

# 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**—Following 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, separating transmitter and receiver, and 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 work was partly funded by the Federal State of Bremen and the University of Bremen as part of the Humans on Mars Initiative, by the German Ministry of Education and Research (BMBF) under grant 16KISK016 (Open6GHub), and by the German Research Foundation (DFG) under grant 500260669 (SCIL).

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 Semantic INformation TraNsmission and RecoverY (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: For a more in-depth survey, we refer the reader to, e.g., [2]–[4]. It remains still unclear how the approaches proposed so far can be implemented in practice which motivates the main contributions of this article:

- We apply the Stochastic Policy Gradient (SPG) to train a semantic communication system, i.e., RL-SINFONY, by RL. By this means, we separate transmitter and receiver, and do not require a known or differentiable channel model – a crucial step towards deployment in practice.
- Further, we derive the application of the SPG for both classic and semantic communication from maximization of the mutual information between target and received variables compared to [8].
- In particular, we investigate a task-oriented system model and a distributed application scenario with multiple sources and transmitters. By this means, our work distinguishes from the RL-based approach in [7] that was extended to handle non-differentiable channels at the time of writing.
- Further, the authors of [7] 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 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.

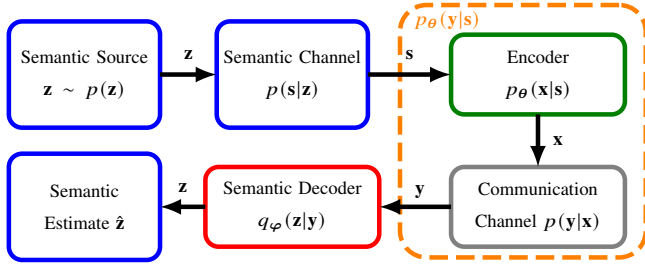


Fig. 1. Block diagram of the considered semantic system model.

## II. SEMANTIC COMMUNICATION FRAMEWORK

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

Then, a semantic channel modeled by conditional distribution  $p(\mathbf{s}|\mathbf{z})$  generates an observation or source signal, a RV  $\mathbf{s} \in \mathcal{M}_s^{N_s \times 1}$ , that enters the communication system. Compared to [5] where the semantic channel is the transmission system, we consider probabilistic semantic channels  $p(\mathbf{s}|\mathbf{z})$  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  $\mathbf{s}$  onto the transmit signal  $\mathbf{x} \in \mathcal{M}_x^{N_{Tx} \times 1}$  (see Fig. 1) for efficient and reliable semantic transmission through the physical communication channel  $p(\mathbf{y}|\mathbf{x})$ , where  $\mathbf{y} \in \mathcal{M}_y^{N_{Rx} \times 1}$  is the received signal vector, such that the semantic RV  $\mathbf{z}$  at a recipient is best preserved [4]. We parametrize the encoder  $p_\theta(\mathbf{x}|\mathbf{s})$  by a parameter vector  $\theta \in \mathbb{R}^{N_\theta \times 1}$  and assume  $p_\theta(\mathbf{x}|\mathbf{s})$  to be deterministic in communications with  $p_\theta(\mathbf{x}|\mathbf{s}) = \delta(\mathbf{x} - \mu_\theta(\mathbf{s}))$  and encoder function  $\mu_\theta(\mathbf{s})$ . In summary, we bring the semantic source  $\mathbf{z}$  to the context of communications by considering the complete Markov chain  $\mathbf{z} \leftrightarrow \mathbf{s} \leftrightarrow \mathbf{x} \leftrightarrow \mathbf{y}$  in contrast to [5].

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

<sup>1</sup>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.

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

### B. Semantic Communication Design

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

1) *InfoMax Principle*: First, we like to find the encoder  $p_\theta(\mathbf{x}|\mathbf{s})$  that maps  $\mathbf{s}$  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)  $I_\theta(\mathbf{z}; \mathbf{y})$  w.r.t.  $p_\theta(\mathbf{x}|\mathbf{s})$ :

$$\arg \max_{p_\theta(\mathbf{x}|\mathbf{s})} I_\theta(\mathbf{z}; \mathbf{y}) \quad (1)$$

$$= \arg \max_{p_\theta(\mathbf{x}|\mathbf{s})} \mathbb{E}_{\mathbf{z}, \mathbf{y} \sim p_\theta(\mathbf{z}, \mathbf{y})} \left[ \ln \frac{p_\theta(\mathbf{z}, \mathbf{y})}{p(\mathbf{z})p_\theta(\mathbf{y})} \right] \quad (2)$$

