that SINFONY outperforms classical communication systems.

Semantic communication is a developing field with many survey papers aiming to provide interpretations (e.g., [2], [3]). It remains still unclear how the approaches proposed so far can be implemented in practice which motivates the main contribution of this article:

- We apply the Stochastic Policy Gradient (SPG) to design a semantic communication system, i.e., RL-SINFONY, by RL. By this means, we do not require a known or differentiable channel model - a crucial step towards deployment in practice.
- Further, we motivate the application of the SPG for both classic and semantic communication from maximization of the mutual information between received, and target variables compared to [8].
- At the time of writing, the RL-based approach in [7] was extended to handle non-differentiable channels. Our work distinguishes from [7] in using a different system model akin to task-oriented communication and not deriving our approach from a RL view. Further, the authors observed that training does not converge within their time limit to comparable results as the baseline approach in their setup for text transmission. We confirm the problem of slow convergence hinting at solution approaches and demonstrate feasibility in our distributed scenario.

In the following, we revisit our theoretical framework from [4] in Sec. II. For RL-based optimization, we introduce the SPG in Sec. III. Finally, in Sec. IV and V, we provide one numerical example for SINFONY application from [4] and summarize the main results, respectively.

II. A Framework for Semantics

A. Semantic System Model

1) Semantic Source and Channel: First, we define our information-theoretic system model of semantic communication shown in Fig. 1. Motivated by the approach of Bao, Basu et al. [5], we adopt the terminus of a semantic source as in [4] and describe it as a hidden target multivariate Random Variable (RV) $\mathbf{z} \in \mathcal{M}_z^{N_z \times 1}$ from domain $\mathcal{M}_z$ of dimension N_z distributed according to a probability density or mass function

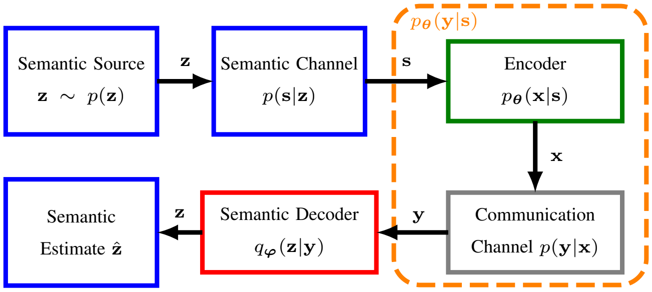

Semantic Source
$\mathbf{z} \sim p(\mathbf{z})$
$\mathbf{z}$
Semantic Channel
$p(\mathsf{s}|\mathbf{z})$
s
$p_{\boldsymbol{\theta}}(\mathbf{y}|\mathsf{s})$
Encoder
$p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$
$\mathbf{x}$
Semantic Estimate $\hat{\mathbf{z}}$
$\mathbf{z}$
Semantic Decoder
$q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$
$\mathbf{y}$
Communication Channel $p(\mathbf{y}|\mathbf{x})$

Fig. 1. Block diagram of the considered semantic system model.

(pdf/pmf) $p(\mathbf{z})$. To simplify the discussion, we assume it to be discrete and memoryless.[1]

Then, a semantic channel modeled by conditional distribution $p(\mathsf{s}|\mathbf{z})$ generates a source signal, a RV $\mathsf{s} \in \mathcal{M}_s^{N_s \times 1}$, that is usually observed and enters the communication system. Compared to [5][2], we consider probabilistic semantic channels $p(\mathsf{s}|\mathbf{z})$ using the definition from [4]. We refer the reader to [4] for an example of what these RVs may look like.

2) Semantic Channel Encoding: Our challenge is to encode the source s onto the transmit signal $\mathbf{x} \in \mathcal{M}_x^{N_{\text{Tx}} \times 1}$ (see Fig. 1) for efficient and reliable semantic transmission through the physical communication channel $p(\mathbf{y}|\mathbf{x})$, where $\mathbf{y} \in \mathcal{M}_y^{N_{\text{Rx}} \times 1}$ is the received signal vector, such that the semantic RV $\mathbf{z}$ at a recipient is best preserved [4]. We parametrize the encoder $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$ by a parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{N_{\boldsymbol{\theta}} \times 1}$ and assume $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$ to be deterministic in communications with $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s}) = \delta(\mathbf{x} - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathsf{s}))$. In summary, we bring the semantic source $\mathbf{z}$ to the context of communications by considering the complete Markov chain $\mathbf{z} \leftrightarrow \mathsf{s} \leftrightarrow \mathbf{x} \leftrightarrow \mathbf{y}$ in contrast to [5].

In classic Shannon design, the posterior $p_{\boldsymbol{\theta}}(\mathsf{s}|\mathbf{y})$ is processed to recover the source signal s as accurately as possible at the receiver side. Instead, we recover semantics $\mathbf{z}$ processing $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})$: Since the entropy $\mathcal{H}(\mathbf{z}) = \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z})}[-\ln p(\mathbf{z})]$[3] of the semantic RV $\mathbf{z}$ is expected to be less or equal to the entropy $\mathcal{H}(\mathsf{s})$ of the source s, i.e., $\mathcal{H}(\mathbf{z}) \leq \mathcal{H}(\mathsf{s})$, we can compress by transmitting the semantic RV $\mathbf{z}$.

B. Semantic Communication Design

Now, we revisit our two design approaches from [4].

