A Rate-Distortion View of Uncertainty Quantification

Ifigeneia Apostolopoulou [1] **Benjamin Eysenbach** [2] **Frank Nielsen** [3] **Artur Dubrawski** [1]

Abstract

In supervised learning, understanding an input's proximity to the training data can help a model decide whether it has sufficient evidence for reaching a reliable prediction. While powerful probabilistic models such as Gaussian Processes naturally have this property, deep neural networks often lack it. In this paper, we introduce Distance Aware Bottleneck (DAB), i.e., a new method for enriching deep neural networks with this property. Building on prior information bottleneck approaches, our method learns a codebook that stores a compressed representation of all inputs seen during training. The distance of a new example from this codebook can serve as an uncertainty estimate for the example. The resulting model is simple to train and provides deterministic uncertainty estimates by a single forward pass. Finally, our method achieves better out-of-distribution (OOD) detection and misclassification prediction than prior methods, including expensive ensemble methods, deep kernel Gaussian Processes, and approaches based on the standard information bottleneck.

1. Introduction

Deep learning models that "know what they know" are becoming increasingly useful since they can better understand when to make confident predictions and when to ask for human help (Kivlichan et al., 2021). Early approaches to uncertainty estimation built from probabilistic approaches tailored to deep neural networks (DNNs) (Blundell et al., 2015; Osawa et al., 2019; Gal & Ghahramani, 2016) or deep ensembles (Lakshminarayanan et al., 2017; Wilson & Izmailov, 2020). A shared characteristic of these methods is that they require multiple model samples to produce a reliable uncertainty estimate. Despite growing research interest in uncertainty quantification, we still lack reliable and efficient methods for real-world ML deployment.

[1]Machine Learning Department, AutonLab, Carnegie Mellon University [2]Computer Science Department, Princeton University [3]Sony Computer Science Laboratories Inc., Tokyo, Japan. Correspondence to: Ifigeneia Apostolopoulou <iapostol@andrew.cmu.edu, ifiaposto@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

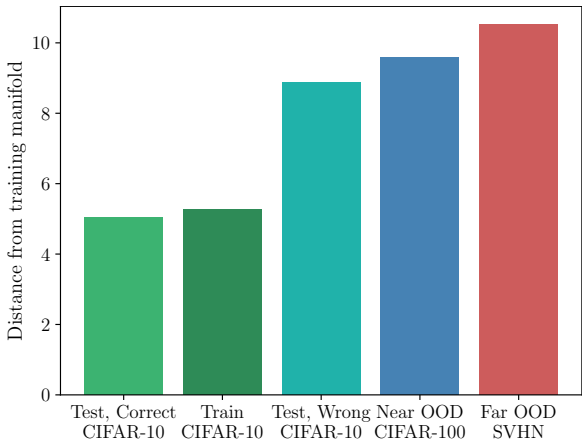


Figure 1: **Distance awareness for principled uncertainty quantification.** A distance-aware model can measure the distance between input examples and the training examples. Our method learns distances where misclassified datapoints, semantic (near OOD), and domain (far OOD) deviations can be identified by larger distances. Our method learns and uses a codebook for representing the training dataset. Here, we report distances from a codebook trained on CIFAR-10.

A recently emerged class of scalable uncertainty estimation methods, the *Deterministic Uncertainty Methods* (DUMs) (Postels et al., 2022; Charpentier et al., 2023), affords uncertainty estimates with a single forward-pass. These methods are *distance-aware* (Liu et al., 2020) since they can quantify a distance score or measure of a new test example from previously trained-upon datapoints. Distance awareness renders DUMs a principled and theoretically motivated (Liu et al., 2020; 2023) solution to uncertainty quantification. In particular, the distance score can indicate Out-Of-Distribution (OOD) examples of varying dissimilarity from the training datapoints or in-distribution areas where the model fails to generalize (Fig. 1).

Existing DUMs are usually tied to specific regularization techniques (Miyato et al., 2018; Gulrajani et al., 2017) to mitigate feature collapse. Although such additional weight constraints help these methods reach state-of-the-art OOD detection results, they may undermine their calibration, i.e., how well a DNN can predict its incorrectness (Postels et al., 2022). Moreover, in the absence of similar constraints in

large, pre-trained models, integration of current DUMs into industrial applications becomes difficult.

