Learning Task-Oriented Communication for Edge Inference: An Information Bottleneck Approach

Jiawei Shao, *Student Member, IEEE,* Yuyi Mao, *Member, IEEE,* and Jun Zhang, *Senior Member, IEEE*

***Abstract*—This paper investigates task-oriented communication for edge inference, where a low-end edge device transmits the extracted feature vector of a local data sample to a powerful edge server for processing. It is critical to encode the data into an *informative* and *compact* representation for low-latency inference given the limited bandwidth. We propose a learning-based communication scheme that jointly optimizes feature extraction, source coding, and channel coding in a task-oriented manner, i.e., targeting the downstream inference task rather than data reconstruction. Specifically, we leverage an information bottleneck (IB) framework to formalize a rate-distortion tradeoff between the informativeness of the encoded feature and the inference performance. As the IB optimization is computationally prohibitive for the high-dimensional data, we adopt a variational approximation, namely the variational information bottleneck (VIB), to build a tractable upper bound. To reduce the communication overhead, we leverage a sparsity-inducing distribution as the variational prior for the VIB framework to sparsify the encoded feature vector. Furthermore, considering dynamic channel conditions in practical communication systems, we propose a variable-length feature encoding scheme based on dynamic neural networks to adaptively adjust the activated dimensions of the encoded feature to different channel conditions. Extensive experiments evidence that the proposed task-oriented communication system achieves a better rate-distortion tradeoff than baseline methods and significantly reduces the feature transmission latency in dynamic channel conditions.**

***Index Terms*—Task-oriented communication, edge inference, information bottleneck, variational inference.**

I. Introduction

The recent revival of artificial intelligence (AI) has led to their adaptations in a broad spectrum of application domains, ranging from speech recognition [1] and natural language processing (NLP) [2], to computer vision [3] and augmented/virtual reality (AR/VR) [4]. Most recently, the potential of AI technologies has also been exemplified in communication systems [5], [6]. Aiming at delivering data with extreme levels of reliability and efficiency, various design problems of *data-oriented communication*, including transceiver structures [7], source/channel coding [8], signal detection [9], and radio resource management [10], have been revisited intensively using AI techniques, especially deep neural networks (DNNs), breeding the emerging area of "*learning to communicate*". It is widely perceived that learning-driven techniques are critical complements to traditional model-driven approaches for communication system designs that rely heavily on expert knowledge, and will undoubtedly transform the wireless networks toward the next generation [11].

Meanwhile, emerging AI applications also raise new communication problems [12], [13]. To provide an immersive user experience, DNN-based mobile applications need to be performed within the edge of wireless networks, which eliminates the excessive latency incurred by routing data to the Cloud, and is referred to as *edge inference* [14], [13]. Edge inference can be implemented by deploying DNNs at an edge server located in close proximity to mobile devices, known as *edge-only inference*. However, the transmission latency remains a bottleneck for applications with stringent delay requirements [15], [16], [17], as a huge volume of data (e.g., 3D images, high-definition videos, and point cloud data) need to be uploaded. On the other hand, the resource-demanding nature of DNNs often makes it infeasible to be deployed as a whole locally for *device-only inference* due to the limited on-device computational resources [18].

Device-edge co-inference appears to be a prominent solution for fast edge inference [14], [19], [20], which reduces the communication overhead by harvesting the available computational resources at both the edge servers and mobile devices. A mobile device first extracts a compact feature vector from the raw input data using an affordable neural network and then uploads it for server-based processing. Nevertheless, most existing device-edge co-inference proposals simply split a pre-trained DNN into two subnetworks to be deployed at a device and a server, leaving feature compression and transmission to a traditional communication module [20]. Such kind of decoupled treatment ignores the interplay between wireless communications and the inference tasks, and thus fails to exploit the full benefits of collaborative inference since the communication strategies can be adaptive to specific tasks. To address this limitation and improve the inference performance, in this paper, we propose a *task-oriented communication* principle for edge inference and develop an innovative learning-driven approach under the framework of information bottleneck (IB) [21].

A. Related Works and Motivations

The line of research on "learning to communicate" stems from the introductory article on deep learning for the physical layer design in [7], where information transmission was viewed as a data reconstruction task, and a communication system can thus be modeled by a DNN-based autoencoder with the wireless channel simulated by a non-trainable layer.

J. Shao and J. Zhang are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong (E-mail: jiawei.shao@connect.ust.hk, eejzhang@ust.hk). Y. Mao is with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hong Kong (E-mail: yuyi-eie.mao@polyu.edu.hk). (The corresponding author is J. Zhang.)

The autoencoder-based framework for communication systems was later extended to a deep joint source-channel coding (JSCC) architecture for wireless image transmission in [8], which enjoys significant improvement of image reconstruction quality over separate source/channel coding techniques. JSCC has also been applied to natural language processing for text transmission, which was accomplished by incorporating the semantic information of sentences using recurrent neural networks [22]. It is worth noting that the aforementioned works focus on *data-oriented communication*, which targets at transmitting data reliably given the limited radio resources.

Nevertheless, the shifted objective of feature transmissions for accurate edge inference with low latency is not aligned with that of data-oriented communication, as it regards a part of the raw input data (e.g., nuisance, task-irrelevant information) as meaningless. Thus, recovering the original data sample with high fidelity at the edge server results in redundant communication overhead, which leaves room for further compression. This insight is also supported by a basic principle from representation learning [23]: A good representation should be insensitive (or invariant) to nuisances such as translations, rotations, occlusions. Thus, we advocate for *task-oriented communication* for applications such as edge inference, to improve the efficiency by transmitting *sufficient* but *minimal* information for the downstream task.

There have been recent studies on feature compression for efficient transmission in edge inference [24], [25], [26], [27], [28]. In particular, for the image classification task, an end-to-end architecture was proposed in [26] to jointly optimize the feature compression and encoding by integrating deep JSCC. In contrast to data-oriented communication that concerns the data recovery metrics (e.g., the l_2-distance or bit error rate), the proposed method was directly trained with the cross-entropy loss for the targeted classification task and ignored the data reconstruction quality. The end-to-end training facilitates the mapping of task-relevant information to the channel symbols and omits the irrelevance. Similar ideas were utilized to design feature compression and encoding schemes for image retrieval tasks at the wireless network edge in [29] and for point cloud data processing in [30].

While the end-to-end learning-driven architectures for task-oriented communication have been proven effective in saving communication bandwidth, there remain multiple restrictions unsolved in order to unleash their highest potentials: First, there lacks a systematic way to quantify the informativeness of the encoded feature vector and its impact on the inference tasks, hindering to achieve the best inference performance given the available resources; Besides, the dynamic wireless channel condition necessitates adaptive encoding scheme for reliable feature transmission, which has received less attention in existing frameworks (e.g. [26], [27], [28], [31]). These form the main motivations of our study.

Data-oriented communication relies on classical source coding and channel coding theory, which, however, is not optimized for task-oriented communication. Recently, an information theoretical design principle, named information bottleneck (IB) [21], has been applied to investigate deep learning, which seeks the right balance between data fit and generalization by using the mutual information as both a cost function and a regularizer. Particularly, the IB framework maximizes the mutual information between the latency representation and the label of the data sample to promote high accuracy, while minimizing the mutual information between the representation and the input sample to promote generalization. Such a trade-off between preserving the *relevant* information and finding a *compact* representation fits nicely with bandwidth-limited edge inference and thus will be adopted as the main design principle in our study for task-oriented communication. The IB framework is inherently related to the communication problem of remote source coding (RSC) [32]. It has recently attracted great attention from both the machine learning and information theory communities [33], [34], [35], [36]. Nevertheless, applying it to task-oriented communication demands additional optimization, which forms the main technical contributions of our study.

B. Contributions

In this paper, we develop effective methods for task-oriented communication for device-edge co-inference based on the IB principle [21]. Our major contributions are summarized as follows:

- We design the task-oriented communication system by formalizing a rate-distortion tradeoff using the IB framework. Our formulation aims at maximizing the mutual information between the inference result and the encoded feature, meanwhile, minimizing the mutual information between the encoded feature and input data. Thus, it addresses the objectives of improving the inference accuracy, while reducing the communication overhead, respectively. To the best of our knowledge, this is the first time that IB is introduced to design wireless edge inference systems.
- As the mutual information terms in the IB formulation are generally intractable for DNNs with high-dimensional features, we leverage the variational approximation, known as variational information bottleneck (VIB), to devise a tractable upper bound. Besides, by selecting a sparsity-inducing distribution as the variational prior, the VIB framework identifies and prunes the redundant dimensions of the encoded feature vector to reduce the communication overhead. The proposed method is named as *variational feature encoding* (VFE).
- We extend the proposed task-oriented communication scheme to dynamic communication environments by enabling flexible adjustment of the transmitted signal length. In particular, we develop a *variable-length variational feature encoding* (VL-VFE) based on dynamic neural networks that can adaptively adjust the active dimensions according to different channel conditions.
- The effectiveness of the proposed task-oriented communication schemes is validated in both static and dynamic channel conditions on image classification tasks. Extensive simulation results demonstrate that VFE and VL-VFE outperform the traditional communication design and existing learning-based joint source-channel coding for data-oriented communication.

C. Organization

The rest of the paper is organized as follows. Section II introduces the system model and describes the design objective of task-oriented communication. Section III and Section IV propose the task-oriented communication schemes in static and dynamic channel conditions, respectively. In Section V, we provide extensive simulation results to evaluate the performance of the proposed task-oriented communication schemes. Finally, Section VI concludes the paper.

D. Notations

Throughout this paper, upper-case letters (e.g. X and Y) and lower-case letters (e.g. $\boldsymbol{x}$ and $\boldsymbol{y}$) stand for random variables and their realizations, respectively. The entropy of Y and the conditional entropy of Y given X are denoted as $H(Y)$ and $H(Y|X)$, respectively. The mutual information between X and Y is represented as $I(X,Y)$, and the Kullback-Leibler (KL) divergence between two probability distributions $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ is denoted as $D_{KL}(p||q)$. The statistical expectation of X is denoted as $\mathbf{E}(X)$. We further denote the Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ as $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ and use $\boldsymbol{I}$ to represent the identity matrix.

II. System Model and Problem Description

A. System Model

We consider task-oriented communication in a device-edge co-inference system as shown in Fig. 1b, where two DNNs are deployed at the mobile device[1] and the edge server respectively so that they can cooperate to perform inference tasks, e.g., image classification and object detection. The input data $\boldsymbol{x}$ and its target variable $\boldsymbol{y}$ (e.g., label) are deemed as different realizations of a pair of random variables (X,Y). The encoded feature, received feature (noise-corrupted feature), and the inference result are respectively instantiated by random variables Z, $\hat{Z}$ and $\hat{Y}$. These random variables constitute the following probabilistic graphical model:

$$Y \rightarrow X \rightarrow Z \rightarrow \hat{Z} \rightarrow \hat{Y}, \tag{1}$$

which satisfies $p(\hat{\boldsymbol{y}}|\boldsymbol{x}) = p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})p_{\text{channel}}(\hat{\boldsymbol{z}}|\boldsymbol{z})p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, with DNN parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ to be discussed below.