1) Infomax Principle: First, we like to find the encoder $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$ that maps s to a representation $\mathbf{y}$ such that most information of the relevant RV $\mathbf{z}$ is included in $\mathbf{y}$, i.e., we maximize the Mutual Information (MI) $I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y})$ w.r.t. $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$:

$$\begin{aligned}
&\arg\max_{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})} I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y}) && (1)\\
=&\arg\max_{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})} \mathrm{E}_{\mathbf{z},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{y})}\left[\ln \frac{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{y})}{p(\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{y})}\right] && (2)\\
=&\arg\max_{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})} \mathcal{H}(\mathbf{z}) - \mathcal{H}(p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{y}), p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})) && (3)\\
=&\arg\max_{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})} \mathrm{E}_{\mathbf{z},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{y})}[\ln p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})]\,. && (4)
\end{aligned}$$

There, $\mathcal{H}(p(\mathbf{x}), q(\mathbf{x})) = \mathrm{E}_{\mathbf{x}\sim p(\mathbf{x})}[-\ln q(\mathbf{x})]$ is the cross entropy between two pdfs/pmfs $p(\mathbf{x})$ and $q(\mathbf{x})$.

If the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y})$ in (4) is intractable to compute, we can replace it with a variational distribution $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ with parameters $\boldsymbol{\varphi}$, i.e., the semantic decoder in Fig. 1. Then, we can define a MI Lower BOund (MILBO) [4]:

$$\begin{aligned}
I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y}) &\geq \mathrm{E}_{\mathbf{z},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{y})}[\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})] && (5)\\
&= -\mathrm{E}_{\mathbf{y}\sim p(\mathbf{y})}[\mathcal{H}(p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{y}), q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y}))] && (6)\\
&= -\mathcal{L}^{\text{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}\,. && (7)
\end{aligned}$$

Now, we can learn optimal parametrizations $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ of the transmitter discriminative model $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$ and of the variational receiver posterior $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ by minimizing the amortized cross entropy $\mathcal{L}^{\text{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}$ in (6), i.e., marginalized across observations $\mathbf{y}$ [4]. The encoder can be seen by rewriting:

$$\begin{aligned}
\mathcal{L}^{\text{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}} &= \mathrm{E}_{\mathsf{s},\mathbf{x},\mathbf{y},\mathbf{z}\sim p_{\boldsymbol{\theta}}(\mathsf{s},\mathbf{x},\mathbf{y},\mathbf{z})}[-\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})] && (8)\\
&= \mathrm{E}_{\mathsf{s},\mathbf{z}\sim p(\mathsf{s},\mathbf{z})}\left[\mathrm{E}_{\mathbf{x}\sim p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})}\left[\mathrm{E}_{\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})}[-\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})]\right]\right]\,.
\end{aligned}$$

The idea is to solve (8) by AEs or - in this article - RL. Thus, we use DNNs for the design of both encoder $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})$ and decoder $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ [6]. Note that in our semantic problem (1) or (8), we do not auto encode the hidden $\mathbf{z}$ or s as in [6] itself, but encode s to obtain $\mathbf{z}$ by decoding. This means our interpretation of semantic information and its recovery deviates from literature: We define semantics $\mathbf{z}$ explicitly compared to, e.g., [6], that optimizes on s and then measures semantic similarity w.r.t. its estimate $\hat{\mathsf{s}}$ explicitly by some semantic metric $\mathcal{L}(\mathsf{s},\hat{\mathsf{s}})$.

2) Information Bottleneck View: Further, introducing a constraint on the information rate in (1), we can formulate an Information Bottleneck (IB) optimization problem [2], where we like to maximize the relevant information $I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y})$ subject to the constraint to limit the compression rate $I_{\boldsymbol{\theta}}(\mathsf{s};\mathbf{x})$ to a maximum information rate I_{C}:

$$\arg\max_{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathsf{s})} I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y}) \quad \text{s.t.} \quad I_{\boldsymbol{\theta}}(\mathsf{s};\mathbf{x}) \leq I_{\text{C}}\,. \tag{9}$$

In this article, we set constraint I_{C} by fixing N_{Tx} since then an upper bound on $I_{\boldsymbol{\theta}}(\mathsf{s};\mathbf{x})$ grows for discrete RVs [4]: $I_{\boldsymbol{\theta}}(\mathsf{s};\mathbf{x}) \leq \sum_{n=1}^{N_{\text{Tx}}} \mathcal{H}(x_n) = I_{\text{C}}$. With fixed constraint I_{C}, we then need to maximize the relevant information $I_{\boldsymbol{\theta}}(\mathbf{z};\mathbf{y})$. As in the infomax problem, we can exploit the MILBO to use the amortized cross entropy $\mathcal{L}^{\text{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}$ in (8) as the optimization criterion.

[1] For the remainder of the article, note that the domain of all RVs $\mathcal{M}$ may be either discrete or continuous. Further, we note that the definition of entropy for discrete and continuous RVs differs. For example, the differential entropy of continuous RVs may be negative whereas the entropy of discrete RVs is always positive [9]. Without loss of generality, we will thus assume all RVs either to be discrete or to be continuous. In this work, we avoid notational clutter by using the expected value operator: Replacing the integral by summation over discrete RVs, the equations are also valid for discrete RVs and vice versa.

[2] In [5], the semantic channel is the transmission system.