In this work, we seek to improve the quality of uncertainty estimates using a single-model, deterministic characterization. The key contributions of this paper are as follows:

- We formulate uncertainty quantification as the computation of a rate-distortion function to obtain a compressed representation of the training dataset. This representation is a set of prototypes defined as centroids of the training datapoints with respect to a distance measure. The expected distance of a datapoint from the centroids provides model's uncertainty for the datapoint (Fig. 1).
- We take a "meta-probabilistic" perspective to the rate-distortion problem. In particular, the distortion function operates on distributions of embeddings and corresponds to a statistical distance (Fig. 2). To do so, we use the Information Bottleneck (IB) framework. The proposed formulation, the Distance Aware Bottleneck (DAB), jointly regularizes DNN's representations and renders it distance-aware.
- We design and qualitatively verify a practical deep learning algorithm that is based on successive estimates of the rate-distortion function to identify the centroids of the training data (Algorithm 1).
- We show experimentally that our method can detect both OOD samples and misclassified samples. In particular, DAB outperforms baselines when used for OOD tasks and closes the gap between single forward pass methods and expensive ensembles in terms of calibration (Tables 2, 4).
- Finally, we show that DAB can be trained and applied post-hoc to a large, pre-trained feature extractor offering similar advantages for challenging and large-scale datasets (Table 5).

2. Related Work

In this section, we provide an overview of existing DUMs and relate them to the proposed model. Most competitive DUMs can be taxonomized as Gaussian Process models or cluster-based approaches.

Gaussian Processes (GPs) are intrinsically distance-aware models since they are defined by a kernel function that quantifies similarity to the training datapoints. SNGP (Liu et al., 2020; 2023) relies on a Laplace approximation of the GP based on Random Fourier Features (RFF) (Rahimi & Recht, 2007). DUE (Van Amersfoort et al., 2021) uses the inducing point GP approximation (Titsias, 2009). In Table 1, we provide some analogies between Gaussian Processes and the model proposed in this work. Both SNGP and DUE enforce bi-Lipschitz constraints on the network by spectral normalization (Miyato et al., 2018) to encourage sensitivity and smoothness of the extracted features.

Table 1: **Analogies between GP and DAB.**

	Gaussian Process	**Distance Aware Bottleneck**
Compression of the training dataset $\mathcal{D}_{\text{train}}$	Inducing Points	Codebook
Feature space	$\mathbb{R}^d$	parameter space $\boldsymbol{\Theta}$ of a family of distributions $\mathcal{P} = \{p(\boldsymbol{z};\boldsymbol{\theta}) \mid \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$
Distance measure	Euclidean norm	Statistical distance

In contrast, our work builds on IB methods (Alemi et al., 2018) to avoid feature collapse. IB methods regularize the network by encouraging it to learn informative representations. Therefore, they are simple to implement and train. However, prior IB methods (Alemi et al., 2018) cannot sufficiently represent large and complex datasets. In this paper, we revise and augment prior IBs with a codebook capable of coding high-dimensional and multi-modal training distributions. The training is facilitated by a learning algorithm (Section 5.2) which, along with the gradient updates, matches the training examples with the entries of the codebook.

More closely related to our work is DUQ (Van Amersfoort et al., 2020). Similar to our work, DUQ quantifies uncertainty as the distance from centroids responsible for representing the training data. The distance is computed in terms of a Radial Basis Function (RBF) kernel. In contrast to our work, DUQ is trained to minimize a binary cross entropy loss function. This function assigns datapoints to clusters in a supervised manner. Therefore, the number of centroids is hardwired to the number of classes. This restriction renders the deployment of the model to regression tasks or classification tasks with a large number of classes difficult. On the other hand, our model provides a unified notion of uncertainty for both regression and classification tasks. Estimating regression uncertainty is important in many machine learning subfields. For example, in deep reinforcement learning uncertainty over the Q-values can be leveraged for efficient exploration or risk estimation (Osband et al., 2016; Lee et al., 2021; Fujimoto et al., 2018; Wu et al., 2021). Effective DUMs, such as our model, could mend the current lack of both efficient and reliable uncertainty methods in unsupervised learning settings.

Under a broad definition, data augmentation methods (Hendrycks et al., 2020; Pinto et al., 2022b) can also be considered DUMs. They improve network's learned representations by encouraging the model to be sensitive to or invariant against image perturbations. Design of such perturbations, however, requires domain expertise and/or prior knowledge. This requirement makes it difficult to extend existing data augmentation methods to other tasks (such as regression) or modalities (such as text). Here, we focus on principled, distance-aware DUMs, and borrowing terminol-

ogy of Postels et al. (2022); Mukhoti et al. (2023), unless otherwise noted, we use DUMs to refer to distance-aware DUMs. Finally, we note that deep ensembles are included as a benchmark in our experiments, as they represent the current state-of-the-art for uncertainty quantification. However, while simple in concept and implementation, their computational and memory cost are prohibitive.