$$= \arg \max_{p_\theta(\mathbf{x}|\mathbf{s})} \mathcal{H}(\mathbf{z}) - \mathcal{H}(p_\theta(\mathbf{z}, \mathbf{y}), p_\theta(\mathbf{z}|\mathbf{y})) \quad (3)$$

$$= \arg \max_{p_\theta(\mathbf{x}|\mathbf{s})} \mathbb{E}_{\mathbf{z}, \mathbf{y} \sim p_\theta(\mathbf{z}, \mathbf{y})} [\ln p_\theta(\mathbf{z}|\mathbf{y})] \quad (4)$$

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

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

$$I_\theta(\mathbf{z}; \mathbf{y}) \geq \mathbb{E}_{\mathbf{z}, \mathbf{y} \sim p_\theta(\mathbf{z}, \mathbf{y})} [\ln q_\varphi(\mathbf{z}|\mathbf{y})] \quad (5)$$

$$= -\mathbb{E}_{\mathbf{y} \sim p(\mathbf{y})} [\mathcal{H}(p_\theta(\mathbf{z}|\mathbf{y}), q_\varphi(\mathbf{z}|\mathbf{y}))] \quad (6)$$

$$= -\mathcal{L}_{\theta, \varphi}^{\text{CE}} \quad (7)$$

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

$$\mathcal{L}_{\theta, \varphi}^{\text{CE}} = \mathbb{E}_{\mathbf{s}, \mathbf{x}, \mathbf{y}, \mathbf{z} \sim p_\theta(\mathbf{s}, \mathbf{x}, \mathbf{y}, \mathbf{z})} [-\ln q_\varphi(\mathbf{z}|\mathbf{y})] \quad (8)$$

$$= \mathbb{E}_{\mathbf{s}, \mathbf{z} \sim p(\mathbf{s}, \mathbf{z})} [\mathbb{E}_{\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{s})} [\mathbb{E}_{\mathbf{y} \sim p(\mathbf{y}|\mathbf{x})} [-\ln q_\varphi(\mathbf{z}|\mathbf{y})]]] \quad (9)$$

The idea is to solve (8) by AEs or – in this article – RL. Thus, we use DNNs for the design of both encoder  $p_\theta(\mathbf{x}|\mathbf{s})$  and decoder  $q_\varphi(\mathbf{z}|\mathbf{y})$  [6].

Note that in our semantic problem (1) or (8), we do not auto encode the hidden  $\mathbf{z}$  or  $\mathbf{s}$  as in [6] itself, but encode  $\mathbf{s}$  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  $\mathbf{s}$  and then measures semantic similarity w.r.t. its estimate  $\hat{\mathbf{s}}$  explicitly by some semantic metric  $\mathcal{L}(\mathbf{s}, \hat{\mathbf{s}})$ .

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  $I_\theta(\mathbf{z}; \mathbf{y})$  subject to the constraint to limit the compression rate  $I_\theta(\mathbf{s}; \mathbf{y})$  to a maximum information rate  $I_C$ :

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

In this article, we set constraint  $I_C$  by fixing  $N_{Tx}$  since then an upper bound on  $I_\theta(\mathbf{s}; \mathbf{y})$  grows as shown in [4]. With fixed constraint  $I_C$ , we then need to maximize the relevant information  $I_\theta(\mathbf{z}; \mathbf{y})$ . As in the InfoMax problem, we can exploit the MILBO to use the amortized cross entropy  $\mathcal{L}_{\theta, \varphi}^{CE}$  in (8) as the optimization criterion.

### III. STOCHASTIC POLICY GRADIENT-BASED REINFORCEMENT LEARNING

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

#### A. Stochastic Gradient Descent-based Optimization

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

$$\frac{\partial \mathcal{L}_{\theta, \varphi}^{CE}}{\partial \varphi} = -\mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{s})p(\mathbf{s}|\mathbf{z})p(\mathbf{z})} \left[ \frac{\partial \ln q_\varphi(\mathbf{z}|\mathbf{y})}{\partial \varphi} \right] \quad (10)$$

$$\approx \frac{1}{N} \sum_{i=1}^N \frac{\partial [-\ln q_\varphi(\mathbf{z}_i|\mathbf{y}_i)]}{\partial \varphi} \quad (11)$$

with  $N$  being equal to the batch size  $N_b$  and by application of the backpropagation algorithm in Automatic Differentiation Frameworks (ADF), e.g., TensorFlow or PyTorch.