As shown in Fig. 1b, the on-device network defines the conditional distribution $p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ parameterized by $\boldsymbol{\phi}$, which consists of a feature extractor and a JSCC encoder. The extractor first identifies the task-relevant feature from the raw input $\boldsymbol{x}$, and then the JSCC encoder maps the feature values to the channel input symbols $\boldsymbol{z}$. Since both the extractor and encoder are parameterized by DNNs, these two modules can be jointly trained in an end-to-end manner. Then, the encoded feature $\boldsymbol{z}$ is transmitted to the server over the noisy wireless channel, and the server receives the noise-corrupted feature $\hat{\boldsymbol{z}}$. In this paper, we assume a scalar Gaussian channel between the mobile device and the edge server for simplicity, which is modeled as a non-trainable layer with the transfer function denoted as $\hat{\boldsymbol{z}} = \boldsymbol{z} + \boldsymbol{\epsilon}$. The additive channel noise $\boldsymbol{\epsilon}$ is sampled from a zero-mean Gaussian distribution with σ^2 as the noise variance, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})$. To account for the limited transmit power at the mobile device, we constrain the power of each dimension of the encoded feature vector to be below P, i.e., $z_i^2 \leq P, \forall i = 1,\cdots,n$ with n as the encoded feature vector dimension. Thus, the channel condition can be characterized by the peak signal-to-noise ratio (PSNR) defined as follows:

$$\text{PSNR} = 10 \log \frac{P}{\sigma^2} \text{ (dB)}.$$

Note that although we assume a scalar Gaussian channel model for simplicity, the system can be extended to other channel models as long as we can estimate the channel transfer function [37] and the distribution $p_{\text{channel}}(\hat{\boldsymbol{z}}|\boldsymbol{z})$. Finally, the server-based network leverages $\hat{\boldsymbol{z}}$ for further processing and outputs the inference result $\hat{\boldsymbol{y}}$ with the distribution $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}})$ parameterized by $\boldsymbol{\theta}$.

B. Problem Description

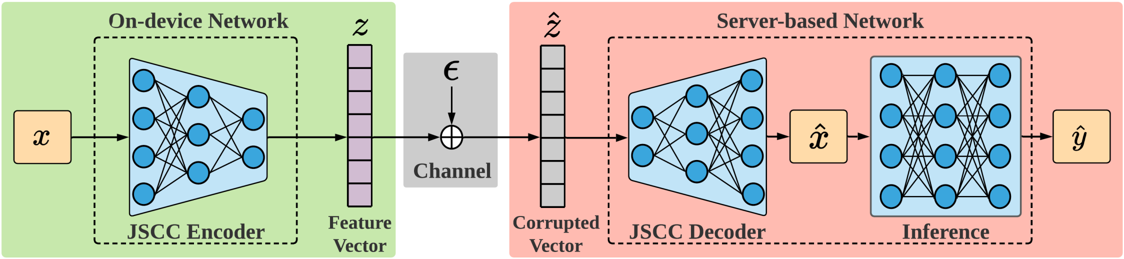
The communication overhead is characterized by the number of nonzero dimensions of the output of the JSCC encoder. Intuitively, if symbols over more dimensions are transmitted, the edge server will get a high-quality feature vector, which leads to higher inference accuracy, but it will induce a higher communication overhead and latency. So there is an inherent tradeoff between the inference performance and the communication overhead, which is a key ingredient for the design of task-oriented communication. This can be regarded as a new and special kind of *rate-distortion tradeoff*. Therefore, we resort to the information bottleneck (IB) principle [21] to formulate an optimization problem that minimizes the following objective function[2]:

$$\begin{aligned}
\mathcal{L}_{IB}(\boldsymbol{\phi}) &= \underbrace{-I(\hat{Z},Y)}_{\text{Distortion}} + \beta \underbrace{I(\hat{Z},X)}_{\text{Rate}} \\
&= \mathbf{E}_{p(\boldsymbol{x},\boldsymbol{y})}\big\{\mathbf{E}_{p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})}[-\log p(\boldsymbol{y}|\hat{\boldsymbol{z}})] \\
&\quad + \beta D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})\|p(\hat{\boldsymbol{z}})\right)\big\} - H(Y) \\
&\equiv \mathbf{E}_{p(\boldsymbol{x},\boldsymbol{y})}\big\{\mathbf{E}_{p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})}[-\log p(\boldsymbol{y}|\hat{\boldsymbol{z}})] \\
&\quad + \beta D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})\|p(\hat{\boldsymbol{z}})\right)\big\},
\end{aligned} \tag{2}$$

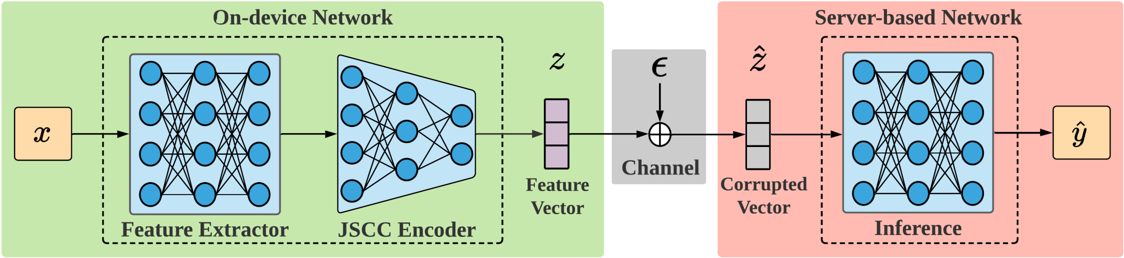
where the equivalence in the last row is in the sense of optimization, ignoring the constant term $H(Y)$. The objective function is a weighted sum of two mutual information terms with $\beta > 0$ controling the tradeoff. Specifically, the quantity $I(\hat{Z},X)$ is comprehended as the preserved information in $\hat{Z}$ given X and measured by the minimum description length [38] (or rate). Besides, since the entropy of Y, i.e., $H(Y)$, is a constant related to the input data distribution, minimizing the term $-I(\hat{Z},Y)$ is equivalent to minimizing the conditional entropy $H(Y|\hat{Z})$, which characterizes the uncertainty (distortion) of the inference result Y given the received noise-corrupted feature vector $\hat{Z}$. Thus, the IB principle formalizes a rate-distortion tradeoff for edge inference systems, and minimizes

[1] While two components, i.e., a feature extractor and a JSCC encoder, are shown in Fig. 1b at the device, they can be regarded as a single DNN. We consider resource-constrained devices that can only afford light DNNs, which are unable to complete the inference task with sufficient accuracy. More details of the adopted neural network architecture will be discussed in Section V.

[2] Note that the IB objective function is unrelated to the parameter $\boldsymbol{\theta}$ since the distribution $p(\boldsymbol{y}|\hat{\boldsymbol{z}})$ is defined by $p(\boldsymbol{x},\boldsymbol{y})$, $p_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, and $p_{\text{channel}}(\hat{\boldsymbol{z}}|\boldsymbol{z})$.



(a) Data-oriented communication for device-edge co-inference.



(b) Task-oriented communication for device-edge co-inference.

Fig. 1. Two kinds of communication schemes for device-edge co-inference: Learning-based data-oriented and task-oriented communication. The green region corresponds to a mobile device, and the red region corresponds to an edge server. In data-oriented communication (top), a mobile device transmits the encoded feature $\boldsymbol{z}$ of the original data $\boldsymbol{x}$ (e.g., an image). Then, an edge server attempts to decode the data $\hat{\boldsymbol{x}}$ based on the noise-corrupted feature $\hat{\boldsymbol{z}}$, and further utilizes $\hat{\boldsymbol{x}}$ as input to obtain the inference result $\hat{\boldsymbol{y}}$ (e.g., the label of input data). In contrast, task-oriented communication (bottom) extracts and encodes useful information $\boldsymbol{z}$ jointly by the on-device network, and the receiver directly leverages $\hat{\boldsymbol{z}}$ to obtain the inference result $\hat{\boldsymbol{y}}$, without recovering the original data. Therefore, $\boldsymbol{z}$ could be a highly compressed representation since the task-irrelevant information can be discarded.

the conditional mutual information $I(X, \hat{Z}|Y)$, which corresponds to the amount of redundant information that needs to be transmitted. Compared with data-oriented communication, the IB framework retains the task-relevant information and results in $I(\hat{Z}, X)$ that is much smaller than $H(X)$, which reduces the communication overhead.

C. Main Challenges

The IB framework is promising for task-oriented communication as it explicitly quantifies the informativeness of the encoded feature vector and offers a formalization of the rate-distortion tradeoff in edge inference. However, there are three main challenges when applying it to develop practical feature encoding methods, listed as follows.

- **Estimation of mutual information**: The computation of mutual information terms for high-dimensional data with unknown distributions is challenging since the empirical estimate for the probability distribution requires the sampling number to increase exponentially with the dimension [39]. Therefore, developing a tractable estimator for mutual information is critical to make the problem solvable.
- **Effective control of communication overhead**: Minimizing the mutual information between the input data and the feature vector indeed reduces the redundancy about task-irrelevant information. However, there is no direct link between redundancy reduction and feature sparsification, which controls the communication overhead with a JSCC encoder. Thus, to reduce the communication overhead, an effective method is needed to aggregate the nuisance to the expandable dimensions so that the number of symbols to be transmitted is minimized.
- **Dynamic channel conditions**: The hostile wireless channel always poses significant challenges for communication systems. Particularly, the channel dynamics have to be accounted for. Dynamically adjusting the encoded feature length based on the DNNs is nontrivial, as the neural network structure is fixed since initialization. Changing the activation of neurons according to the channel conditions calls for other control modules.

The following two sections will tackle these challenges, and develop effective methods for task-oriented communications. The effectiveness of the proposed methods will be tested in Section V.

III. Variational Feature Encoding

In this section, we develop a variational information bottleneck (VIB) framework to resolve the difficulty of mutual information computation of the original IB objective in (2). Besides, we show that by selecting a sparsity-inducing distribution as the variational prior, minimizing the mutual information between the raw input data X and the noise-corrupted feature $\hat{Z}$ facilitates the sparsification of $\hat{Z}$ by pruning the task-irrelevant dimensions. Such an activation pruning scheme, i.e., removing neurons in a DNN, is effective in reducing the overhead of task-oriented communication. Based on this idea, we name our proposed method as variational feature encoding (VFE). This section assumes a static channel condition, while dynamic channels will be treated in Section IV.

A. Variational Information Bottleneck Reformulation

The variational method is a natural way to approximate intractable computations based on some adjustable parameters (e.g., weights in DNNs), and it has been widely applied in machine learning, e.g., the variational autoencoder [40]. In the VIB framework, the central idea is to introduce a set of approximating densities to the intractable distribution.