3. Preliminaries

3.1. Information Bottleneck

The Information Bottleneck (IB) (Tishby et al., 2000) provides an information-theoretic view for balancing the complexity of a stochastic encoder Z for input X [1] and its predictive capacity for the desired output Y. The IB objective is:

$$\min_{\boldsymbol{\theta}} \quad -I(Z, Y; \boldsymbol{\theta}) + \beta\, I(Z, X; \boldsymbol{\theta}), \tag{1}$$

where $\beta \geq 0$ is the trade-off factor between the accuracy term $I(Z, Y; \boldsymbol{\theta})$ and the complexity term $I(Z, X; \boldsymbol{\theta})$. $\boldsymbol{\theta}$ denotes the parameters of the distributional family of encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$[2]. In words, training by Eq. 1 encourages the model to find a representation Z that is maximally expressive about output Y while being maximally compressive about input X. Typically, the mutual information terms in Eq. 1 cannot be computed in closed-form since they involve intractable marginal distributions (Eq. 23, 24). The Variational Information Bottleneck (VIB) (Alemi et al., 2017) considers parametric approximations $m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})$, $q(\boldsymbol{z}; \boldsymbol{\phi})$ to these marginals belonging to a distributional family parametrized by $\boldsymbol{\theta}$ [3] and $\boldsymbol{\phi}$ respectively. The VIB objective (Eq. 26) maximizes a lower bound of $I(Z, Y; \boldsymbol{\theta})$ and minimizes an upper bound of $I(Z, X; \boldsymbol{\theta})$. In this work, we reconsider the complexity term. The upper bound of this term is an expected Kullback-Leibler divergence:

$$\begin{aligned} I(Z, X; \boldsymbol{\theta}) &= \mathbb{E}_X \left[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}), p(\boldsymbol{z})) \right] \\ &\leq \mathbb{E}_X \left[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}), q(\boldsymbol{z}; \boldsymbol{\phi})) \right]. \end{aligned} \tag{2}$$

The expectation in Eq. 2 is taken, in practice, with respect to the empirical distribution of the training dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$:

$$I(Z, X; \boldsymbol{\theta}) \lessapprox \frac{1}{N} \sum_{i=1}^{N} D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta}), q(\boldsymbol{z}; \boldsymbol{\phi})). \tag{3}$$

[1] We denote random variables as X, Y, Z and their instances as $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$.

[2] $\boldsymbol{\theta}$ will represent a function implemented by a neural network. For input $\boldsymbol{x}$ it computes the parameters of the conditional distribution $p(\cdot \mid \boldsymbol{x}; \boldsymbol{\theta})$ in its output. For example, for a Gaussian with diagonal covariance: $\boldsymbol{\theta}(\boldsymbol{x}) = \{\mu(\boldsymbol{x}), \sigma(\boldsymbol{x})\}, \mu(\boldsymbol{x}) \in \mathbb{R}^d, \sigma(\boldsymbol{x}) \in \mathbb{R}^d_{\geq 0}$. Optimization with respect to $\boldsymbol{\theta}$ will refer to optimization with respect to the weights of network $\boldsymbol{\theta}$.

[3] In the rest of the paper, we use $\boldsymbol{\theta}$ to denote the joint set of parameters of encoder and variational decoder.

3.2. Rate Distortion Theory

The rate-distortion theory (Berger, 1971; Berger & Gibson, 1998; Cover, 1999) quantifies the fundamental limit of data compression, i.e., at least how many bits are needed to quantize data coming from a stochastic source given a desired fidelity. Formally, consider random variable $X \sim p(\boldsymbol{x})$ with support set[4] $\mathcal{X}$. Data coming from source X will be compressed by mapping them to a random variable $\hat{X}$ with support set $\hat{\mathcal{X}}$. It is common to refer to $\hat{X}$ as the source code or quantization of X. In this work, we consider a discrete source over $\mathcal{D}_{\text{train}}$ following the empirical distribution. The formal description is deferred to Section 5.1.

The quality of the reconstructed data is assessed using a distortion function $D : \mathcal{X} \times \hat{\mathcal{X}} \to \mathbb{R}^+$. The rate-distortion function is the minimum achievable rate (number of bits) of the quantization scheme for a prescribed level of expected distortion. In Lagrange formulation, it is the problem:

$$R \triangleq \min_{p(\hat{\boldsymbol{x}} \mid \boldsymbol{x})} \quad I(X; \hat{X}) + \alpha \mathbb{E}_{X, \hat{X}}[D(\boldsymbol{x}, \hat{\boldsymbol{x}})], \tag{4}$$