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

$$\begin{aligned} \frac{\partial \mathcal{L}_{\theta, \varphi}^{CE}}{\partial \theta} &= -\frac{\partial}{\partial \theta} \mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{s})p(\mathbf{s}|\mathbf{z})} [\ln q_\varphi(\mathbf{z}|\mathbf{y})] \\ &= -\mathbb{E}_{\mathbf{z}, \mathbf{s} \sim p(\mathbf{s}, \mathbf{z})} \left[ \int_{\mathcal{M}_y^{N_{Rx}}} \underbrace{\frac{\partial p_\theta(\mathbf{y}|\mathbf{s})}{\partial \theta}}_{=p_\theta(\mathbf{y}|\mathbf{s}) \cdot \frac{\partial \ln p_\theta(\mathbf{y}|\mathbf{s})}{\partial \theta}} \cdot \ln q_\varphi(\mathbf{z}|\mathbf{y}) d\mathbf{y} \right] \end{aligned} \quad (12)$$

$$= -\mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{s})p(\mathbf{s}|\mathbf{z})} \left[ \frac{\partial \ln p_\theta(\mathbf{y}|\mathbf{s})}{\partial \theta} \cdot \ln q_\varphi(\mathbf{z}|\mathbf{y}) \right] \quad (13)$$

$$\approx -\frac{1}{N} \sum_{i=1}^N \frac{\partial \ln p_\theta(\mathbf{y}_i|\mathbf{s}_i)}{\partial \theta} \cdot \ln q_\varphi(\mathbf{z}_i|\mathbf{y}_i). \quad (14)$$

$$\approx -\frac{1}{N} \sum_{i=1}^N \frac{\partial \ln p_\theta(\mathbf{y}_i|\mathbf{s}_i)}{\partial \theta} \cdot \ln q_\varphi(\mathbf{z}_i|\mathbf{y}_i). \quad (15)$$

We arrive at the same result with discrete RVs  $\mathbf{y}$  replacing the integral in (13) by a sum. The Monte Carlo approximation (15) is the REINFORCE gradient w.r.t.  $\theta$  [9]. This estimate has high variance since we sample w.r.t. the distribution  $p_\theta(\mathbf{y}|\mathbf{s})$  dependent on  $\theta$ .

2) *Reparametrization Trick*: Leveraging the direct relationship between  $\theta$  and  $\mathbf{y}$  in  $\ln q_\varphi(\mathbf{z}|\mathbf{y})$  can help reduce the estimator's high variance. 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  $\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{s})$  into a differentiable function  $\mathbf{y} = f_\theta(\mathbf{s}, \mathbf{n})$  and a RV  $\mathbf{n} \sim p(\mathbf{n})$  independent of  $\theta$ . Fortunately, the typical

forward model of a communication system  $p_\theta(\mathbf{y}|\mathbf{s})$  fulfills this criterion. Assuming a deterministic (DNN) encoder  $\mathbf{x} = \mu_\theta(\mathbf{s})$  and additive noise  $\mathbf{n}$  with covariance  $\Sigma$ , we can thus rewrite  $\mathbf{y}$  into  $f_\theta(\mathbf{s}, \mathbf{n}) = \mu_\theta(\mathbf{s}) + \Sigma^{1/2} \cdot \mathbf{n}$  and accordingly the amortized cross entropy gradient (12) into:

$$\frac{\partial \mathcal{L}_{\theta, \varphi}^{CE}}{\partial \theta} = -\mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{n} \sim p(\mathbf{n})p(\mathbf{s}|\mathbf{z})p(\mathbf{z})} \left[ \frac{\partial f_\theta(\mathbf{s}, \mathbf{n})}{\partial \theta} \cdot \frac{\partial \ln q_\varphi(\mathbf{z}|\mathbf{y})}{\partial \mathbf{y}} \right] \quad (16)$$

$$\approx -\frac{1}{N} \sum_{i=1}^N \frac{\partial f_\theta(\mathbf{s}_i, \mathbf{n}_i)}{\partial \theta} \cdot \frac{\partial \ln q_\varphi(\mathbf{z}_i|\mathbf{y})}{\partial \mathbf{y}} \Big|_{\mathbf{y}=f_\theta(\mathbf{s}_i, \mathbf{n}_i)} \quad (17)$$