Revisiting the probabilistic graphical model in (1), the distribution $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})$ is determined by the on-device DNN and the channel model, i.e., $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x}) = p_{\boldsymbol{\phi}}(z|\boldsymbol{x})p_{\text{channel}}(\hat{z}|z;\boldsymbol{\epsilon})$. Particularly, as we adopt a deterministic on-device network, $p_{\boldsymbol{\phi}}(z|\boldsymbol{x})$ can be regarded as a Dirac-delta function. Then, we have $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x}) = \mathcal{N}\left(\hat{z}|z(\boldsymbol{x};\boldsymbol{\phi}),\sigma^2\boldsymbol{I}\right)$, where the deterministic function $z(\boldsymbol{x};\boldsymbol{\phi})$ maps $\boldsymbol{x}$ to z parameterized by $\boldsymbol{\phi}$. For notational simplicity, we rewrite $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x}) = \mathcal{N}\left(\hat{z}|z(\boldsymbol{x};\boldsymbol{\phi}),\sigma^2\boldsymbol{I}\right)$ as $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x}) = \mathcal{N}\left(\hat{z}|z,\sigma^2\boldsymbol{I}\right)$.

With a known distribution $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})$ and the joint data distribution $p(\boldsymbol{x},\boldsymbol{y})$, the distributions $p(\hat{z})$ and $p(\boldsymbol{y}|\hat{z})$ are fully characterized by the underlying Markov chain $Y \leftrightarrow X \leftrightarrow \hat{Z}$. Unfortunately, these two distributions are intractable due to the following high-dimensional integrals:

$$p(\hat{z}) = \int p(\boldsymbol{x})p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})d\boldsymbol{x},$$
$$p(\boldsymbol{y}|\hat{z}) = \int \frac{p(\boldsymbol{x},\boldsymbol{y})p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})}{p(\hat{z})}d\boldsymbol{x}.$$

To overcome this issue, we apply two variational distributions $q(\hat{z})$ and $q_{\boldsymbol{\theta}}(\boldsymbol{y}|\hat{z})$ to approximate the true distributions $p(\hat{z})$ and $p(\boldsymbol{y}|\hat{z})$, respectively, where $\boldsymbol{\theta}$ is the parameters of the server-based network shown in Fig. 1b that computes the inference result $\hat{\boldsymbol{y}}$. Therefore, we recast the objective function in (2) as follows:

$$\begin{aligned}\mathcal{L}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta}) = \mathbf{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big\{&\mathbf{E}_{p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})}\left[-\log q_{\boldsymbol{\theta}}(\boldsymbol{y}|\hat{z})\right] \\ &+\beta D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})\|q(\hat{z})\right)\Big\}.\end{aligned} \tag{3}$$

The above formulation is termed as the variational information bottleneck (VIB) [36], which invokes an upper bound on the IB objective function in (2). Details of the derivations are deferred to the Appendix A. By further applying the reparameterization trick [40] and Monte Carlo sampling, we are able to obtain an unbiased estimate of the gradient and hence optimize the objective using stochastic gradient descent. In particular, given a mini-batch of data $\{(\boldsymbol{x_i},\boldsymbol{y_i})\}_{i=1}^{M}$ and sampling the channel noise L times for each pair $(\boldsymbol{x_i},\boldsymbol{y_i})$, we have the following empirical estimation:

$$\begin{aligned}\mathcal{L}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta}) \simeq &\frac{1}{M}\sum_{m=1}^{M}\Bigg\{\frac{1}{L}\sum_{l=1}^{L}\left[-\log q_{\boldsymbol{\theta}}\left(\boldsymbol{y_m}|\hat{z}_{\boldsymbol{m},\boldsymbol{l}}\right)\right] \\ &+\beta D_{KL}\left(p_{\boldsymbol{\phi}}\left(\hat{z}|\boldsymbol{x_m}\right)\|q(\hat{z})\right)\Bigg\},\end{aligned} \tag{4}$$

where $\hat{z}_{\boldsymbol{m},\boldsymbol{l}} = z_{\boldsymbol{m}} + \boldsymbol{\epsilon}_{\boldsymbol{m},\boldsymbol{l}}$ and $\boldsymbol{\epsilon}_{\boldsymbol{m},\boldsymbol{l}} \sim \mathcal{N}\left(\boldsymbol{0},\sigma^2\boldsymbol{I}\right)$.

In the next subsection, we illustrate that minimizing the VIB objective helps to prune the redundant dimensions in the encoded feature vector, and thus it serves as a suitable and tractable objective for task-oriented communication.

B. Redundancy Reduction and Feature Sparsification

As we leverage the IB principle instantiated via a variational approximation, minimizing the KL-divergence term $D_{KL}\left(p(\hat{z}|x)\|q(\hat{z})\right)$ shall reduce the redundancy in feature $\hat{Z}$. However, it does not guarantee sparse activations in the feature encoding process. For example, if the reduced redundancy is distributed equally across all dimensions and each dimension still preserves task-relevant information, the encoded feature vector may have a high dimension that leads to a high communication overhead. To obtain a feature vector $\hat{Z}$ that aggregates the task-irrelevant information into certain expendable dimensions through end-to-end training, we adopt the log-uniform distribution as the variational prior, i.e., $q(\hat{z})$, to induce sparsity [41]. In particular, we choose the mean-field variational approximation [40] to alleviate the computation complexity, i.e., given an n-dimensional $\hat{z}$, $q(\hat{z}) = \prod_i^n q(\hat{z}_i)$. Specifically, for each dimension $\hat{z}_i$, the variational prior distribution is chosen as:

$$q\left(\log|\hat{z}_i|\right) = \text{constant}.$$

Since $p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x}) = \prod_i^n p_{\boldsymbol{\phi}}(\hat{z}_i|\boldsymbol{x})$, the KL-divergence term in (3) can be decomposed into a summation:

$$D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x})\|q(\boldsymbol{x})\right) = \sum_{i=1}^{n} D_{KL}\left(p_{\boldsymbol{\phi}}\left(\hat{z}_i|\boldsymbol{x}\right)\|q\left(\hat{z}_i\right)\right). \tag{5}$$

Nevertheless, as the KL-divergence term in (5) does not have a closed-form expression, we utilize the approximation proposed in [42] as follows:

$$\begin{aligned}&-D_{KL}\left(p_{\boldsymbol{\phi}}\left(\hat{z}_i|\boldsymbol{x}\right)\|q\left(\hat{z}_i\right)\right) = \\ =&\frac{1}{2}\log\alpha_i - \mathbb{E}_{\epsilon\sim\mathcal{N}(1,\alpha_i)}\log|\epsilon| + \mathrm{C} \\ \approx& k_1 S\left(k_2 + k_3\log\alpha_i\right) - 0.5\log\left(1+\alpha_i^{-1}\right) + \mathrm{C},\end{aligned} \tag{6}$$

where

$$\alpha_i = \frac{\sigma^2}{z_i^2} \quad k_1 = 0.63576 \quad k_2 = 1.87320 \quad k_3 = 1.48695,$$

and C is a constant. Besides, z_i is the i-th dimension in z, and $S(\cdot)$ denotes the sigmoid function. It can be verified that the approximate KL-divergence approaches its minimum when α_i goes to infinite (i.e., z_i goes to zero), and minimizing this term encourages the value of z_i to be small. Empirical results in Section V show that the selected sparsity-inducing distribution sparsifies some dimensions in z, i.e., $z_i \equiv 0$ for arbitrary input, which can be pruned to reduce the communication overhead.

C. Variational Pruning on Dimension Importance

While the selected variational prior helps to promote sparsity in the feature vector, we still need an effective method to determine which of the dimensions can be pruned. Maintaining $z_i \equiv 0$ requires all the weights and the bias corresponding to z_i in this layer to converge to zero. However, checking each parameter is time-consuming in a large-scale DNN. To develop an efficient solution, we introduce a *dimension importance* vector $\boldsymbol{\gamma}$ to denote the importance of each output neuron. Revisiting the fully-connected (FC) layer, each neuron has full

Algorithm 1 Training Variational Feature Encoding (VFE)

```
Input: $T$ (number of iterations), $n$ (number of output dimension of encoder), $L$ (number of channel noise samples per datapoint), batch size $M$, channel variance $\sigma^2$, and threshold $\gamma_0$.
while epoch $t = 1$ to $T$ do
    Select a mini-batch of data $\{(\boldsymbol{x_m}, \boldsymbol{y_m})\}_{m=1}^M$
    Compute the encoded feature vector $\{\boldsymbol{z_m}\}_{m=1}^M$ based on (8)
    Compute the appropriate KL-divergence based on (6)
    while $m = 1$ to $M$ do
        Sample the noise $\left\{\boldsymbol{\epsilon_{m,l}}\right\}_{l=1}^L \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$
    end while
    Compute the loss $\mathcal{L}_{VIB}(\boldsymbol{\phi}, \boldsymbol{\theta})$ based on (4)
    Update parameters $\boldsymbol{\phi}, \boldsymbol{\theta}$ through backpropagation.
    while $i = 1$ to $n$ do
        if $\gamma_i \leq \gamma_0$ then
            Prune the $i$-th dimension in the encoded feature vector
        end if
    end while
end while
```

connections to its input $\boldsymbol{a}$, and their activations can thus be computed with a matrix multiplication with $\boldsymbol{W}$ followed by an offset $\boldsymbol{b}$ as follows:

$$\mathcal{FC}(\boldsymbol{a}) = \boldsymbol{W}\boldsymbol{a} + \boldsymbol{b} = \widetilde{\boldsymbol{W}}\tilde{\boldsymbol{a}}, \tag{7}$$

where $\widetilde{\boldsymbol{W}} = [\boldsymbol{W}, \boldsymbol{b}]$ is an augmented weight matrix, and $\tilde{\boldsymbol{a}} = [\boldsymbol{a}^T, 1]^T$ is an augmented input vector. By denoting the i-th row in the augmented weight matrix $\widetilde{\boldsymbol{W}}$ as $\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}$ and the i-th dimension in $\boldsymbol{\gamma}$ as γ_i, we rewrite the augmented weight matrix as $\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot} = \gamma_i \frac{\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}}{\|\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}\|_2}$, where γ corresponds to the scale factor for each row. The proposed VFE method defines the mapping from the input $\boldsymbol{x}$ to the encoded feature $\boldsymbol{z}$ according to the following formula:

$$z_i = \operatorname{Tanh}\left(\gamma_i \frac{\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}}{\|\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}\|_2} \boldsymbol{f}(\boldsymbol{x})\right), \tag{8}$$

where z_i is the i-th dimension of $\boldsymbol{z}$, and $\operatorname{Tanh}(\cdot)$ is the activation function. Besides, function $\boldsymbol{f}(\cdot)$ is defined by the previous on-device layers, and its output $\boldsymbol{f}(\boldsymbol{x})$ is the input of the fully-connected layer (i.e., $\boldsymbol{a} = \boldsymbol{f}(\boldsymbol{x})$ in (7)). As the weight vector $\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}$ is normalized by its l_2-norm, the magnitude of z_i is highly dependent on the scale factor γ_i. When γ_i is close to zero, z_i is also close to zero, and the corresponding $p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})$ degrades to the channel noise distribution without valid information. Based on this idea, we eliminate the redundant channels when the parameter γ_i is less than a threshold γ_0. Since the Tanh activation function has an output range from -1 to 1, the peak transmitted power P is constrained to 1. Note that the formula in (8) can be easily extended to convolutional layers by replacing the matrix multiplication with convolution. Such a variational pruning process is one of the main components of the proposed VFE method. The training procedures for VFE are illustrated in Algorithm 1.