where $\alpha \geq 0$ is the optimal Lagrange multiplier that corresponds to a distortion constraint $\mathbb{E}_{X, \hat{X}}[D(\boldsymbol{x}, \hat{\boldsymbol{x}})] \leq d$.[5] It can be shown that the problem in Eq. 4 is equivalent to a double minimization problem over $p(\hat{\boldsymbol{x}})$, $p(\hat{\boldsymbol{x}} \mid \boldsymbol{x})$ (Lemma 10.8.1 of Cover (1999)). This equivalence enables an alternating minimization algorithm (Csiszár, 1984) – the Blahut–Arimoto (BA) algorithm (Blahut, 1972; Matz & Duhamel, 2004) – for solving R. In practice, numerical computation of the rate-distortion function through the BA algorithm is often infeasible, primarily due to lack of knowledge of the optimal support of $\hat{X}$. The *Rate Distortion Finite Cardinality (RDFC)* formulation (Rose, 1994; Banerjee et al., 2004) simplifies the computation of R by assuming finite support $\hat{\mathcal{X}}$ that is jointly optimized:

$$\begin{aligned} &\min_{\hat{\mathcal{X}}, p(\hat{\boldsymbol{x}} \mid \boldsymbol{x})} \quad I(X, \hat{X}) + \alpha\, \mathbb{E}_{X, \hat{X}}[D(\boldsymbol{x}, \hat{\boldsymbol{x}})] \\ &\text{subject to: } \mid \hat{\mathcal{X}} \mid = k. \end{aligned} \tag{5}$$

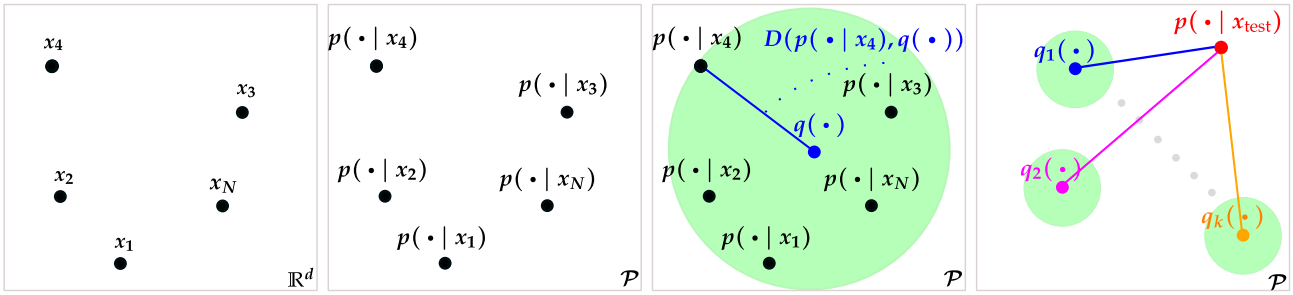
The RDFC objective in Eq. 5 can be greedily estimated by alternating optimization over $\hat{\mathcal{X}}$, $p(\hat{\boldsymbol{x}})$, $p(\hat{\boldsymbol{x}} \mid \boldsymbol{x})$ yielding a solution that is locally optimal (Banerjee et al., 2004).

4. Motivation

The idea behind our approach is visualized in Fig. 2. The crux of our approach is the observation that the variational marginal $q(\boldsymbol{z}; \boldsymbol{\phi})$ in Eq. 2 and Eq. 3 encapsulates all encoders $p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})$ of datapoints in $\mathcal{D}_{\text{train}}$ encountered during training. To see this formally, we introduce a random variable P_X defined by $X \sim p(\boldsymbol{x})$. The value of P_X

[4] Support set of $X \sim p(\boldsymbol{x})$ is the set $\mathcal{X} = \{\boldsymbol{x} : p(\boldsymbol{x}) > 0\}$.

[5] Formally, α is a function of d: $\alpha \equiv \alpha(d)$. However, we omit this dependence for notational brevity.



(a) $\mathcal{D}_{\text{train}}$ as a set of points in the Euclidean space $\mathbb{R}^d$.

(b) $\mathcal{D}_{\text{train}}$ as a set of points in distribution space $\mathcal{P}$.

(c) Support of $\mathcal{D}_{\text{train}}$ as a statistical ball ($k = 1$).

(d) Distance from the codebook of encoders ($k > 1$).