The trick can be easily implemented in ADFs by adding a noise layer after function  $\mathbf{x} = \mu_\theta(\mathbf{s})$ , typically used for regularization in ML literature. Then, our loss function (8) is the empirical cross entropy:

$$\mathcal{L}_{\theta, \varphi}^{CE} \approx -\frac{1}{N} \sum_{i=1}^N \ln q_\varphi(\mathbf{z}_i|\mathbf{y}_i = f_\theta(\mathbf{s}_i, \mathbf{n}_i)). \quad (18)$$

This allows for joint learning of both  $\theta$  and  $\varphi$ , as demonstrated in recent works [4], [10], treating unsupervised optimization of AEs and SINFONY as a supervised learning problem.

#### B. Stochastic Policy Gradient

We note that optimization of encoder and decoder with both gradients (15) or (17) requires model-awareness, i.e., a known and differentiable forward model  $p_\theta(\mathbf{y}|\mathbf{s})$ . But the gradient

$$\frac{\partial \ln p_\theta(\mathbf{y}|\mathbf{s})}{\partial \theta} = \frac{\partial \mu_\theta(\mathbf{s})}{\partial \theta} \cdot \frac{\partial p(\mathbf{y}|\mathbf{x})}{\partial \mathbf{x}} \cdot \frac{\partial \ln p(\mathbf{y}|\mathbf{x})}{\partial p(\mathbf{y}|\mathbf{x})} \quad (19)$$

with deterministic encoder  $\mathbf{x} = \mu_\theta(\mathbf{s})$  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.  $\theta$  and  $\varphi$ : The transmitter does not know  $q_\varphi(\mathbf{z}|\mathbf{y})$  and the receiver  $p_\theta(\mathbf{x}|\mathbf{s})$ , vice versa.

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

$$\frac{\partial \mathcal{L}_{\theta, \varphi}^{CE}}{\partial \theta} = -\frac{\partial}{\partial \theta} \mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y} \sim p(\mathbf{y}|\mathbf{x})p_\theta(\mathbf{x}|\mathbf{s})p(\mathbf{s}, \mathbf{z})} [\ln q_\varphi(\mathbf{z}|\mathbf{y})] \quad (20)$$

$$= -\mathbb{E}_{\mathbf{z}, \mathbf{s} \sim p(\mathbf{s}, \mathbf{z})} \left[ \int_{\mathcal{M}_x^{N_{Tx}}} \underbrace{\frac{\partial p_\theta(\mathbf{x}|\mathbf{s})}{\partial \theta}}_{=p_\theta(\mathbf{x}|\mathbf{s}) \cdot \frac{\partial \ln p_\theta(\mathbf{x}|\mathbf{s})}{\partial \theta}} \cdot \mathbb{E}_{\mathbf{y} \sim p(\mathbf{y}|\mathbf{x})} [\ln q_\varphi(\mathbf{z}|\mathbf{y})] d\mathbf{x} \right] \quad (21)$$

$$= -\mathbb{E}_{\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y} \sim p_\theta(\mathbf{x}|\mathbf{s})p(\mathbf{y}|\mathbf{x})p(\mathbf{s}, \mathbf{z})} \left[ \frac{\partial \ln p_\theta(\mathbf{x}|\mathbf{s})}{\partial \theta} \cdot \ln q_\varphi(\mathbf{z}|\mathbf{y}) \right] \quad (22)$$

$$\approx \frac{1}{N} \sum_{i=1}^N \frac{\partial \ln p_\theta(\mathbf{x}_i|\mathbf{s}_i)}{\partial \theta} \cdot [-\ln q_\varphi(\mathbf{z}_i|\mathbf{y}_i)]. \quad (23)$$

Again the log-trick is applied in (21) to arrive in (22) and the results hold for discrete RVs  $\mathbf{x}$ . Most importantly, (22) 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 into ADFs, usually, an objective function whose gradient is the Monte Carlo policy gradient estimator of (22), i.e., the REINFORCE gradient (23), is constructed:

$$\mathcal{L}_{\theta}^{\text{SPG}} = \frac{1}{N} \sum_{i=1}^N \ln p_{\theta}(\mathbf{x}_i | \mathbf{s}_i) \cdot [-\ln q_{\varphi}(\mathbf{z}_i | \mathbf{y}_i)] . \quad (24)$$