IV. Variable-length Variational Feature Encoding

The task-oriented communication scheme developed in Section III assumes static wireless channels. In practice, wireless data transmission may experience changes due to various factors such as beam blockage and signal attenuation. This necessitates instant link adaptation to improve the efficiency of feature encoding for low-latency inference. In this section, we extend our findings in Section III and propose a new encoding scheme, namely variable-length variational feature encoding (VL-VFE), by designing a dynamic neural network, which admits flexible control of the encoded feature dimension.

A. Background on Dynamic Neural Networks

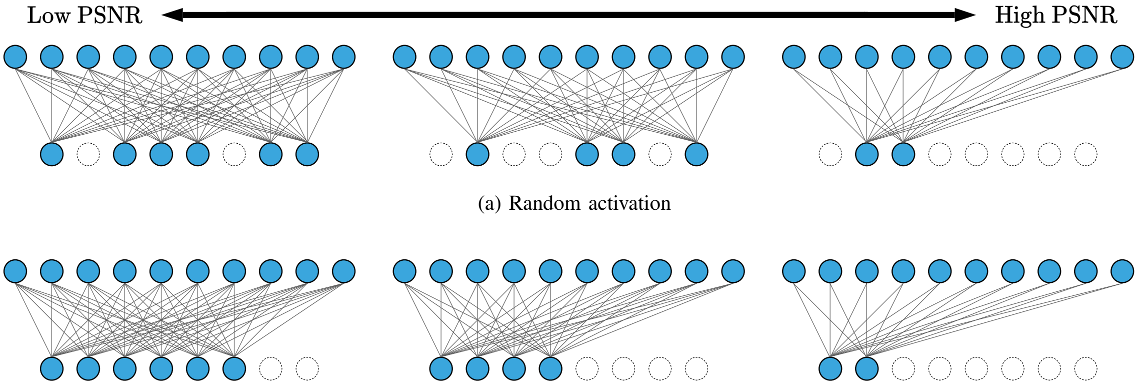
Dynamic neural networks are able to adapt their architectures to the given input and are effective in improving the efficiency of the network processing via selective execution. For example, several prior works (e.g. [43], [44], [45]) proposed to learn a binary gating module to adaptively skip layers or prune channels based on the input data. Besides, there are also some variants of dynamic neural networks, including the slimmable neural networks and the "*Once-for-All*" architecture. In particular, inventors of the slimmable neural networks [46] proposed to train a single model to support layers with arbitrary widths; while authors of [18] proposed the "Once-for-All" architecture with a progressive shrinking algorithm that trains one network to support diverse sub-networks. In this work, we employ the idea of selective activation, as shown in Fig. 2, to learn a set of neurons that can adjust the number of activated neurons according to the channel conditions.

B. Selective Activation for Dynamic Channel Conditions

We propose the variable-length variational feature encoding (VL-VFE), which is empowered with the capability of adjusting its output length under different channel conditions. Such kinds of channel-adaptive feature encoding schemes favor the following two properties:

- The activated dimensions of the feature $\boldsymbol{z}$ can be adjusted in the DNN forward propagation according to the channel conditions. More dimensions should be activated during the bad channel conditions and vice versa.
- The activated dimensions start consecutively from the first dimension (shown in Fig. 2b), which avoids transmitting the indexes of the activated dimensions using extra communication resources.

In practical communication systems, the mobile device could be aware of the channel condition via a feedback channel. Therefore, the channel condition can be incorporated in the feature encoding process. Because the amplitude of the encoded feature vector is constrained to 1 by Tanh function, the noise variance σ^2 suffices to represent the PSNR and is adopted as an extra input of the feature encoder. In the training process, the noise variance σ^2 is regarded as a random variable distributed within a range to model the dynamic channel conditions. For simplicity, we sample the channel variance σ^2 from the uniform distribution $p(\sigma^2)$. As the noise variance $p(\sigma^2)$ is independent to the dataset, we have



(a) Random activation

(b) Consecutive activation

Fig. 2. Two types of selective activations: Random activation and consecutive activation. In different channel conditions (e.g., different PSNRs), the same DNN can be executed with different activated dimensions to balance the achievable inference performance and the incurred communication overhead. Random activation does not require the dimensions to be activated in order, while consecutive activation forces the activated dimensions to be consecutive starting from the first dimension.

$p(\boldsymbol{x},\boldsymbol{y},\sigma^2) = p(\boldsymbol{x},\boldsymbol{y})p(\sigma^2)$. The loss function in (3) is thus revised as follows:

$$\widetilde{\mathcal{L}}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta}) = \mathbf{E}_{p(\mathbf{x},\boldsymbol{y},\sigma^2)}\Big\{\mathbf{E}_{p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x},\sigma^2)}\left[-\log q_{\boldsymbol{\theta}}(\boldsymbol{y}|\hat{z})\right] + \beta D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x},\sigma^2)\|q(\hat{z})\right)\Big\}. \tag{9}$$

Similarly, we adopt Monte Carlo sampling as in (4) to estimate $\widetilde{\mathcal{L}}_{VIB}$. The formula is as follows:

$$\widetilde{\mathcal{L}}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta}) \simeq \frac{1}{M}\sum_{m=1}^{M}\left\{\frac{1}{L}\sum_{l=1}^{L}\left[-\log q_{\boldsymbol{\theta}}\left(\boldsymbol{y_m}|\hat{\boldsymbol{z}}_{\boldsymbol{m},\boldsymbol{l}}\right)\right] + \beta D_{KL}\left(p_{\boldsymbol{\phi}}(\hat{z}|\boldsymbol{x_m},\sigma_m^2)\|q(\hat{z})\right)\right\}, \tag{10}$$

where $\hat{\boldsymbol{z}}_{\boldsymbol{m},\boldsymbol{l}} = \boldsymbol{z_m} + \boldsymbol{\epsilon}_{\boldsymbol{m},\boldsymbol{l}}$, $\sigma_m^2 \sim p(\sigma^2)$, and $\boldsymbol{\epsilon}_{\boldsymbol{m},\boldsymbol{l}} \sim \mathcal{N}(\mathbf{0},\sigma_m^2\boldsymbol{I})$, and for a given $\boldsymbol{z_m}$, the channel noise is sampled L times. Then, as the encoding scheme should be channel-adaptive, we have $p_{\boldsymbol{\phi}}\left(\hat{z}|\boldsymbol{x},\sigma^2\right) = \mathcal{N}(\hat{z}|z(\boldsymbol{x};\boldsymbol{\phi},\sigma^2),\sigma^2\boldsymbol{I})$, where the function $z(\boldsymbol{x};\boldsymbol{\phi},\sigma^2)$ determined by the on-device network incorporates σ^2 as an input variable. Hence, the function in (8) is modified as follows:

$$z_i = \mathrm{Tanh}\left(\gamma_i(\sigma^2)\frac{\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}}{\|\widetilde{\boldsymbol{W}}_{\boldsymbol{i}\cdot}\|_2}\boldsymbol{f}(\boldsymbol{x})\right), \tag{11}$$

where the dimension importance $\gamma_i(\sigma^2)$ (i.e., the i-th element in $\boldsymbol{\gamma}(\sigma^2)$) is the function of the channel condition (i.e., channel noise variance σ^2). Rather than directly training a gating network to control the activated dimensions like other dynamic neural networks (e.g., [43], [44], [45]), $\boldsymbol{\gamma}(\sigma^2)$ can adaptively prune the redundant dimensions in the encoded feature vector for different σ^2 due to the intrinsic sparsity discussed in Section III. As a result, in the device-edge co-inference system, the activated dimensions of the encoded feature vector can be easily decided by setting a threshold for $\boldsymbol{\gamma}(\sigma^2)$. Besides, as VL-VFE needs to meet the consecutive activation property, we define the function $\boldsymbol{\gamma}(\sigma^2)$ to induce a particular group sparsity pattern, and for the i-th element $\gamma_i(\sigma^2)$, the expression is constructed as follows:

Algorithm 2 Training Variable-Length Variational Feature Enoding (VL-VFE)

```
Input: $T$ (number of iterations), $n$ (number of output dimension of encoder), $L$ (number of channel noise samples per datapoint), batch size $M$, noise variance distribution $p(\sigma^2)$, and threshod $\gamma_0$.
while epoch $t = 1$ to $T$ do
    Get a mini-batch of data $\{(\boldsymbol{x_m},\boldsymbol{y_m})\}_{m=1}^M$
    Sample the channel variance $\{\sigma_m^2\}_{m=1}^M \sim p(\sigma^2)$
    Compute the encoded feature vector $\{\boldsymbol{z_m}\}_{m=1}^M$ based on (11)
    while $m = 1$ to $M$ do
        Sample the channel noise $\{\boldsymbol{\epsilon}_{\boldsymbol{m},\boldsymbol{l}}\}_{l=1}^L \sim \mathcal{N}(0,\sigma_m^2\boldsymbol{I})$
        while $i = 1$ to $n$ do
            if $\gamma_i(\sigma_m^2) \le \gamma_0$ then
                Deactivate the $i$-th dimension of $z_m$ in this epoch
            end if
        end while
    end while
    Compute the appropriate KL-divergence based on (6)
    Compute loss $\widetilde{\mathcal{L}}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta})$ based on (10)
    Update parameters $\boldsymbol{\phi},\boldsymbol{\theta}$ through backpropagation
end while
```

$$\gamma_i(\sigma^2) = \sum_{j=i}^{n} g_j(\sigma^2), \tag{12}$$

where $g_j(\cdot)$ denotes the j-th output dimension of the function $\boldsymbol{g}(\cdot)$, which is parameterized by a lightweight multi-layer perceptron (MLP). By constraining the range of parameters in the MLP, each function $g_j(\sigma^2)$ can be a non-negative increasing function, which naturally leads to $\gamma_i(\sigma^2) \ge \gamma_j(\sigma^2), \forall j > i$ and $\gamma_i(\sigma^2) \ge \gamma_i(\bar{\sigma}^2), \forall \sigma^2 \ge \bar{\sigma}^2$. Therefore, given a threshold γ_0, the VL-VFE method summarized in Algorithm 2 can activate the dimensions consecutively, and more dimensions can be activated during the adverse channel conditions. Details of the MLP structure and parameter constraints are deferred to Appendix B.