Figure 2: **Overview of DAB.** Uncertainty quantification in DAB is based on compressing the training dataset $\mathcal{D}_{\text{train}}$ by learning a codebook and computing distances from the codebook. The datapoints in $\mathcal{D}_{\text{train}}$, originally lying in $\mathbb{R}^d$ (2a), are embedded into distribution space $\mathcal{P}$ of a parametric family of distributions through their encoders (2b). Compression of $\mathcal{D}_{\text{train}}$ amounts to finding the centroids of the encoders in terms of a statistical distance D (2c). For complex datasets, usually multiple centroids are needed (2d). The uncertainty for a previously unseen test datapoint is quantified by its expected distance from the codebook: $\text{uncertainty}(x_{\text{test}}) = \mathbb{E}[D(p(\boldsymbol{z} \mid \boldsymbol{x}_{\text{test}}; \boldsymbol{\theta}), q_\kappa(\boldsymbol{z}; \boldsymbol{\phi}))]$.

corresponding to $\boldsymbol{x}$ is the encoder's density $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$ (Fig. 2a, 2b). In other words, a value of P_X is itself a probability distribution. From proposition 1 of Banerjee et al. (2005), $\mathbb{E}[P_X]$ is the unique centroid of encoders $p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})$ with respect to *any* Bregman divergence D_f defined by a strictly convex and differentiable function f (Bregman, 1967; Brekelmans & Nielsen, 2022) (def. A.1):

$$\begin{aligned}\mathbb{E}[P_X] &= \frac{1}{N}\sum_{i=1}^{N} p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta}) \\ &= \arg\min_{q(\boldsymbol{z})} \frac{1}{N}\sum_{i=1}^{N} D_f(p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta}), q(\boldsymbol{z})). \qquad (6)\end{aligned}$$

We note that the upper bound in Eq. 3 emerges as a special case of the minimization objective in Eq. 6. This is because the Kullback-Leibler divergence is a Bregman divergence (Azoury & Warmuth, 2001; Nielsen et al., 2007) with the negative entropy as the generator function f (Frigyik et al., 2008; Csiszár, 1995)[6]. Therefore, $q(\boldsymbol{z}; \boldsymbol{\phi})$ in the VIB can also be viewed as a variational centroid of the training datapoints' encoders (Fig. 2c). In this work, we consider learnable parameters $\boldsymbol{\phi}$. Under this view, the role of the regularization term $I(Z, X; \boldsymbol{\theta})$ when upper bounded by Eq. 2 is now twofold: i) it both regularizes encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$ and ii) it learns a distributional centroid $q(\boldsymbol{z}; \boldsymbol{\phi})$ for encoders $p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})$ of training examples $\boldsymbol{x}_i$.

For complex data, it usually does not suffice to represent $\{p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})\}_{i=1}^N$ by a single distribution $q(\boldsymbol{z}; \boldsymbol{\phi})$. Therefore, we will need to learn a collection (codebook) of k centroids $\{q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})\}_{\kappa=1}^k$[7] (Fig. 2d). In Section 5, we formalize how such a set of distributions can be learned and used to effectively quantify distance from $\mathcal{D}_{\text{train}}$.

[6]We consider functional Bregman divergences, i.e., the generalization of Bregman divergence that acts on functions.

[7]$\boldsymbol{\phi}$ will represent the joint set of parameters of all centroids q_κ.

5. Distance Aware Bottleneck

5.1. Model

In this section, we present the *Distance Aware Bottleneck (DAB)*: An IB problem with a complexity constraint that regularizes the network *and* renders the network distance-aware given a compressed representation of $\mathcal{D}_{\text{train}}$. We keep an information-geometric interpretation of this representation. In this case, the features of $\boldsymbol{x}$ and the codes used for computing distance from $\mathcal{D}_{\text{train}}$ lie in the parameter space of a distributional family $\mathcal{P}$[8] (Fig. 2b). As we will see in Section 5.3, the characterization of datapoints at a distributional granularity provides the model with deterministic uncertainty estimates. Moreover, we argue that an input $\boldsymbol{x}$ is better characterized by its encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$. This is because standard Euclidean distances might disregard aspects of data that are essential for characterizing distance from $\mathcal{D}_{\text{train}}$. In Section 6, we empirically confirm our hypothesis.

The mathematical construction of our work was alluded in Section 4 when we introduced random variable P_X. P_X is defined by X and takes as value the distribution $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$, i.e., the encoder, as we sample $X \sim p(\boldsymbol{x})$. In its empirical form over a finite number of N training datapoints $\mathcal{D}_{\text{train}}$, the distribution of P_X is a discrete *distribution over distributions*: P_X is discrete taking values in the set $\mathcal{P}_X = \{p(\boldsymbol{z} \mid \boldsymbol{x}_i; \boldsymbol{\theta})\}_{i=1}^N$ with probability $1/N$. We also define a random variable Q. By fixing the number k of distributional centroids, Q takes values $[q_1(\boldsymbol{z}; \boldsymbol{\phi}), q_2(\boldsymbol{z}; \boldsymbol{\phi}), \ldots, q_k(\boldsymbol{z}; \boldsymbol{\phi})]$ following distribution $\pi = [\pi(1), \pi(2), \ldots, \pi(k)]$. We will refer to its support set $\mathcal{Q} = \{q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})\}_{\kappa=1}^k$ as the codebook. $\pi_{\boldsymbol{x}}$ is the conditional assignment probabilities of encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$ to the