With objective (24) or REINFORCE gradient (23), we can finally optimize  $\mathcal{L}_{\theta, \varphi}^{\text{CE}}$  w.r.t.  $\theta$ , since we can sample  $\{\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y}\} \sim p_{\theta}(\mathbf{z}, \mathbf{s}, \mathbf{x}, \mathbf{y})$  and compute  $\frac{\partial \ln p_{\theta}(\mathbf{x}_i | \mathbf{s}_i)}{\partial \theta}$  at the transmitter and  $-\ln q_{\varphi}(\mathbf{z}_i | \mathbf{y}_i)$  being equal to the per-sample cross entropy at the receiver.

Note that  $\mathbf{s}_i$  and  $\mathbf{x}_i$  only have to be known at the transmitter and both  $\mathbf{z}_i$  and  $\mathbf{y}_i$  at the receiver, respectively. This enables the separation or spatial distribution of transmitter and receiver when the following conditions are met:

- Only an a priori known pilot sequence  $\mathcal{D}_p = \{\mathbf{z}_i, \mathbf{s}_i\}_{i=1}^{N_{\text{pilot}}}$  of size  $N_{\text{pilot}}$  is required. This sequence translates into the training set  $\mathcal{D}_T = \{\mathbf{z}_i, \mathbf{s}_i, \mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{N_{\text{train}}}$  of size  $N_{\text{train}}$  which is divided into batches of size  $N_b$  for SGD-based optimization.
- Moreover, we require a feedback link to transmit the per-sample cross entropy  $-\ln q_{\varphi}(\mathbf{z}_i | \mathbf{y}_i)$  to the encoder. This 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  $p_{\theta}(\mathbf{x} | \mathbf{s})$ . 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. Replacing  $p(\mathbf{y} | \mathbf{s})$  and  $p(\mathbf{y} | \mathbf{x})$  by  $p(\mathbf{x} | \mathbf{s})$  and  $p(\mathbf{x} | \bar{\mathbf{x}})$  in (19) and applying the result to (23), we derive that this function needs to be differentiable. If the encoder output, i.e., the action space, is continuous with  $\mathcal{M}_x = \mathbb{R}$ , we can achieve this using for example a Gaussian policy, i.e., a multivariate Gaussian pdf

$$p(\mathbf{x} | \bar{\mathbf{x}}) = p(\mathbf{x} | \bar{\mathbf{x}}, \sigma_{\text{exp}}^2) = \mathcal{N}\left((1 - \sigma_{\text{exp}}^2)^{1/2} \cdot \bar{\mathbf{x}}, \sigma_{\text{exp}}^2 \cdot \mathbf{I}\right) \quad (25)$$

with exploration variance  $\sigma_{\text{exp}}^2 \in (0, 1)$  where scaling of the mean  $\bar{\mathbf{x}} = \mu_{\theta}(\mathbf{s})$  is done to ensure the conservation of average energy. Furthermore, the Gaussian policy offers the benefit of simplicity in parametrization, requiring tuning of only two pdf parameters. Hence, we employ it in our numerical experiments. For discrete action spaces  $\mathcal{M}_x^{N_{\text{Tx}} \times 1}$ , a continuous differentiable relaxation such as the Gumbel Softmax is required [12].

In the special case  $\sigma_{\text{exp}}^2 \rightarrow 0$ , the Gaussian policy  $p(\mathbf{x} | \bar{\mathbf{x}}, \sigma_{\text{exp}}^2)$  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

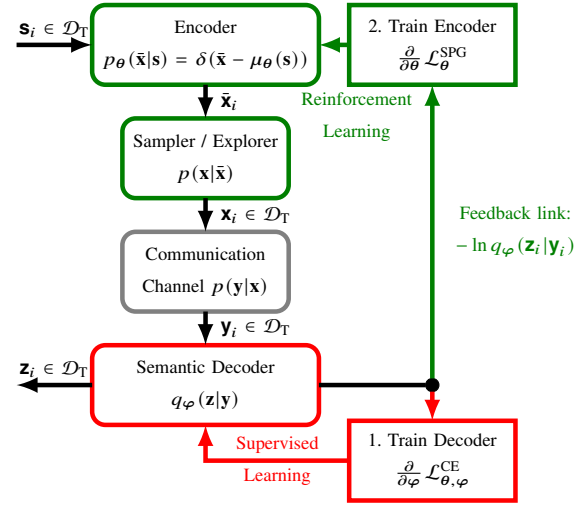


Fig. 2. Optimization procedure of a semantic encoder and decoder without a differentiable channel model: 1. Train the decoder supervised based on the training sequence and updated encoder but without sampler. 2. Encoder explores transmit signals  $\mathbf{x}_i$  and improves its policy according to the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