[3] There, $\mathrm{E}_{\mathbf{x}\sim p(\mathbf{x})}[f(\mathbf{x})]$ denotes the expected value of $f(\mathbf{x})$ w.r.t. both discrete or continuous RVs $\mathbf{x}$.

III. Stochastic Policy Gradient-based Reinforcement Learning

If calculating the expected value of the amortized cross entropy $\mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}$ in (8) is analytically or computationally intractable as typical with DNNs, we can approximate it using Monte Carlo sampling techniques with N samples $\{\mathbf{z}_i, \mathbf{s}_i, \mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N}$. Then, our loss function (8) amounts to

$$\mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}} \approx -\frac{1}{N}\sum_{i=1}^{N} \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)\,. \tag{10}$$

A. Stochastic Gradient Descent-based Optimization

For Stochastic Gradient Descent (SGD) - based optimization, the gradient w.r.t. $\boldsymbol{\varphi}$ can then be calculated by

$$\frac{\partial \mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}}{\partial \boldsymbol{\varphi}} = -\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})p(\mathbf{s}|\mathbf{z})p(\mathbf{z})}\left[\frac{\partial \ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})}{\partial \boldsymbol{\varphi}}\right] \tag{11}$$

$$\approx -\,\frac{1}{N}\sum_{i=1}^{N}\frac{\partial \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)}{\partial \boldsymbol{\varphi}} \tag{12}$$

and by application of the backpropagation algorithm in Automatic Differentiation Frameworks (ADF), e.g., TensorFlow or PyTorch.

1) Reinforce Gradient: Computation of the gradient w.r.t. $\boldsymbol{\theta}$ is not straightforward since we sample w.r.t. the distribution $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$ dependent on $\boldsymbol{\theta}$ [9]. Assuming continuous-valued $\mathbf{y}$ and using the log-trick $\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}} = \frac{\partial p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}} / p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$, we can derive:

$$\frac{\partial \mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}}{\partial \boldsymbol{\theta}}$$

$$= -\,\frac{\partial}{\partial \boldsymbol{\theta}}\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})p(\mathbf{s},\mathbf{z})}[\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})] \tag{13}$$

$$= -\,\mathrm{E}_{\mathbf{z},\mathbf{s}\sim p(\mathbf{s},\mathbf{z})}\left[\int_{\mathcal{M}_y^{N_{\mathrm{Rx}}}} \underbrace{\frac{\partial p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}}}_{=p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})\cdot\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}}} \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})\ \mathrm{d}\mathbf{y}\right] \tag{14}$$

$$= -\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})p(\mathbf{s},\mathbf{z})}\left[\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}} \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})\right] \tag{15}$$

$$\approx -\,\frac{1}{N}\sum_{i=1}^{N}\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{s}_i)}{\partial \boldsymbol{\theta}} \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)\,. \tag{16}$$

We arrive at the same result with discrete RVs $\mathbf{y}$ replacing the integral in (14) by a sum. The Monte Carlo approximation (15) is the REINFORCE gradient w.r.t. $\boldsymbol{\theta}$ [9]. This estimate has high variance since we sample w.r.t. the distribution $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$ dependent on $\boldsymbol{\theta}$.

2) Reparametrization Trick: Leveraging the direct relationship between $\boldsymbol{\theta}$ and $\mathbf{y}$ in $\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ can help reduce the estimator's high variance compared to (15). Typically, e.g., in Variational AEs (VAE), the reparametrization trick is used to achieve this [9]. Here we can apply it if we can decompose the latent variable $\mathbf{y} \sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$ into a differentiable function $\mathbf{y} = f_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{n})$ and a RV $\mathbf{n} \sim p(\mathbf{n})$ independent of $\boldsymbol{\theta}$. Fortunately, the typical forward model of a communication system $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$ fulfills this criterion. Assuming a deterministic DNN encoder $\mathbf{x} = \mu_{\boldsymbol{\theta}}(\mathbf{s})$ and additive noise $\mathbf{n}$ with covariance $\boldsymbol{\Sigma}$, we can thus rewrite $\mathbf{y}$ into $f_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{n}) = \mu_{\boldsymbol{\theta}}(\mathbf{s}) + \boldsymbol{\Sigma}^{1/2} \cdot \mathbf{n}$ and accordingly the amortized cross entropy gradient into:

$$\frac{\partial \mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}}{\partial \boldsymbol{\theta}} = -\,\frac{\partial}{\partial \boldsymbol{\theta}}\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})p(\mathbf{s},\mathbf{z})}[\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})] \tag{17}$$

$$= -\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{n}\sim p(\mathbf{n})p(\mathbf{s}|\mathbf{z})p(\mathbf{z})}\left[\frac{\partial f_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{n})}{\partial \boldsymbol{\theta}} \cdot \frac{\partial \ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})}{\partial \mathbf{y}}\right] \tag{18}$$

$$\approx -\,\frac{1}{N}\sum_{i=1}^{N}\frac{\partial f_{\boldsymbol{\theta}}(\mathbf{s}_i,\mathbf{n}_i)}{\partial \boldsymbol{\theta}} \cdot \left.\frac{\partial \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)}{\partial \mathbf{y}}\right|_{\mathbf{y}=f_{\boldsymbol{\theta}}(\mathbf{n}_i,\mathbf{s}_i)}\,. \tag{19}$$

The trick can be easily implemented in ADFs by adding a noise layer after (DNN) function $\mathbf{x} = \mu_{\boldsymbol{\theta}}(\mathbf{s})$, typically used for regularization in ML literature. This allows for joint optimization of both $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$, as demonstrated in recent works [10], treating unsupervised optimization of AEs as a supervised learning problem.