C. Training Procedure for the Dynamic Neural Network

To train a dynamic neural network with the selective activation under different channel conditions, we naturally average losses sampled from different cases. In each training iteration, for simplicity, we sample σ^2 from the possible PSNR range. Different from the training procedure in Algorithm 1, VL-VFE deactivate each dimension with $\gamma_i\left(\sigma^2\right) \leq \gamma_0$, rather than permanently pruning it, as the function $\boldsymbol{\gamma}(\sigma^2)$ is not stable until convergence. More details about the algorithm are summarized in Algorithm 2.

V. Performance Evaluation

In this section, we evaluate the performance of the proposed task-oriented communication schemes on image classification tasks and investigate the rate-distortion tradeoff for both static and dynamic channel conditions. An ablation study is also conducted to illustrate the importance of an appropriate choice of the variational prior distribution discussed in Section III, i.e., a sparsity-inducing prior distribution can force some dimensions of the encoded feature vector to zero without over-shrinking other dimensions.

A. Experimental Setup

1) Datasets: In this section, we select two benchmark datasets for image classification, including MNIST [47] and CIFAR-10 [48]. The MNIST dataset of handwritten digits from “0” to “9” has a training set of 60,000 sample images and a test set of 10,000 sample images. The CIFAR-10 dataset consists of 60,000 color images in 10 classes with 5,000 training images per class and 10,000 test images. In Appendix D, we further test the performance of the proposed methods on the Tiny Imagenet dataset [49].

2) Baselines: We compare the proposed methods against two learning-based communication schemes for device-edge co-inference, including **DeepJSCC** [8], [29] and **learning-based Quantization** [50].

- **DeepJSCC**: DeepJSCC is a learning-based JSCC method, which maps the input data directly to the channel symbols via a JSCC encoder. We set the loss function of DeepJSCC to cross-entropy, and its communication cost is proportional to the output dimension of the feature encoder.
- **Learning-based Quantization**: This scheme quantizes the floating-point values in the encoded feature vector into low-precision data representations (e.g., the 2-bit fixed-point format). Such a quantization method imitates the lossy source coding and therefore it requires an extra step of channel coding before transmission for error correction. Note that designing a universally optimal channel coding scheme for different channel conditions in the finite block-length regime is highly nontrivial [51]. For fair comparisons, we assume an adaptive channel coding scheme that achieves the following communication rate:

$$C(P, \sigma^2) = \\ = \min\left\{\log_2\left(1+\sqrt{\frac{2P}{\pi e \sigma^2}}\right), \frac{1}{2}\log_2\left(1+\frac{P}{\sigma^2}\right)\right\} \text{(b.p.c.u)}, \tag{13}$$

where $\frac{P}{\sigma^2}$ is the PSNR. This formula was shown to be a tight upper bound on the capacity of the amplitude-limited scalar Gaussian channel in [52].

TABLE I
The DNN structure for MNIST classification task.

	Layer	Output Dimensions
On-device Network	Fully-connected Layer + Tanh	n
Server-based Network	Fully-connected Layer + ReLU	1024
	Fully-connected Layer + ReLU	256
	Fully-connected Layer + Softmax	10

TABLE II
The DNN structure for CIFAR-10 classification task.

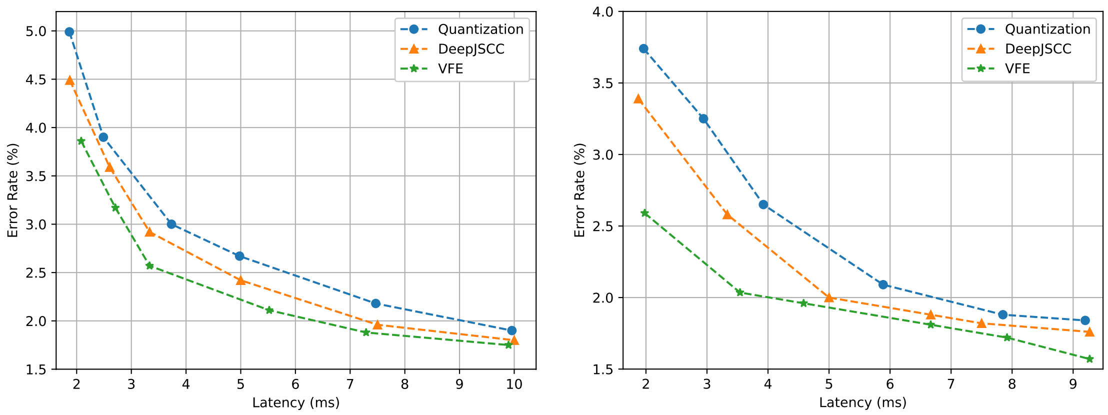
	Layer	Output Dimensions
On-device Network	[Convolutional Layer + ReLU] × 2	128 × 32 × 32
	ResNet Building Block	128 × 16 × 16
	[Convolutional Layer + ReLU] × 2	4 × 4 × 4
	Reshape + Fully-connected Layer + Tanh	n
Server-based Network	Fully-connected Layer + ReLU + Reshape	64
	[Convolutional Layer + ReLU] × 2	512 × 4 × 4
	ResNet Building Block	512 × 4× 4
	Pooling Layer	512
	Fully-connected Layer + Softmax	10

3) Metrics: We mainly concern the rate-distortion tradeoff in task-oriented communication. For the classification tasks, we use the classification accuracy to denote the inference performance (corresponding to “distortion”), and adopt the communication latency as an indicator of “rate”. In the following experiments, we set the bandwidth W as 12.5kHz with a symbol rate of 9,600 Baud, corresponding to the limited bandwidth at the wireless edge.

4) Neural Network Architecture: Carefully designing the on-device network is important due to the limited onboard computation and memory resources. Besides, as the DNN structure affect the inference performance and communication overhead, all methods adopt the same architecture for fair comparisons as follows[3].

- For the MNIST classification experiment, we assume a microcontroller unit (e.g., ARM STM32F4 series) as the mobile device, and its memory (RAM) is less than 0.5 MB. Therefore, we use only one fully-connected layer as the on-device network to meet the memory constraint. At the edge server, we select an MLP as the server-based network. The corresponding network structure is shown in Table I. Note that a 4-layer MLP achieves an error rate of 1.38% as reported in [36].
- For the CIFAR-10 classification task, we assume a single-board computer (e.g., Raspberry Pi series) as the mobile device and adopt ResNet [53] as the backbone for the CIFAR-10 processing, which can achieve the classification accuracy of around 92%. As the single-board computer has much more memory compared to a microcontroller, we deploy convolutional layers on the mobile device to extract a compact representation. Besides, to

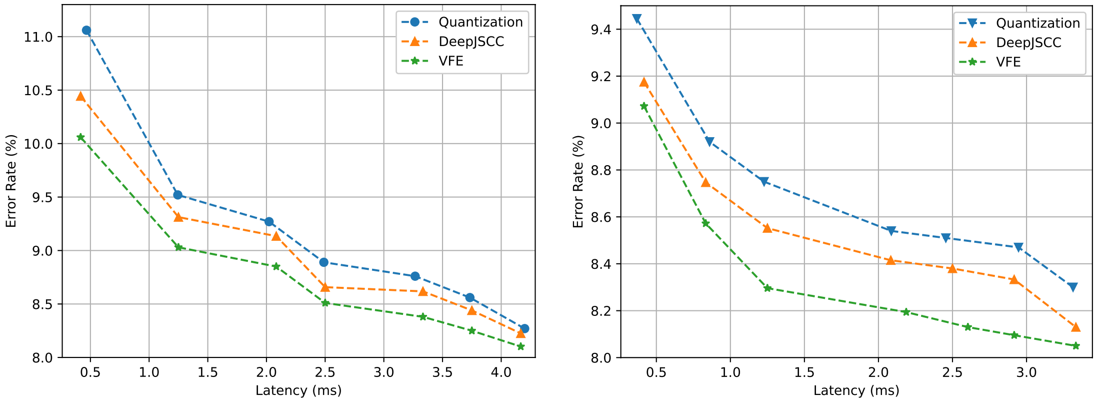
[3] The code is available at github.com/shaojiawei07/VL-VFE.



(a) PSNR = 10 dB

(b) PSNR = 20 dB

Fig. 3. The rate-distortion curves in the MNIST classification task with (a) PSNR = 10 dB and (b) PSNR = 20 dB.



(a) PSNR = 10 dB

(b) PSNR = 20 dB

Fig. 4. The rate-distortion curves in the CIFAR-10 classification task with (a) PSNR = 10 dB and (b) PSNR = 20 dB.

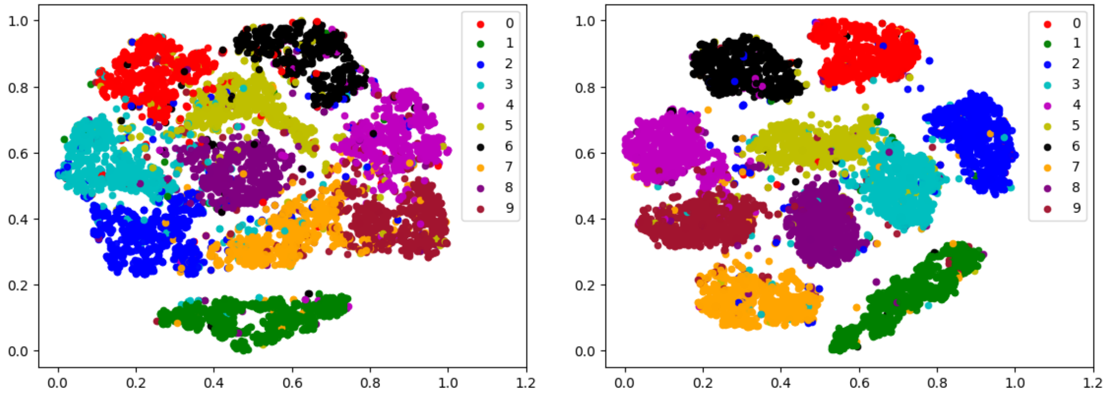
reduce the communication overhead, we add a fully-connected layer at the end of the on-device network to map the intermediate tensor to an n-dimensional encoded feature. Correspondingly, there is a fully-connected layer in the server-based network that maps the received feature vector back to a tensor, and several server-based layers are adopted for further processing. The network structure is shown in Table II.

Since the proposed methods can prune the redundant dimensions in the encoded feature vector, our methods initialize n to 64 or 128 in the following experiments. Moreover, the function $\boldsymbol{g}(\cdot)$ in (12) for variable-length encoding is a 3-layer MLP with 16 hidden units each, which brings negligible computation compared with other computation-intensive modules[4].