policy leads in their experiments to high variance of the gradient estimate (23) resulting in slow convergence. To compensate for this effect, we require a much larger and computationally expensive batch size  $N = N_b$ . From the view of RL, using a stochastic policy with  $\sigma_{\text{exp}}^2 \neq 0$  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. It does not require any channel model but a fixed pilot, i.e., training, sequence and a feedback link. Further, it enables separation of encoder and decoder. We show it in Fig. 2:

- 1) We note that according to (11) decoder optimization reduces to supervised learning w.r.t.  $\mathcal{L}_{\theta, \varphi}^{\text{CE}}$  and  $\varphi$  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  $\mathbf{x}_i$ . It is optimized based on the policy gradient of  $\mathcal{L}_{\theta}^{\text{SPG}}$  and the reward  $-\ln q_{\varphi}(\mathbf{z}_i | \mathbf{y}_i)$  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.

Reminiscent of the RL fashion of the stochastic policy optimization of Semantic INFORMATION traNsmission and recovery [4], we name this approach RL-SINFONY. Finally, we have derived the SPG for semantic communication starting from the InfoMax problem (1). Replacing  $I_{\theta}(\mathbf{z}; \mathbf{y})$  by  $I_{\theta}(\mathbf{s}; \mathbf{y})$ , this result also holds more general for classic communications.

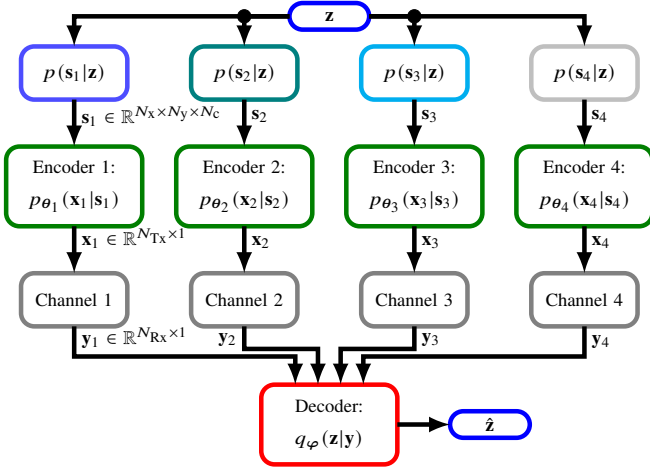


Fig. 3. RL-SINFONY scenario: Four distributed agents extract features for rate-efficient transmission to a decoder that extracts semantics.

#### 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 observation  $\mathbf{s}_i \sim p(\mathbf{s}_i|\mathbf{z})$  with  $i = 1, \dots, 4$ , 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  $p_{\theta_i}(\mathbf{x}_i|\mathbf{s}_i)$  jointly with a decoder  $q_{\varphi}(\mathbf{z}|\mathbf{y} = [\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4]^T)$  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  $\mathbf{s}_i \in \mathbb{R}^{N_x \times N_y \times N_c}$  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 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 [4] and our source code [13].

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, $N_w$ )
	GlobalAvgPool2D	$N_w$
Classifier	Softmax	$M = 10$

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  $\mathbf{x}_i \in \mathbb{R}^{N_{Tx} \times 1}$  for transmission through the wireless channel (see Fig. 3). We note that  $\mathbf{x}_i \in \mathbb{R}^{N_{Tx} \times 1}$  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 (25) after the Tx output compared to [4]. Further, we assume all agents and the Rx module to share a training set  $\mathcal{D}_T$  and a perfect reward feedback link from the Rx module to all agents.