B. Stochastic Policy Gradient

We note that unsupervised learning of encoder and decoder with both gradients (16) or (19) requires a known and fully differentiable forward model $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})$. But the gradient

$$\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{s})}{\partial \boldsymbol{\theta}} = \frac{\partial \mu_{\boldsymbol{\theta}}(\mathbf{s})}{\partial \boldsymbol{\theta}} \cdot \frac{\partial p(\mathbf{y}|\mathbf{x})}{\partial \mathbf{x}} \cdot \frac{\partial \ln p(\mathbf{y}|\mathbf{x})}{\partial p(\mathbf{y}|\mathbf{x})} \tag{20}$$

with deterministic encoder $\mathbf{x} = \mu_{\boldsymbol{\theta}}(\mathbf{s})$ may not be computable, as the channel model $p(\mathbf{y}|\mathbf{x})$ could be non-differentiable or unknown without any channel estimate. Further, in practice, the transmitter and receiver are separated at different locations and have at most a rudimentary feedback link, requiring independent optimization w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$: The transmitter does not know $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ and the receiver $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})$, vice versa.

To tackle these challenges in gradient computation, we now introduce a stochastic policy $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s}) \neq \delta(\mathbf{x} - \mu_{\boldsymbol{\theta}}(\mathbf{s}))$ that fulfills the reparametrization property:

$$\frac{\partial \mathcal{L}^{\mathrm{CE}}_{\boldsymbol{\theta},\boldsymbol{\varphi}}}{\partial \boldsymbol{\theta}} = -\,\frac{\partial}{\partial \boldsymbol{\theta}}\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{x},\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})p(\mathbf{s},\mathbf{z})}[\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})] \tag{21}$$

$$= -\,\mathrm{E}_{\mathbf{z},\mathbf{s}\sim p(\mathbf{s},\mathbf{z})}\left[\int_{\mathcal{M}_x^{N_{\mathrm{Tx}}}} \underbrace{\frac{\partial p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})}{\partial \boldsymbol{\theta}}}_{=p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})\cdot\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})}{\partial \boldsymbol{\theta}}} \cdot \mathrm{E}_{\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})}[\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})]\ \mathrm{d}\mathbf{x}\right] \tag{22}$$

$$= -\,\mathrm{E}_{\mathbf{z},\mathbf{s},\mathbf{x},\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{s},\mathbf{x},\mathbf{y})}\left[\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})}{\partial \boldsymbol{\theta}} \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})\right] \tag{23}$$

$$\approx -\,\frac{1}{N}\sum_{i=1}^{N}\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{x}_i|\mathbf{s}_i)}{\partial \boldsymbol{\theta}} \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)\,. \tag{24}$$

Again the log-trick is applied in (22) to arrive in (23) and the results hold for discrete RVs $\mathbf{x}$. Most importantly, (23) is the policy gradient and the derivation is equivalent to the Stochastic Policy Gradient (SPG) theorem, a fundamental result of continuous-action RL [11]. For integration in ADFs, usually, an objective function whose gradient is the Monte Carlo policy gradient estimator of (23), i.e., the REINFORCE gradient (24), is constructed:

$$\mathcal{L}_{\boldsymbol{\theta}}^{\text{SPG}} = -\frac{1}{N}\sum_{i=1}^{N} \ln p_{\boldsymbol{\theta}}(\mathbf{x}_i|\mathbf{s}_i) \cdot \ln q_{\boldsymbol{\varphi}}(\mathbf{z}_i|\mathbf{y}_i)\,. \tag{25}$$

With objective (25) or REINFORCE gradient (24), we can finally optimize $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\varphi}}^{\text{CE}}$ w.r.t. $\boldsymbol{\theta}$, since we are able to sample $\{\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y}\} \sim p_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y})$ and compute $\frac{\partial \ln p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})}{\partial \boldsymbol{\theta}}$ at the transmitter and $\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ at the receiver. Note that $\mathbf{s}$ and $\mathbf{x}$ only have to be known at the transmitter and both $\mathbf{z}$ and $\mathbf{y}$ at the receiver, respectively. This means an a priori known pilot/training sequence $\mathbf{s}_{\text{train}}$, $\mathbf{z}_{\text{train}}$ is required.

Moreover, we require a feedback link to transmit $\ln q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ evaluated for $\mathbf{z}_{\text{train}}$ and $\mathbf{y}_{\text{train}}$ to the encoder. The term can be interpreted as a reward or critic known from RL [11]. Accordingly, the transmitter can be seen as an actor with a policy $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{s})$. The best continuous action/policy is then learned by optimization w.r.t. these rewards.