B. Results for Static Channel Conditions

In this set of experiments, we assume the wireless channel model has the same value of PSNR in both the training and test phases. Then, we record the inference accuracy achieved with different communication latency to obtain the rate-distortion tradeoff curves. In the proposed VFE method, varying the weighting parameter β can adjust the encoded feature length, where $\beta \in [10^{-4}, 10^{-2}]$ in the MNIST classification, and $\beta \in [5\times 10^{-5}, 10^{-2}]$ in the CIFAR-10 classification. The communication latency of DeepJSCC is determined by the encoded feature dimension n, while for the learning-based Quantization method, the communication latency is determined by the dimension n and the number of quantization levels. Adjusting these parameters affects both the communication latency and accuracy. The rate-distortion tradeoff curves are shown in Fig. 3 and Fig. 4 for the MNIST and CIFAR-10 classification tasks, respectively. It shows that our proposed method outperforms the baselines by achieving a better rate-distortion tradeoff, i.e., with a given latency requirement, a higher classification accuracy is maintained, and vice versa. This is because the proposed VFE method is able to identify and eliminate the redundant dimensions of the encoded feature vector for the task-oriented communication. Besides, we also depict the noisy feature vector $\hat{z}$ in the MNIST classification tasks in Fig. 5 using a two-dimensional t-distributed stochastic neighbor

[4] Note that there is a tradeoff between the on-device computation latency and the communication overhead caused by the complexity of the on-device network [28]. In this paper, as we assume an extreme bandwidth-limited situation, we mainly consider the communication overhead in the experiments.



(a) DeepJSCC: Accuracy = 96.77%, dimension n = 24.

(b) Proposed VFE: Accuracy = 97.39%, dimension n = 24.

Fig. 5. 2-dimensional t-SNE embedding of the received feature in the MNIST classification task with PSNR = 20 dB.

TABLE III
THE CLASSIFICATION ACCURACY UNDER DIFFERENT PSNR WITH COMMUNICATION LATENCY $t \leq 3.25$ MS. THE BASELINE CLASSIFIERS ACHIEVE AN ACCURACY OF 98.64% FOR MNIST CLASSIFICATION AND AN ACCURACY OF AROUND 92% FOR CIFAR-10 CLASSIFICATION.

MNIST	10 dB	15 dB	20 dB	25 dB
DeepJSCC	97.04	97.13	97.45	97.56
Quantization	95.32	95.96	96.81	97.12
Proposed	**97.29**	**97.79**	**98.01**	**98.17**
CIFAR-10	10 dB	15 dB	20 dB	25 dB
DeepJSCC	91.58	91.60	91.67	91.72
Quantization	90.68	91.07	91.53	91.65
Proposed	**91.62**	**91.72**	**91.90**	**92.04**

embedding (t-SNE) [54]. Since the IB principle can preserve less nuisance from the input and make $\hat{z}$ less affected by the channel noise, our VFE method can better distinguish the data from different classes compared with DeepJSCC.

Next, we test the robustness of the proposed method by further evaluating its inference performance over different channel conditions. Particularly, we set a transmission latency tolerance of 3.25 ms and record the best inference accuracy achieved by different schemes[5]. Since the channel achievable rate decreases with the PSNR, it requires the learning-based Quantization method to reduce the encoded data size to meet the latency constraint. The latency constraint can also be translated to an encoded feature vector with less than 32 dimensions for both the VFE method and DeepJSCC. Table III shows the classification accuracy under various values of PSNR for the MNIST and CIFAR-10 tasks. It is observed that, our method consistently outperforms the two baselines, implying that the IB framework can effectively identify the task-relevant information in the encoding scheme, and our VFE method is capable of achieving resilient transmission for task-oriented communication.

C. Results for Dynamic Channel Conditions

In this subsection, we evaluate the performance of the proposed VL-VFE method in dynamic channel conditions. We assume the PSNR is changing from 10 to 25 dB. As the peak transmit power is constrained below 1 by the Tanh activation function, it equivalently means that the channel noise variance σ^2 varies in $[3 \times 10^{-3}, 0.1]$. We compare the inference performance between the proposed method and DeepJSCC when testing in a wide range of PSNR. The DeepJSCC is trained with PSNR = 20 dB, and its feature dimension is set to $n = 36$ in the MNIST classification task and $n = 16$ in the CIFAR-10 classification task.

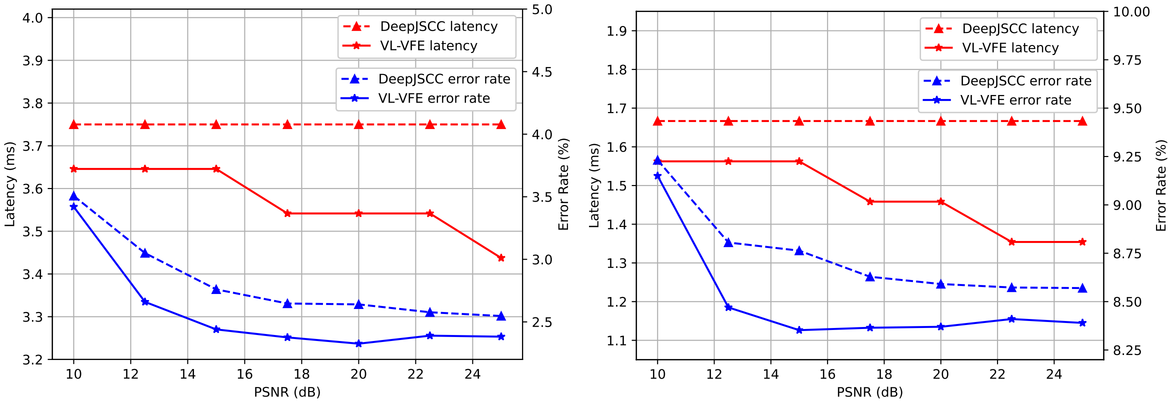
Fig. 6 shows the latency and inference accuracy for the two classification tasks, which illustrates that the proposed VL-VFE method achieves a higher accuracy and lower latency compared with DeepJSCC. The proposed VL-VFE method can adaptively adjust the encoded feature dimension according to the instantaneous channel noise level, and thus it can reduce the communication latency in the high PSNR regime. Specifically, when the channel conditions are unfavorable, VL-VFE tends to activate more dimensions for transmission to make the received feature vector robust to maintain the inference performance, which is analogous to adding redundancy for error correction in conventional channel coding techniques. On the contrary, when the channel conditions are good enough, VL-VFE tends to activate less dimensions to reduce the communication overhead.

Note that in existing communication systems, channel estimation plays a very important role in the performance of the whole system. To evaluate the influence of the non-ideal estimation of the channel noise variance σ^2, we conduct the experiments to test the robustness of the proposed VL-VFE method given inaccurate noise variance $\hat{\sigma}^2$. More details of the experimental settings and results are deferred to Appendix C.

D. Ablation Study

To verify the effectiveness of the log-uniform distribution as the variational prior $q(\hat{z})$ for sparsity induction, we further

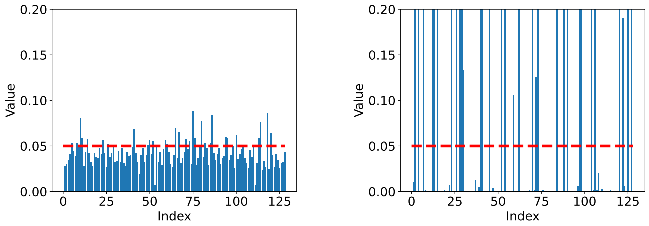
[5] Theoretically, based on the channel capacity bound in (13), transmitting a MNIST image takes around 8 ms when PSNR = 25 dB and 20 ms when PSNR = 10 dB. Similarly, transmitting a CIFAR-10 image takes around 70 ms when PSNR = 25 dB and 180 ms when PSNR = 10 dB.



(a) The MNIST classification task

(b) The CIFAR-10 classification task

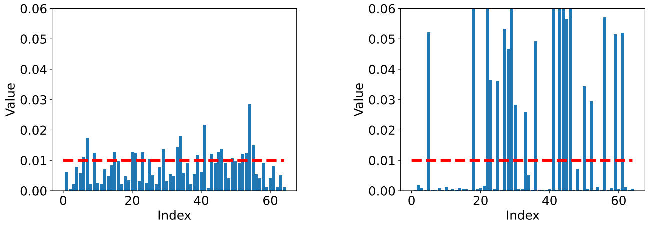
Fig. 6. Communication latency and error rate as a function of the channel PSNR in dynamic channel conditions.



(a) The γ value with a Gaussian distribution as the variational prior. Task accuracy = 95.91 % with 21 activated dimensions.

(b) The γ value with a log-uniform distribution as the variational prior. Task accuracy = 97.99 % with 32 activated dimensions.

Fig. 7. The γ value in the MNIST classification task with (a) a Gaussian distribution as the variational prior and (b) a log-uniform distribution as the variational prior. The red dashed line denotes the pruning threshold $\gamma_0 = 0.05$.



(a) The γ value with a Gaussian distribution as the variational prior. Task accuracy = 91.18 % with 21 activated dimensions.

(b) The γ value with a log-uniform distribution as the variational prior. Task accuracy = 91.83 % with 21 activated dimensions.

Fig. 8. The γ value in the CIFAR-10 classification task with (a) a Gaussian distribution as the variational prior and (b) a log-uniform distribution as the variational prior. The red dashed line denotes the pruning threshold $\gamma_0 = 0.01$.

conduct an ablation study that selects a Gaussian distribution with a diagonal covariance matrix for comparison. Note that the Gaussian distribution is widely used in the previous variational approximation studies (e.g., [40], [35]) as it generally has a closed-form solution. Since the Gaussian distribution is not a parameter-free distribution, the mean value and covariance matrix are optimized in the training process to minimize the KL-divergence $D_{KL}(p(\hat{\boldsymbol{z}}|\boldsymbol{x})\|q(\hat{\boldsymbol{z}}))$. The experiments are conducted for MNIST and CIFAR-10 classification assuming PSNR = 20 dB. The values of $\boldsymbol{\gamma}$ with different variational prior distributions are shown in Fig. 7 and 8. The dashed line corresponds to the value of threshold γ_0 used to prune the dimensions. From these two figures, it can be seen that, although using the Gaussian distribution can also confine some dimensions of $\boldsymbol{\gamma}$ to close-to-zero values, it is prone to shrinking the remaining informative dimensions that eventually results in inference accuracy degradation.

VI. Conclusions

In this work, we investigated task-oriented communication for edge inference, where a low-end edge device transmits the extracted feature vector of a local data sample to a powerful edge server for processing. Our proposed methodology is built upon the information bottleneck (IB) framework, which provides a principled way to characterize and optimize a new rate-distortion tradeoff in edge inference. Assisted by a variational approximation with a log-normal distribution as the variational prior to promote sparsity in the output feature, we obtained a tractable formulation that is amenable to end-to-end training, named variational feature encoding. We further extended our method to develop a variable-length variational feature encoding scheme based on the dynamic neural networks, which makes it adaptive to dynamic channel conditions. The effectiveness of our methods was verified by extensive simulations on image classification datasets.