[8]For example, for the family $\mathcal{P} = \{p(\boldsymbol{z}; \boldsymbol{\theta}) \mid \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ of $d-$dimensional Normal distributions, the parameter space is $\boldsymbol{\Theta} = \{\mathbb{R}^d \times \text{Sym}^+(d; \mathbb{R})\}$ with $\text{Sym}^+(d; \mathbb{R})$ the set of $d \times d$ symmetric, positive definite matrices.

centroids such that $\pi_{\boldsymbol{x}} = [\pi_{\boldsymbol{x}}(1), \pi_{\boldsymbol{x}}(2), \ldots, \pi_{\boldsymbol{x}}(k)]$ with:

$$\pi_{\boldsymbol{x}}(\kappa) = p(Q = q_\kappa(\boldsymbol{z};\boldsymbol{\phi}) \mid P_X = p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})). \quad (7)$$

Compression of $\mathcal{D}_{\text{train}}$ is phrased as a RDFC problem (Eq. 5) for the source of encoders P_X using the source code Q:

$$R^k(\boldsymbol{\theta}) = \min_{\mathcal{Q},\pi_{\boldsymbol{x}}} \quad \mathcal{L}_{\text{RDFC}} \quad \text{subject to: } \mid \mathcal{Q} \mid = k, \text{ where:} \quad (8)$$

$$\mathcal{L}_{\text{RDFC}} \triangleq I(P_X, Q; \boldsymbol{\theta}, \boldsymbol{\phi}) + \alpha \mathbb{E}_{P_X,Q}\big[D(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q_\kappa(\boldsymbol{z};\boldsymbol{\phi}))\big]. \quad (9)$$

At this point, we underline that the source of encoders P_X depends on $\boldsymbol{\theta}$. Since centroids $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$ are used to quantize the set of encoders in $\mathcal{D}_{\text{train}}$, we will also call them code distributions. Albeit in this work we investigate only the behavior of the Kullback-Leibler divergence, the distortion function D in Eq. 9 can be *any* statistical distance measure between two *probability distributions*.

Optimizing with respect to the support set $\mathcal{Q}$ amounts to optimizing with respect to parameters $\boldsymbol{\phi}$ of codes $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$. Therefore, the problem in Eq. 8 can be written as:

$$R^k(\boldsymbol{\theta}) = \min_{\boldsymbol{\phi},\pi_{\boldsymbol{x}}} \quad \mathcal{L}_{\text{RDFC}}, \quad (10)$$

where $\mathcal{L}_{\text{RDFC}}$ is defined in Eq. 9. DAB replaces the rate term $I(Z, X; \boldsymbol{\theta})$ of the IB (Eq. 1) with the achievable rate $R^k(\boldsymbol{\theta})$ (Eq. 10). Formally, a DAB of cardinality k is defined as:

$$\min_{\boldsymbol{\theta}} \quad -I(Z, Y; \boldsymbol{\theta}) + \beta R^k(\boldsymbol{\theta}) \Longleftrightarrow \min_{\boldsymbol{\theta}} \min_{\boldsymbol{\phi},\pi_{\boldsymbol{x}}} \quad \mathcal{L}_{\text{DAB}},$$
$$\text{where: } \mathcal{L}_{\text{DAB}} \triangleq -I(Z, Y; \boldsymbol{\theta}) + \beta I(P_X, Q; \boldsymbol{\theta}, \boldsymbol{\phi}) + \alpha\beta\, \mathbb{E}_{P_X,Q}\big[D(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q_\kappa(\boldsymbol{z};\boldsymbol{\phi}))\big]. \quad (11)$$

Training the network with the loss function $\mathcal{L}_{\text{DAB}}$ encourages encoders $p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})$ whose samples $\boldsymbol{z}$ are informative about output $\boldsymbol{y}$ while staying statistically close to codes $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$. To get a better insight into Eq. 11, we consider two edge cases. In the case of a single code, i.e., $k = 1$, with D taken as the Kullback-Leibler divergence, Eq. 11 is equivalent to the empirical form (Eq. 3) of the VIB (Alemi et al., 2017) (Eq. 26) with regularization coefficient $\alpha \times \beta$. For $k = N$, the optimal codes would correspond to training datapoints' encoders: $q_\kappa(\boldsymbol{z};\boldsymbol{\phi}) = p(\boldsymbol{z} \mid \boldsymbol{x}_\kappa;\boldsymbol{\theta})$ yielding zero compression (and regularization).