At the receiver side, we use a single Rx module only with shared DNN layers of width  $N_w$  and parameters  $\varphi_{Rx}$  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  $q_{\varphi}(\mathbf{z}|\mathbf{y})$  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 with training set  $\mathcal{D}_T$  [13]. For cross-entropy loss minimization, we use the gradient approximations from Sec. III and the SGD-variant Adam with a batch size of  $N_b = 500$ . 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  $\text{SNR} = 1/\sigma_n^2$  with noise variance  $\sigma_n^2$ . We set  $N_w = N_{Feat}$  as default and refer to [4], [13] for more implementation



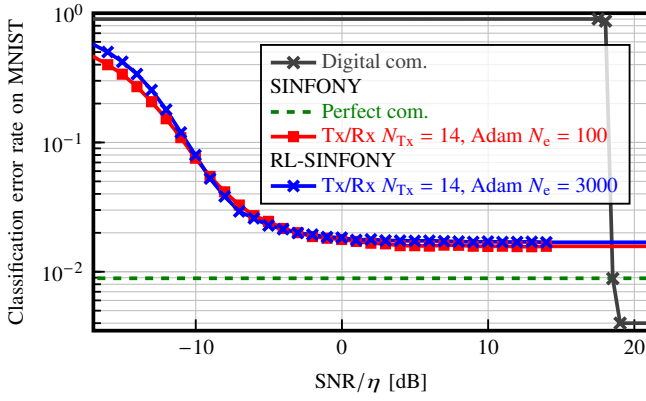


Fig. 4. Comparison of the classification error rate of RL-SINFONY and SINFONY with  $N_{Tx} = 14$  on MNIST as a function of normalized SNR.

details. In the following, we compare the performance of<sup>2</sup>:

- **Digital com.:** Digital transmission baseline from [4] with capacity achieving LDPC code and ResNet classifier.
- **SINFONY:** The distributed SINFONY design from [4] trained *model-aware* as one DNN with channel noise layer using the reparametrization trick (17) to approximate the gradients. We train for  $N_e = 100$  epochs with the MNIST dataset.
- **RL-SINFONY:** New approach trained *model-free* via RL as shown in Fig. 2 using SPG (23). 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  $N_e = 3000$  and add  $N_{e,rx} = 600$  epochs of receiver fine-tuning at the end [8]. To decrease the SPG estimator variance, we choose a rather high exploration variance  $\sigma_{exp}^2 = 0.15$ .
- **Perfect com.:** SINFONY trained with perfect communication links without Tx and Rx modules, but with Tx normalization. Thus, the plain power-constrained features are transmitted with  $N_{Tx} = 56$  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  $\mathbf{z}$  and normalize the SNR by the spectral efficiency  $\eta = N_{Feat}/N_{Tx}$  [4].

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

<sup>2</sup>It is not straightforward to compare the approach from [7] with RL-SINFONY as different models were investigated. We leave a detailed comparison with other approaches from the literature for future work.

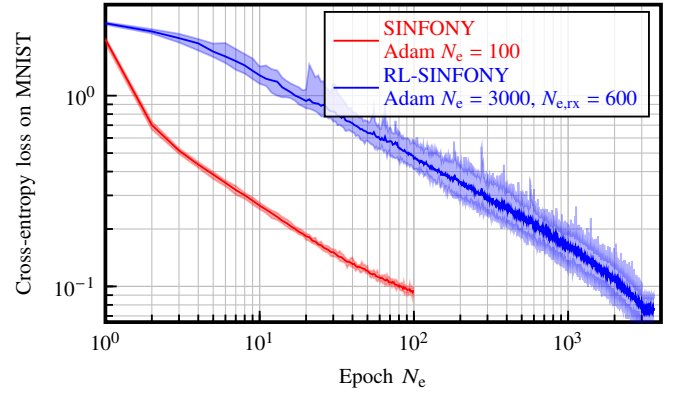


Fig. 5. Comparison of training convergence between RL-SINFONY and SINFONY with  $N_{Tx} = 14$  in terms of the cross-entropy loss on MNIST averaged over 10 runs as a function of training epochs  $N_e$ .

benchmark with ideal links (SINFONY - Perfect com.) at high SNR and beat Digital com. w.r.t. communication efficiency. 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. Note that Digital com. classifies the entire image at once and thus outperforms at high SNR [4].

2) **Convergence Rate:** Since the number of training epochs required to achieve the same performance deviates significantly with  $N_e + N_{e,rx} = 3000 + 600 = 3600$  compared to  $N_e = 100$ , 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 (23) compared to the reparametrization trick gradient (17). Also, we attribute the increased variance in training losses (blue-shaded area) to it.