1) Stochastic Policy: Introducing a stochastic policy means we need to add a probabilistic sampler/explorer function $p(\mathbf{x}|\bar{\mathbf{x}})$ to the encoder as shown in Fig. 2. In this article, we use a Gaussian policy, i.e., a multivariate Gaussian pdf

$$p(\mathbf{x}|\bar{\mathbf{x}}) = \mathcal{N}\left((1-\sigma_{\text{exp}}^2)^{1/2} \cdot \bar{\mathbf{x}}, \sigma_{\text{exp}}^2 \cdot \mathbf{I}\right) \tag{26}$$

with exploration variance $\sigma_{\text{exp}}^2 \in (0, 1)$ where scaling of the mean $\bar{\mathbf{x}} = \mu_{\boldsymbol{\theta}}(\mathbf{s})$ is done to ensure the conservation of average energy. For $\sigma_{\text{exp}}^2 \to 0$, $p(\mathbf{x}|\bar{\mathbf{x}})$ approaches a deterministic policy. In [8], the authors show that the true channel gradient $\frac{\partial}{\partial \mathbf{x}} p(\mathbf{y}|\mathbf{x})$ is then perfectly approximated. However, using a near-deterministic policy leads in their experiments to high variance of the gradient estimate (24) resulting in slow convergence. To compensate for this effect, we require a much larger and computationally expensive batch size $N = N_{\text{b}}$. From the view of RL, using a stochastic policy $\sigma_{\text{exp}}^2 \neq 0$ enables the exploration of the set of possible actions.

C. Alternating RL-based Training

After introducing the SPG, we now derive an optimization procedure akin to [8] for the whole semantic communication system, i.e., encoder and decoder. It does not require any channel model but a fixed training/pilot sequence and a feedback link. We show it in Fig. 2:

1) We note that according to (12) decoder optimization reduces to supervised learning w.r.t. $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\varphi}}^{\text{CE}}$ and $\boldsymbol{\varphi}$ at the receiver side. Thus, in the first step, we train the decoder based on the training sequence and updated encoder, but without sampler/explorer ($\sigma_{\text{exp}}^2 = 0$).
2) Second, the encoder explores with transmit signals $\mathbf{x}_{\text{train}}$. It is optimized based on the policy gradient of $\mathcal{L}_{\boldsymbol{\theta}}^{\text{SPG}}$ and the reward $\ln q_{\boldsymbol{\varphi}}(\mathbf{z}_{\text{train}}|\mathbf{y}_{\text{train}})$ that the decoder feeds back.
3) We alternate between the first and second training steps until convergence. Note that we can use one or multiple SGD steps and batches for each alternating training step, respectively.

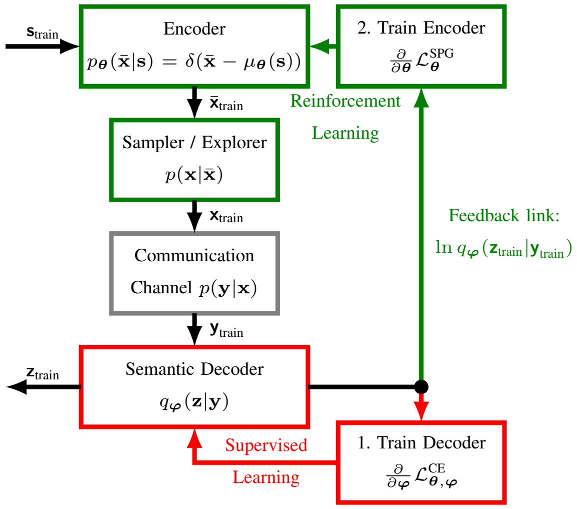

$\mathbf{s}_{\text{train}}$
Encoder
$p_{\boldsymbol{\theta}}(\bar{\mathbf{x}}|\mathbf{s}) = \delta(\bar{\mathbf{x}} - \mu_{\boldsymbol{\theta}}(\mathbf{s}))$
2. Train Encoder
$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{\theta}}^{\text{SPG}}$
$\bar{\mathbf{x}}_{\text{train}}$
Reinforcement
Learning
Sampler / Explorer
$p(\mathbf{x}|\bar{\mathbf{x}})$
$\mathbf{x}_{\text{train}}$
Feedback link:
$\ln q_{\boldsymbol{\varphi}}(\mathbf{z}_{\text{train}}|\mathbf{y}_{\text{train}})$
Communication
Channel $p(\mathbf{y}|\mathbf{x})$
$\mathbf{y}_{\text{train}}$
$\mathbf{z}_{\text{train}}$
Semantic Decoder
$q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$
Supervised
Learning
1. Train Decoder
$\frac{\partial}{\partial \boldsymbol{\varphi}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\varphi}}^{\text{CE}}$

Fig. 2. Optimization procedure of a semantic encoder and decoder without a differentiable channel model: 1. Train the decoder supervised based on the training sequence and updated encoder but without sampler. 2. Encoder explores transmit signals $\mathbf{x}_{\text{train}}$ and improves its policy according to the decoder reward feedback. 3. Alternate between both steps until convergence.

Reminiscent of the RL fashion of the stochastic policy optimization of Semantic INFOrmation traNsmission and recoverY [4], we name this approach RL-SINFONY.