We note that DAB's objective (Eq. 11) uses two separate terms for accuracy and for controlling the distance of training datapoints from the codebook. Such formulation enables DAB to choose between pulling correctly classified datapoints close to the codebook (being less uncertain) or pushing misclassified datapoints away (more uncertain), ultimately leading to better calibration (Section 6.3).

As in the standard VIB (Alemi et al., 2017), $I(Z, Y; \boldsymbol{\theta})$ can be estimated by the lower bound $\mathbb{E}_{X,Y,Z}[\log m(\boldsymbol{y} \mid \boldsymbol{z};\boldsymbol{\theta})]$ that is maximized with respect to variational decoder $m(\boldsymbol{y} \mid \boldsymbol{z};\boldsymbol{\theta})$ (Eq. 26). We emphasize that, in this work, the decoder does not utilize model's proposed distance for eventually *improving* its predictions. In DAB, this could be achieved by designing a *stochastic* decoder that induces variance proportionate to the estimated distance (uncertainty) in its final prediction. Such a decoder could be viewed as a distance-aware epinet (Osband et al., 2021) and its design is left as future work.

5.2. Learning Algorithm

The optimization problem of Eq. 11 can be solved by alternating minimizations (Banerjee et al., 2004). We note that $I(P_X, Q; \boldsymbol{\theta}, \boldsymbol{\phi})$ in Eq. 11 is tractable since P_X, Q are discrete random variables taking N (size of training dataset) and k (size of codebook) possible values, respectively. At each step, a single block of parameters is optimized. The most recent value is used for the parameters that are not optimized at the step. The internal minimization step corresponds to the computation of a RDFC (Eq. 5). The minimization steps are summarized as follows:

$$\text{repeat} \begin{cases} t. & \text{Update decoder } m(\boldsymbol{y} \mid \boldsymbol{z};\boldsymbol{\theta}), \text{encoder} \\ & p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}) : \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{DAB}} \\ t+1. & \text{Update } \pi_{\boldsymbol{x}} \text{ from Eq. 12} \\ t+2. & \text{Update centroids } q_\kappa(\boldsymbol{z};\boldsymbol{\phi}) : \\ & \boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \eta_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\text{DAB}} \\ t+3. & \text{Update } \pi \text{ from Eq. 13} \end{cases}$$

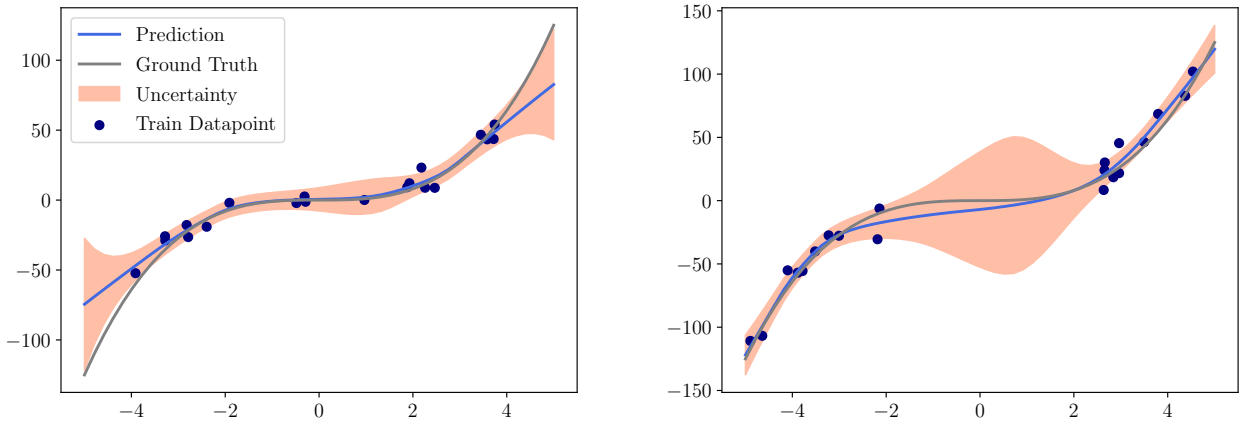
Steps $t+1$, $t+3$ are computationally cheap and can be performed analytically with a single forward pass:

$$\pi_{\boldsymbol{x}}(\kappa) = \frac{\pi(\kappa)}{\mathcal{Z}_{\boldsymbol{x}}(\alpha)} \exp\big(-\alpha D\left(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q_\kappa(\boldsymbol{z};\boldsymbol{\phi})\right)\big), \quad (12)$$

$$\pi(\kappa) = \frac{1}{N} \sum_{i=1}^{N} \pi_{\boldsymbol{x}_i}(\kappa), \quad (13)$$

where $\mathcal{Z}_{\boldsymbol{x}}(\alpha)$ is the partition function: $\mathcal{Z}_{\boldsymbol{x}}(\alpha) = \sum_{\kappa=1}^{k} \pi(\kappa) \exp\big(-\alpha D\big(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q_\kappa(\boldsymbol{z};\boldsymbol{\phi})\big)\big)$.