Through this study, we would like to advocate for rethinking the communication system design for emerging applications such as edge inference. In these applications, communication will keep playing a critical role, but it will serve for the downstream task rather than for data reconstruction as in the classical communication setting. Thus we should take a task-oriented perspective to design the communication module for such applications. New design tools and methodologies will be needed, and the IB framework is a promising candidate. It bridges machine learning and information theory, and leverages theory and tools from both fields. There are many interesting future research directions on this exciting topic, e.g., to apply the IB-based framework to the scenario with multiple devices, to develop a theoretical understanding of the new rate-distortion tradeoff, to improve the robustness of the method, etc.

$$\begin{aligned}
\mathcal{L}_{IB}(\boldsymbol{\phi}) &= -I(Y,\hat{Z}) + \beta I(\hat{Z},X) \\
&= -\int p(\boldsymbol{y}|\hat{\boldsymbol{z}})p(\hat{\boldsymbol{z}}) \log \frac{p(\boldsymbol{y}|\hat{\boldsymbol{z}})}{p(\boldsymbol{y})} d\boldsymbol{y} d\hat{\boldsymbol{z}} + \beta \int p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})p(\boldsymbol{x}) \log \frac{p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})}{p(\hat{\boldsymbol{z}})} d\boldsymbol{x} d\hat{\boldsymbol{z}} \\
&= -\int p(\boldsymbol{y}|\hat{\boldsymbol{z}})p(\hat{\boldsymbol{z}}) \log p(\boldsymbol{y}|\hat{\boldsymbol{z}}) d\boldsymbol{y} d\hat{\boldsymbol{z}} + \beta \int p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})p(\boldsymbol{x}) \log \frac{p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})}{p(\hat{\boldsymbol{z}})} d\boldsymbol{x} d\hat{\boldsymbol{z}} - H(Y) \\
&= \underbrace{-\int p(\boldsymbol{y}|\hat{\boldsymbol{z}})p(\hat{\boldsymbol{z}}) \log q_{\theta}(\boldsymbol{y}|\hat{\boldsymbol{z}}) d\boldsymbol{y} d\hat{\boldsymbol{z}} + \beta \int p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})p(\boldsymbol{x}) \log \frac{p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})}{q(\hat{\boldsymbol{z}})} d\boldsymbol{x} d\hat{\boldsymbol{z}}}_{\mathcal{L}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta})} - \\
&\quad \underbrace{-\int p(\boldsymbol{y}|\hat{\boldsymbol{z}})p(\hat{\boldsymbol{z}}) \log \frac{p(\boldsymbol{y}|\hat{\boldsymbol{z}})}{q_{\theta}(\boldsymbol{y}|\hat{\boldsymbol{z}})} d\boldsymbol{y} d\hat{\boldsymbol{z}}}_{-D_{KL}(p(\boldsymbol{y}|\hat{\boldsymbol{z}})\|q_{\boldsymbol{\theta}}(\boldsymbol{y}|\hat{\boldsymbol{z}}))\leq 0} - \beta \underbrace{\int p_{\boldsymbol{\phi}}(\hat{\boldsymbol{z}}|\boldsymbol{x})p(\boldsymbol{x}) \log \frac{p(\hat{\boldsymbol{z}})}{q(\hat{\boldsymbol{z}})} d\boldsymbol{x} d\hat{\boldsymbol{z}}}_{-D_{KL}(p(\hat{\boldsymbol{z}})\|q(\hat{\boldsymbol{z}}))\leq 0} - \underbrace{H(Y)}_{\text{constant}}.
\end{aligned} \tag{14}$$

Appendix A
Derivation of the Variational Upper Bound

Recall that the IB objective in (2) has the form $\mathcal{L}_{IB}(\boldsymbol{\phi}) = -I(\hat{Z},Y) + \beta I(\hat{Z},X)$. Writing it out in full, the derivation is show in (14). $\mathcal{L}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta})$ in this formulation is the VIB objective function in (3). As the KL-divergence is nonnegative and the entropy of Y is a constant, $\mathcal{L}_{VIB}(\boldsymbol{\phi},\boldsymbol{\theta})$ is a variational upper bound of the IB objective $\mathcal{L}_{IB}(\boldsymbol{\phi})$.

Appendix B
MLP Structure of the Function $\boldsymbol{g}(\sigma^2)$

We parameterize $\boldsymbol{g}(\sigma^2)$ by a K-layer MLP, and thus it can be written as a composition of K non-linear functions:

$$\boldsymbol{g}(\sigma^2) = \boldsymbol{h_K} \circ \boldsymbol{h_{K-1}} \cdots \boldsymbol{h_1}(\sigma^2),$$

where $\boldsymbol{h_k}$ represents the k-th layer in the MLP and has $\boldsymbol{h_k}(\boldsymbol{x}) = \tanh(\boldsymbol{W^{(k)}x})$[6]. To maintain the desired properties of the proposed VL-VFE method, each function $g_j(\sigma^2)$ (the j-th output dimension of the vector function $\boldsymbol{g}(\sigma^2)$ should be non-negative and increase with the noise variance σ^2. Therefore, functions $g_j(\sigma^2)$ should satisfy the following constraints:

$$g_j(\sigma^2) \geq 0; \quad g_j'(\sigma^2) = \frac{\partial g_j(\sigma^2)}{\partial \sigma^2} \geq 0.$$

The function $g(\sigma_j^2)$ can be writtern as follows:

$$g_j(\sigma^2) = h_{K,j} \circ \boldsymbol{h_{K-1}} \cdots \boldsymbol{h_1}(\sigma^2),$$

where $h_{K,j}$ is j-th output dimension of $\boldsymbol{h_K}$. The derivative of $g_j(\sigma^2)$ can be obtained using the chain rule:

$$g_j'(\sigma^2) = \boldsymbol{h'_{K,j}} \circ \boldsymbol{h'_{K-1}} \cdots \boldsymbol{h'_1}(\sigma^2),$$

where we denote the Jacobian matrix of $\boldsymbol{h_k}$ as $\boldsymbol{h'_k}$, and $\boldsymbol{h'_{K,j}}$ is the j-th row of $\boldsymbol{h'_K}$. The derivatives work out as follows:

$$\boldsymbol{h'_k x} = \operatorname{diag}\left(\tanh'\left(\boldsymbol{W^{(k)}x}\right)\right) \cdot \boldsymbol{W^{(k)}}.$$

To guarantee that each $g_j(\sigma^2)$ is a non-negative increasing function, we set $\boldsymbol{W^{(k)}} = \operatorname{abs}(\widehat{\boldsymbol{W}}^{(k)})$, which means that $g_j(\sigma^2)$ outputs a non-negative value, and all entries in Jacobian matrices are non-negative[7].

Appendix C
Robustness of the VL-VFE method given inaccurate channel noise variance

We conduct the experiments to evaluate the robustness of the proposed method given inaccurate channel noise variance. In particular, by assuming m pilot symbols are transmitted from the mobile device for noise variance estimation, and adopting the uniformly minimum-variance unbiased estimator, the noise variance is estimated as $\hat{\sigma}^2 = \frac{1}{m-1}\sum_i^m(\hat{z}_{i,p} - z_{i,p})^2$, where $\hat{z}_{i,p}$ and $z_{i,p}$ correspond to the i-th transmitted and received pilot symbols, respectively. It can be easily verified that $\mathbf{E}\left[\hat{\sigma}^2\right] = \sigma^2$ and $p(\hat{\sigma}^2|\sigma^2) = \frac{\sigma^2}{m-1}\chi^2(m)$, where $\chi^2(m)$ denotes the chi-square distribution with m degrees of freedom. The variance of $\hat{\sigma}^2$ reduces as m increases, i.e., the noise variance estimation becomes more accurate. With the inaccurate noise variance $\hat{\sigma}^2$ at the transmitter, we test the performance of the proposed VL-VFE method based on the CIFAR-10 image classification task for the following three cases:

- VL-VFE ($m = 0$): This corresponds to the case that the transmitter has no knowledge about the noise variance, and the PSNR is set to be 10 dB for feature encoding;
- VL-VFE ($m = 8$): The noise variance is estimated via 8 pilot symbols, which corresponds to the case of imperfect channel knowledge for feature encoding;
- VL-VFE ($m = \infty$): This corresponds to the case of perfect channel knowledge for feature encoding.

Following the experimental settings in Section V, we also adopt DeepJSCC as the baseline in comparison. The experimental results on the error rate and feature transmission latency are shown in Fig. 9 and Fig. 10, with the new findings summarized as follows:

- The proposed method achieves lower communication latency compared with DeepJSCC in all the three cases in the dynamic channel conditions;
- While reducing the number of pilot symbols to 8 incurs performance degradation to the proposed method due to the inaccurate noise variance, the proposed method still achieves a much better rate-distortion tradeoff than DeepJSCC;
- Even when the transmitter has no knowledge of the noise variance, i.e., $m = 0$, the proposed method still shows a comparable performance as DeepJSCC.

[6] $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ and $\tanh'(x) = 1 - \tanh(x)$. For simplicity, we define $\tanh(x)$ and $\tanh'(x)$ as element-wise functions.

[7] $\operatorname{abs}(\cdot)$ denotes the element-wise absolute function. $\widehat{\boldsymbol{W}}^{(k)}$ are the actual parameters in the K-layer MLP.

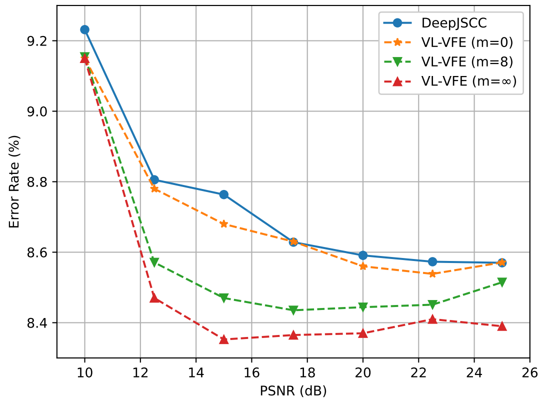


Fig. 9. Error rate as a function of the channel PSNR in dynamic channel conditions.