IV. Example of Model-free Semantic Recovery

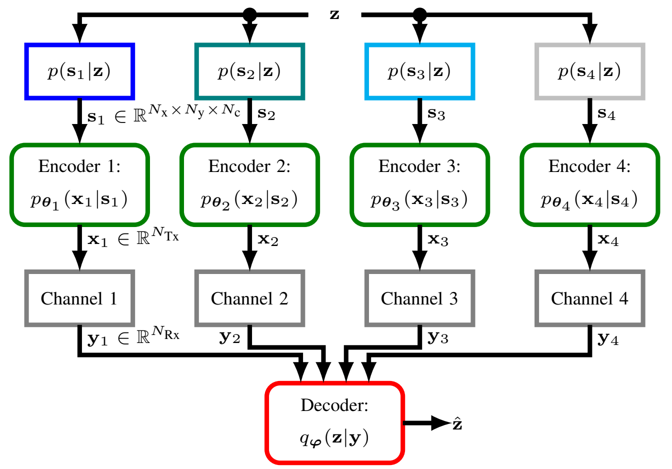
To evaluate the proposed model-free optimization approach RL-SINFONY, we use the numerical example of distributed image classification with SINFONY from [4] shown in Fig. 3. Thus, we will now assume the hidden semantic RV to be a one-hot vector $\mathbf{z} \in \{0, 1\}^{M\times 1}$ representing one of M image classes. Then, each of the four agents observes its image, i.e., the source signal $\mathbf{s}_i \sim p(\mathbf{s}_i|\mathbf{z})$ with $i = 1, \ldots, 4$, through a semantic channel, being generated by the same semantic RV $\mathbf{z}$ and thus belonging to the same class. Based on these images, a central unit shall extract semantics, i.e., perform classification.

We propose to optimize the four encoders $p_{\boldsymbol{\theta}_i}(\mathbf{x}_i|\mathbf{s}_i)$ **jointly** with a decoder $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4]^T)$ w.r.t. cross entropy (8) of the semantic labels (see Fig. 3). Hence, we maximize the system's overall semantic measure, i.e., classification accuracy.

To show the basic working principle and ease implementation, we use the grayscale MNIST and colored CIFAR10 datasets with $M = 10$ image classes [4]. We assume that the semantic channel generates an image that we divide into four equally sized quadrants and each agent observes one quadrant $\mathbf{s}_i \in \mathbb{R}^{N_{\text{x}}\times N_{\text{y}}\times N_{\text{c}}}$ where N_{x} and N_{y} is the number of image pixels in the x- and y-dimension, respectively, and N_{c} is the color channel number.

A. Distributed SINFONY Approach

For the design of SINFONY, we rely on the powerful DNN approach ResNet for feature extraction [4]. We use the



$\mathbf{z}$
$p(\mathbf{s}_1|\mathbf{z})$
$p(\mathbf{s}_2|\mathbf{z})$
$p(\mathbf{s}_3|\mathbf{z})$
$p(\mathbf{s}_4|\mathbf{z})$
$\mathbf{s}_1 \in \mathbb{R}^{N_x \times N_y \times N_c}$
$\mathbf{s}_2$
$\mathbf{s}_3$
$\mathbf{s}_4$
Encoder 1: $p_{\boldsymbol{\theta}_1}(\mathbf{x}_1|\mathbf{s}_1)$
Encoder 2: $p_{\boldsymbol{\theta}_2}(\mathbf{x}_2|\mathbf{s}_2)$
Encoder 3: $p_{\boldsymbol{\theta}_3}(\mathbf{x}_3|\mathbf{s}_3)$
Encoder 4: $p_{\boldsymbol{\theta}_4}(\mathbf{x}_4|\mathbf{s}_4)$
$\mathbf{x}_1 \in \mathbb{R}^{N_{\mathrm{Tx}}}$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
Channel 1
Channel 2
Channel 3
Channel 4
$\mathbf{y}_1 \in \mathbb{R}^{N_{\mathrm{Rx}}}$
$\mathbf{y}_2$
$\mathbf{y}_3$
$\mathbf{y}_4$
Decoder: $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$
$\hat{\mathbf{z}}$

Fig. 3. RL-SINFONY for distributed agents. Four agents extract features for bandwidth-efficient transmission. Based on the received signals, the decoder extracts semantics.

TABLE I
RL-SINFONY - DNN ARCHITECTURE FOR IMAGE EXAMPLE.

Component	Layer	Dimension
Input	Image (MNIST, CIFAR10)	$(14, 14, 1)$, $(16, 16, 3)$
4×	Conv2D	$(14, 14, 14)$, $(16, 16, 16)$
Feature	ResNetBlock (2/3 res. un.)	$(14, 14, 14)$, $(16, 16, 16)$
Extractor	ResNetBlock (2/3 res. un.)	$(7, 7, 28)$, $(8, 8, 32)$
	ResNetBlock (2/3 res. un.)	$(4, 4, 56)$, $(4, 4, 64)$
	Batch Normalization	$(4, 4, 56)$, $(4, 4, 64)$
	ReLU activation	$(4, 4, 56)$, $(4, 4, 64)$
	GlobalAvgPool2D	(56), (64)
4× Tx	ReLU	N_{Tx}
	Linear	N_{Tx}
	Normalization (dim.)	N_{Tx}
4× Sampler	AWGN + Normalization	N_{Tx}
4× Channel	AWGN	N_{Tx}
Rx	ReLU (4× shared)	$(2, 2, N_{\mathrm{w}})$
	GlobalAvgPool2D	N_{w}
Classifier	Softmax	$M = 10$

pre-activation version of ResNet without bottlenecks implemented for CIFAR10 classification. In Tab. I, we show its structure modified for the distributed scenario from Fig. 3. There, ResNetBlock is the basic building block of the ResNet architecture. Each block consists of multiple residual units (res. un.) and we use 2 for the MNIST and 3 for the CIFAR10 dataset. For further implementation details, we refer the reader to the original work and our source code [12].