3) **CIFAR10 dataset and convergence issues:** We further evaluate RL-SINFONY on the more challenging CIFAR10 validation dataset with  $N_{Tx} = 16$  and fine-tuned learning rate  $\epsilon = 10^{-4}$ . The performance curves of SINFONY and RL-SINFONY with Adam, depicted in Fig. 6, closely align, affirming the effectiveness of RL-SINFONY.

Nevertheless, it is crucial to highlight that training with Adam does not converge to a local minimum with the same 80% validation accuracy achieved by the SINFONY benchmark in [4]. In that work, we utilized SGD with a batch size of  $N_b = 64$ , ran for  $N_e = 200$  epochs, and employed a dedicated learning rate schedule. Despite exploring various hyperparameter settings, replicating the same performance with RL-SINFONY has proven elusive.

Additionally, we observed that the training of RL-SINFONY on the CIFAR10 dataset exhibits slow convergence. For example, using SGD with  $N_b = 128$  and  $\epsilon = 10^{-4}$  (see Fig. 6), we achieve a validation accuracy of 50% at high SNR after  $N_e + N_{e,rx} = 5000 + 1000 = 6000$  epochs, still gradually improving to a maximum of 60% after an extensive training

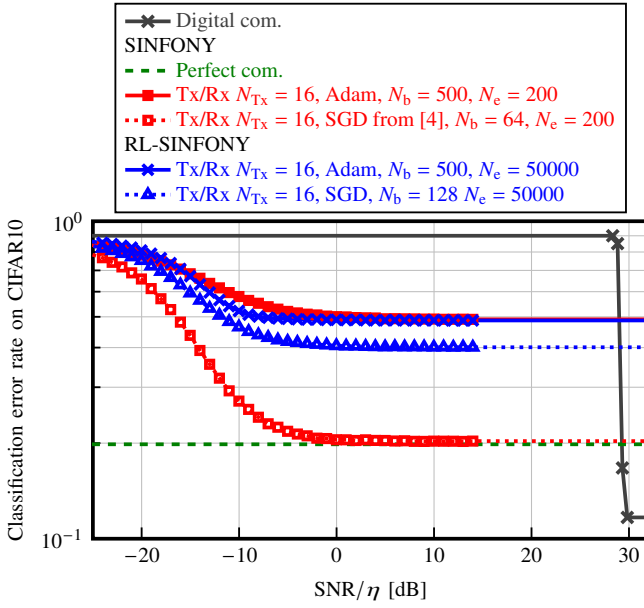


Fig. 6. Comparison of the classification error rate of RL-SINFONY and SINFONY with  $N_{Tx} = 16$  on CIFAR10 as a function of normalized SNR.

period of  $N_e + N_{e,rx} = 50000 + 10000 = 60000$  epochs.

We assume the slow convergence to be caused by the high variance of the REINFORCE gradient (23), which increases by decreasing  $\sigma_{exp}^2$  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], [14]. Note that, analogous to the mean  $\bar{\mathbf{x}} = \mu_{\theta}(\mathbf{s})$  of the Gaussian policy (25), also the exploration variance  $\sigma_{exp}^2$  can be parametrized by a DNN with shared parameters  $\theta$  or independent parameters  $\theta = [\theta_{\bar{\mathbf{x}}}, \theta_{\sigma_{exp}^2}]^T$ . Both approaches could facilitate quicker convergence and more efficient hyperparameter tuning, ultimately leading to higher validation accuracy.

## 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, separating transmitter and receiver, and not requiring a known or differentiable channel model – a crucial step towards deployment in practice. Further, we derived 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 J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 5–41, Jan. 2023.

[4] E. Beck, C. Bockelmann, and A. Dekorsy, “Semantic Information Recovery in Wireless Networks,” *Sensors*, vol. 23, no. 14, p. 6347, July 2023.

[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, June 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, C. Bockelmann, and A. Dekorsy, “CMDNet: Learning a Probabilistic Relaxation of Discrete Variables for Soft Detection With Low Complexity,” *IEEE Trans. Commun.*, vol. 69, no. 12, pp. 8214–8227, Dec. 2021.

[13] E. Beck, “Semantic Information Transmission and Recovery (SINFONY) Software,” July 2023, Zenodo. [Online]. Available: <https://doi.org/10.5281/zenodo.8006567>

[14] 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.