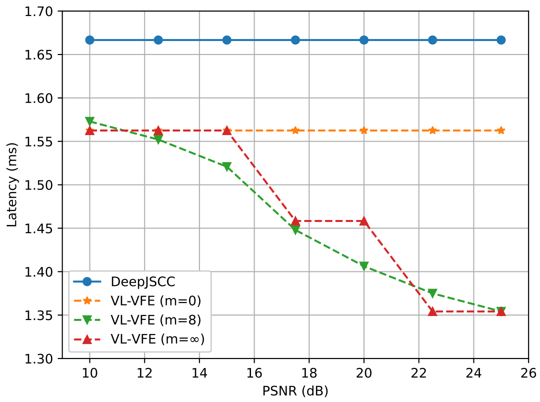
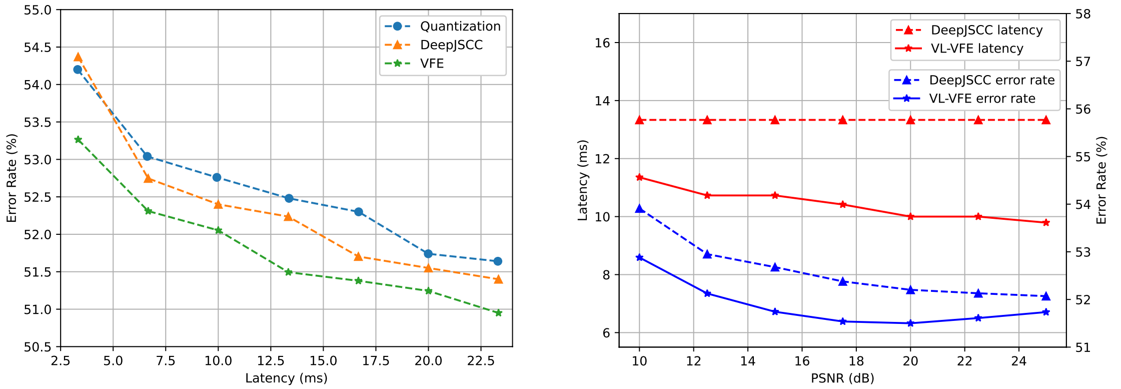


Fig. 10. Communication latency as a function of the channel PSNR in dynamic channel conditions.



(a) The rate-distortion curves with PSNR = 20 dB.

(b) Communication latency and error rate as a function of the channel PSNR in dynamic channel conditions.

Fig. 11. Experimental results of the classification task on the Tiny ImageNet dataset.

In conclusion, these experimental results demonstrate that our proposed method is robust against the inaccurate channel knowledge, i.e., the channel noise variance.

Appendix D
Additional Experiments on Tiny ImangeNet Dataset

We further evaluate the performance of the proposed variational feature encoding (VFE) method and variable-length variational feature encoding (VL-VFE) method on the Tiny ImageNet classification task [49]. Tiny ImageNet contains 200 image classes, a training dataset of 100,000 images, and a validation dataset of 10,000 images. All images are of size 64 × 64. We select the ResNet18 as the backbone for this task, which can achieve the top-1 accuracy of around 50.5%. The whole neural network structure is shown in Table IV. Following the basic settings in Section V, we compare our proposed methods with **DeepJSCC** and **Learning-based Quantization**. The initialized feature dimension of the proposed methods is 224 in this set of experiments. Fig. 11a shows the rate-distortion curves in the static channel condition (PSNR = 20 dB), where our VFE method changes the feature dimension by adjusting the β value in the range of $[10^{-4}, 10^{-3}]$. Similar to the previous results on the MNIST and CIFAR-10 datasets, our proposed VFE method outperforms the baselines by achieving a better rate-distortion tradeoff. In the dynamic channel conditions, we set $\beta = 5 \times 10^{-4}$ in the training phase when PSNR is changing from 10 dB to 25 dB. Fig. 11b shows that the proposed VL-VFE method achieves higher accuracy and lower latency compared with DeepJSCC.

TABLE IV
The DNN structure for Tiny ImageNet classification task.

	Layer	Output Dimensions
On-device Network	[ResNet Building Block] × 5	512 × 4 × 4
	Pooling + Fully-connected Layer + Tanh	n
Server-based Network	Fully-connected Layer + ReLU	512
	Fully-connected Layer + Softmax	200

References

[1] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. Int. Conf. Acoust. Speech Process.*, Vancouver, Canada, May 2013, pp. 6645–6649.

[2] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. Int. Conf. Mach. Learn.*, Helsinki, Finland, Jul. 2008, pp. 160–167.

[3] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput. intell. neurosci.*, vol. 2018, Feb. 2018.

[4] X. Hou, S. Dey, J. Zhang, and M. Budagavi, "Predictive view generation to enable mobile 360-degree and VR experiences," in *Proc. Morning Workshop VR AR Netw.*, Budapest, Hungary, Aug. 2018, pp. 20–26.

[5] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3039–3071, Jul. 2019.

[6] J. Downey, B. Hilburn, T. O'Shea, and N. West, "Machine learning remakes radio," *IEEE Spectr.*, vol. 57, no. 5, pp. 35–39, Apr. 2020.

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

[8] E. Bourtsoulatze, D. Burth Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 3, pp. 567–579, May 2019.

[9] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Singal Process.*, vol. 67, no. 10, pp. 2554–2564, Feb. 2019.

[10] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 101–115, Jan. 2021.

[11] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.

[12] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Toward an intelligent edge: wireless communication meets machine learning," *IEEE Commun. Mag.*, vol. 58, no. 1, pp. 19–25, Jan. 2020.

[13] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge AI: Algorithms and systems," *IEEE Commun. Surv. Tut.*, vol. 22, no. 4, pp. 2167–2191, Jul. 2020.

[14] E. Li, Z. Zhou, and X. Chen, "Edge intelligence: On-demand deep learning model co-inference with device-edge synergy," in *Proc. Workshop Mobile Edge Commun.*, Budapest, Hungary, Aug. 2018, pp. 31–36.

[15] A. I. Maqueda, A. Loquercio, G. Gallego, N. García, and D. Scaramuzza, "Event-based vision meets deep learning on steering prediction for self-driving cars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 5419–5427.

[16] X. Hou, S. Dey, J. Zhang, and M. Budagavi, "Predictive view generation to enable mobile 360-degree and vr experiences," in *Proc. Morning Workshop VR AR Netw.*, Budapest, Hungary, Aug. 2018, pp. 20–26.

[17] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *Proc. Annu. Int. Conf. Mobile Comput. Netw.*, Los Cabos, Mexico, Oct. 2019, pp. 1–16.

[18] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once for all: Train one network and specialize it for efficient deployment," in *Proc. Int. Conf. Learn. Represent.*, Addis Ababa, Ethiopia, Apr. 2020.

[19] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Comput. Archit. News*, vol. 45, no. 1, pp. 615–629, Apr. 2017.

[20] H. Li, C. Hu, J. Jiang, Z. Wang, Y. Wen, and W. Zhu, "JALAD: Joint accuracy-and latency-aware deep structure decoupling for edge-cloud execution," in *Proc. Int. Conf. Parallel Distrib. Syst.*, Singapore, Dec. 2018, pp. 671–678.

[21] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *Proc. Annu. Allerton Conf. Commun. Control Comput.*, Monticello, IL, USA, Oct. 2000.

[22] N. Farsad, M. Rao, and A. Goldsmith, "Deep learning for joint source-channel coding of text," in *Proc. Int. Conf. Acoust. Speech Process.*, Calgary, Canada, Apr. 2018, pp. 2326–2330.

[23] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Mar. 2013.

[24] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 2082–2090.

[25] W. Shi, Y. Hou, S. Zhou, Z. Niu, Y. Zhang, and L. Geng, "Improving device-edge cooperative inference of deep learning via 2-step pruning," in *Proc. IEEE Conf. Comput. Commun. Workshop*, 2019, pp. 1–6.

[26] J. Shao and J. Zhang, "Bottlenet++: An end-to-end approach for feature compression in device-edge co-inference systems," in *Proc. Int. Conf. Commun. Workshop*, Dublin, Ireland, Jun. 2020, pp. 1–6.

[27] M. Jankowski, D. Gündüz, and K. Mikolajczyk, "Wireless image retrieval at the edge," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 89–100, May 2021.

[28] J. Shao and J. Zhang, "Communication-computation trade-off in resource-constrained edge inference," *IEEE Commun. Mag.*, Dec. 2020.

[29] M. Jankowski, D. Gündüz, and K. Mikolajczyk, "Deep joint source-channel coding for wireless image retrieval," in *Proc. Int. Conf. Acoust. Speech Process.*, Barcelona, Spain, May 2020, pp. 5070–5074.

[30] J. Shao, H. Zhang, Y. Mao, and J. Zhang, "Branchy-GNN: a device-edge co-inference framework for efficient point cloud processing," 2020. [Online]. Avaliable: https://arxiv.org/abs/2011.02422.

[31] K. Choi, K. Tatwawadi, A. Grover, T. Weissman, and S. Ermon, "Neural joint source-channel coding," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, Jun. 2019, pp. 1182–1192.

[32] R. Dobrushin and B. Tsybakov, "Information transmission with additional noise," *IRE Trans. Inf. Theory*, vol. 8, no. 5, pp. 293–304, Sep. 1962.

[33] Z. Goldfeld and Y. Polyanskiy, "The information bottleneck problem and its applications in machine learning," *IEEE J. Sel. Areas Inf. Theory*, Apr. 2020.

[34] A. Zaidi, I. Estella-Aguerri *et al.*, "On the information bottleneck problems: Models, connections, applications and information theoretic views," *Entropy*, vol. 22, no. 2, p. 151, Jan. 2020.

[35] A. Achille and S. Soatto, "Information dropout: Learning optimal representations through noisy computation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2897–2905, Jan. 2018.

[36] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," in *Proc. Int. Conf. Learn. Represent.*, Toulon, France, Apr. 2017.

[37] S. Dörner, S. Cammerer, J. Hoydis, and S. Brink, "Deep learning based communication over the air," *IEEE J. Selected Topics Singal Process.*, vol. 12, no. 1, pp. 132–143, Dec. 2018.

[38] T. M. Cover and J. A. Thomas, *Elements of information theory*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2012.

[39] Z. Wang and D. W. Scott, "Nonparametric density estimation for high-dimensional data—algorithms and applications," *Wiley Interdiscip. Rev. Comput. Statist.*, vol. 11, no. 4, p. 1461, Apr. 2019.

[40] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. Int. Conf. Learn. Represent.*, Banff, Canada, Apr. 2014.

[41] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Proc. Adv. Neural Inf. Process. Syst.*, San Diego, CA, USA, May 2015, pp. 2575–2583.

[42] D. Molchanov, A. Ashukha, and D. Vetrov, "Variational dropout sparsifies deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, Sydney, Australia, Aug. 2017, pp. 2498–2507.

[43] X. Wang, F. Yu, Z.-Y. Dou, T. Darrell, and J. E. Gonzalez, "Skipnet: Learning dynamic routing in convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 409–424.

[44] Z. Wu, T. Nagarajan, A. Kumar, S. Rennie, L. S. Davis, K. Grauman, and R. Feris, "Blockdrop: Dynamic inference paths in residual networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 8817–8826.

[45] Z. Chen, Y. Li, S. Bengio, and S. Si, "You look twice: Gaternet for dynamic filter selection in cnns," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Seoul, Korea, Oct. 2019, pp. 9172–9180.

[46] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[47] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, May 1998.

[48] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[49] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," 2015. [Online]. Available: http://cs231n.stanford.edu/reports/2017/pdfs/930.pdf.

[50] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6869–6898, Jan 2017.

[51] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[52] A. L. McKellips, "Simple tight bounds on capacity for the peak-limited discrete-time channel," in *Proc. Int. Symp. Inf. Theory.*, Chicago, IL, USA, Jun. 2004, pp. 348–348.

[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[54] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, Nov. 2008.