Our key idea here is to modify ResNet w.r.t. the communication task by splitting it where a low-bandwidth representation of semantic information is present. Therefore, we aim to transmit each agent's local features of length N_{Feat} provided by the Feature Extractors in Tab. I instead of all sub-images $\mathbf{s}_i$ and add the component Tx to encode the features into $\mathbf{x}_i \in \mathbb{R}^{N_{\mathrm{Tx}} \times 1}$ for transmission through the wireless channel (see Fig. 3). We note that $\mathbf{x}_i \in \mathbb{R}^{N_{\mathrm{Tx}} \times 1}$ is analog and that the output dimension N_{Tx} defines the number of channel uses per agent and thus information rate. To limit the transmit power to one, we constrain the Tx Linear layer output by the norm along the training batch or the encode vector dimension (dim.). For RL-SINFONY, we add a Gaussian Sampler (26) after the Tx output compared to [4].

At the receiver side, we use a single Rx module only with shared DNN layers of width N_{w} and parameters $\boldsymbol{\varphi}_{\mathrm{Rx}}$ for all inputs $\mathbf{y}_i$ [4]. Based on an aggregation of the four Rx outputs, a softmax layer with $M = 10$ units finally computes class probabilities $q_{\boldsymbol{\varphi}}(\mathbf{z}|\mathbf{y})$ whose maximum is the maximum a posteriori estimate $\hat{\mathbf{z}}$.

B. Optimization Details

We evaluate RL-SINFONY in TensorFlow 2 on the MNIST and CIFAR10 datasets [12]. For cross-entropy loss minimization, we use the gradient approximations from III and the SGD-variant Adam with a batch size of $N_{\mathrm{b}} = 500$. We add l_2-regularization with a weight decay of 0.0001. To optimize the transceiver for a wider SNR range, we choose the SNR to be uniformly distributed within $[-4, 6]$ dB where SNR $= 1/\sigma_{\mathrm{n}}^2$ with noise variance σ_{n}^2. We set $N_{\mathrm{w}} = N_{\mathrm{Feat}}$ as default and refer to [4], [12] for more implementation details. In the following, we compare the performance of:

- **SINFONY:** The distributed SINFONY design from [4] trained model-aware as one DNN with channel noise layer using the reparametrization trick (19) to approximate the gradients. We train for $N_{\mathrm{e}} = 100$ epochs with the MNIST dataset.
- **RL-SINFONY:** New approach trained model-free via RL as shown in Fig. 2 using SPG (24). We alternate between 10 decoder and encoder optimization steps. Note that one decoder and encoder step amounts to one iteration of the model-aware approach where the encoder and decoder are optimized jointly. Hence, for a fair comparison, we divide the number of alternating iterations or epochs N_{e} of the SPG approach by 2. We choose $N_{\mathrm{e}} = 3000$ and add $N_{\mathrm{e,rx}} = 600$ epochs of receiver fine-tuning at the end [8]. To decrease the SPG estimator variance, we choose a rather high exploration variance $\sigma_{\mathrm{exp}}^2 = 0.15$.
- **perfect comm.:** SINFONY trained with perfect communication links without Tx and Rx modules, but with Tx normalization. Thus, the plain power-constrained features are transmitted with $N_{\mathrm{Tx}} = 56$ or 64 channel uses. It serves as the benchmark, as it indicates the maximum performance of the distributed design.
- **Tx/Rx N_{Tx}:** Default SINFONY from Tab. I trained with Tx and Rx module and N_{Tx} channel uses.

C. Numerical Results

To measure semantic transmission quality, we use classification error rate on semantic RV $\mathbf{z}$.

1) MNIST dataset: The numerical results of our proposed approach RL-SINFONY on the MNIST validation dataset are shown in Fig. 4. We observe that both approaches RL-SINFONY and SINFONY with Tx/Rx module approach the

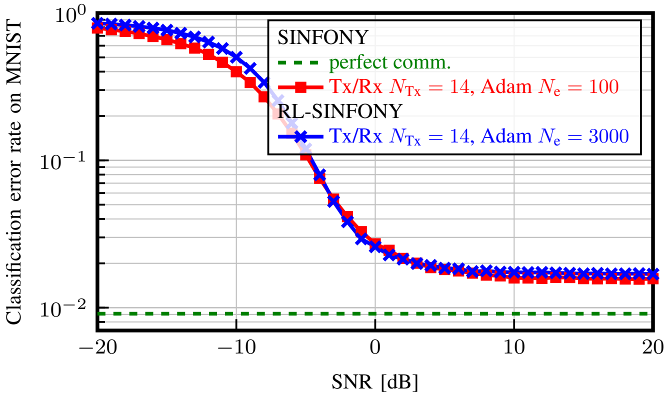


Fig. 4. Comparison of the classification error rate of RL-SINFONY and SINFONY with $N_{\text{Tx}} = 14$ on MNIST as a function of SNR.

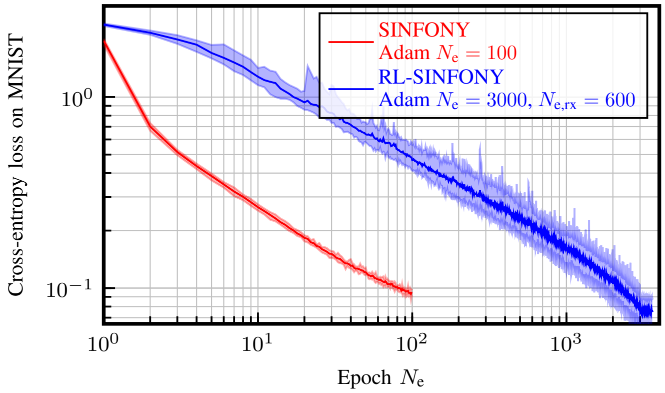


Fig. 5. Comparison of training convergence between RL-SINFONY and SINFONY with $N_{\text{Tx}} = 14$ in terms of the cross-entropy loss on MNIST averaged over 10 runs as a function of training epochs N_{e}.

benchmark with ideal links (SINFONY - perfect comm.) at high SNR. Notably, both curves are very close to each other, i.e., the performance gap after training is minor. This means training of RL-SINFONY converged successfully.

2) Convergence Rate: Since the number of training epochs required to achieve the same performance deviates significantly with $N_{\text{e}} + N_{\text{e,rx}} = 3000 + 600 = 3600$ compared to $N_{\text{e}} = 100$, we take a closer look at training convergence in terms of the cross-entropy loss shown in Fig. 5. We averaged the loss over 10 training runs and illustrate the interval between the maximum and minimum loss value using shaded areas. To reach the same loss, we require more than 10 times more epochs with RL-SINFONY compared to SINFONY. The reason for the decreased convergence is the increased variance of the REINFORCE gradient (24) compared to the reparametrization trick gradient (19). Also, we attribute the increased variance in training losses (blue-shaded area) to it.

3) CIFAR10 dataset and convergence issues: Trying various hyperparameter settings, we found that training of RL-SINFONY with $N_{\text{Tx}} = 16$ on the CIFAR10 dataset converges slowly. Using SGD with high $N_{\text{b}} = 512$, $\sigma_{\text{exp}}^2 = 0.15$ and learning rate $\epsilon = 10^{-4}$, we achieved a maximum validation accuracy of 50% at high SNR after $N_{\text{e}}+N_{\text{Rx}} = 5000+1000 = 6000$ epochs compared to 80% with SINFONY after $N_{\text{e}} = 200$ epochs [4]. The loss continued to decrease even after reaching our computation time limit of one day. We believe this is due to the high variance of the REINFORCE gradient (24), which increases by decreasing σ_{exp}^2 and increasing the continuous output space N_{Tx} of $\mathbf{x}$. Training with the more challenging CIFAR10 dataset may require more accurate gradient estimates compared to MNIST. Thus, we suggest exploring variance-reduction techniques in future work [9], [13].

V. Conclusion

In this work, we expanded on our previous idea from [4] by introducing the Stochastic Policy Gradient (SPG): We designed a semantic communication system via reinforcement learning, not requiring a known or differentiable channel model - a crucial step towards deployment in practice. Further, we motivated the use of the SPG for both classic and semantic communication from the maximization of the mutual information between received and target variables. Numerical results show that our approach achieves comparable performance to a model-aware approach, albeit at the cost of a decreased convergence rate by at least a factor of 10. It remains the question of how to improve the convergence rate with more challenging datasets.

References

[1] C. Shannon and W. Weaver, *The Mathematical Theory of Communication*, 16th ed. The University of Illinois Press, Sep. 1949.

[2] E. C. Strinati and S. Barbarossa, “6G networks: Beyond Shannon towards semantic and goal-oriented communications,” *Computer Networks*, vol. 190, p. 107930, May 2021.

[3] D. Gündüz, Z. Qin, I. E. Aguerri, H. S. Dhillon, Z. Yang, A. Yener, K. K. Wong, and C.-B. Chae, “Beyond Transmitting Bits: Context, Semantics, and Task-Oriented Communications,” *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 1, pp. 5–41, Jan. 2023.

[4] E. Beck, C. Bockelmann, and A. Dekorsy, “Semantic Information Recovery in Wireless Networks,” Mar. 2023. [Online]. Available: https://arxiv.org/abs/2204.13366

[5] J. Bao, P. Basu, M. Dean, C. Partridge, A. Swami, W. Leland, and J. A. Hendler, “Towards a theory of semantic communication,” in *2011 IEEE Network Science Workshop (NSW)*, West Point, NY, USA, Jun. 2011, pp. 110–117.

[6] H. Xie, Z. Qin, G. Y. Li, and B.-H. Juang, “Deep Learning Enabled Semantic Communication Systems,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 2663–2675, 2021.

[7] K. Lu, R. Li, X. Chen, Z. Zhao, and H. Zhang, “Reinforcement Learning-powered Semantic Communication via Semantic Similarity,” Apr. 2022. [Online]. Available: https://arxiv.org/abs/2108.12121

[8] F. A. Aoudia and J. Hoydis, “Model-Free Training of End-to-End Communication Systems,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 11, pp. 2503–2516, Nov. 2019.

[9] O. Simeone, “A Brief Introduction to Machine Learning for Engineers,” *Foundations and Trends® in Signal Processing*, vol. 12, no. 3-4, pp. 200–431, Aug. 2018.

[10] 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.

[11] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, “Deterministic Policy Gradient Algorithms,” in *31st International Conference on Machine Learning (PMLR)*, Jan. 2014, pp. 387–395.

[12] E. Beck, “Semantic Information Transmission and Recovery,” Apr. 2023. [Online]. Available: https://github.com/ant-uni-bremen/SINFONY

[13] E. Greensmith, P. L. Bartlett, and J. Baxter, “Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning,” *J. Mach. Learn. Res.*, vol. 5, pp. 1471–1530, Dec. 2004.