$\pi_{\boldsymbol{x}}$ in Eq. 12 (see also Eq. 10.124 of Cover (1999)) assigns higher probability to the centroid statistically closer in terms of D to the encoder of $\boldsymbol{x}$. π in Eq. 13 is derived in Lemma 10.8.1 of Cover (1999) and is the marginal of $\pi_{\boldsymbol{x}}$. Steps t, $t+2$ require back-propagation and correspond to gradient descent steps. The pseudocode of our method (Algorithm 1) along with a practical implementation for mini-batch training is given in Appendix B.



(a) A single cluster of training data points.

(b) Two clusters of training data points.

Figure 3: **Uncertainty estimation on noisy regression tasks.** We consider the Kullback-Leibler divergence as the distortion function in the uncertainty score of Eq. 14. A larger distance from the training datapoints (blue dots) is consistently quantified by higher uncertainty (width of pink area). Moreover, the true values lie well within $\pm 2\times$ the proposed uncertainty score around the predictive mean.

5.3. Uncertainty Quantification in the IB

The solution to the problem of Eq. 11 provides us with codes $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$ for encoders in $\mathcal{D}_{\text{train}}$ (Fig. 2d). Large distance from these codes signals an unfamiliar input $\boldsymbol{x}$ for which the network should be less confident when predicting $\boldsymbol{y}$ (Fig. 3). Formally, we define uncertainty over datapoint $\boldsymbol{x}$ as the conditional expected distortion (from last term in Eq. 11):

$$\begin{aligned}&\text{uncertainty}\left(\boldsymbol{x}\right) = \text{distance}\left(\boldsymbol{x}, \mathcal{D}_{\text{train}}\right)\\ &= \mathbb{E}_{Q|P_X=p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})}\left[D\big(p(\boldsymbol{z}\mid\boldsymbol{x};\boldsymbol{\theta}), q_\kappa(\boldsymbol{z};\boldsymbol{\phi})\big)\right]. \qquad (14)\end{aligned}$$

The distribution of Q in the expectation of Eq. 14 conditioned on encoder $p(\boldsymbol{z}\mid\boldsymbol{x};\boldsymbol{\theta})$ (also defined in Eq. 7) is given in Eq. 12. The expectation in Eq. 14 is taken over a finite number of values k. Under certain choices of D and distribution families, the uncertainty score of Eq. 14 can be computed *deterministically* with a *single forward pass* of the network without requiring Monte Carlo approximations. In this work, we consider the Kullback-Leibler divergence as the distortion function and multivariate Gaussians for the codes and the encoder.

5.4. Connections with Maximum Likelihood Mixture Estimation.

Limited work has sought connections between Maximum Likelihood Mixture Estimation (MLME) and computation of the rate-distortion function. Banerjee et al. (2004) prove the equivalence between these two problems for Bregman distortions and exponential families. In this case and under the assumption of constant variance for all mixture's components, learning the support set in RDFC corresponds to learning the mixture means. For MLME on parametric distributions, i.e., encoders, a straightforward way to leverage this connection is to define the "sample space" of the MLME as the "parameter space" of encoder's distribution family. Similarly, training with a mixture (for the marginal) VIB (Alemi et al., 2018) entails an MLME problem where the data points (to be clustered) are latent samples drawn from encoders. To get better insights, in Appendix C we anatomize the loss function. As we will see in Table 2, a full statistical description of encoders (instead of using a finite–single in the experiment– number of its samples) along with the proposed alternating minimization algorithm that guides assignments to centroids during training, helps DAB capture uncertainty *exactly* with a *single* forward pass. From a theoretical standpoint, deriving rigorous connections between the two problems would be interesting for future work.

6. Experiments

6.1. Synthetic Example

Before we compare with other DUMs, we first need to sanity-check the proposed model and learning algorithm. Synthetic experiments are handy for this task since they allow us to test the behavior of the model under different conditions. In this work, we apply DAB to synthetic regression tasks. In Fig. 3, we visualize the predictive uncertainty, i.e., the value of the distortion function in Eq. 14. We verify that as we move far away from the data, the model's confidence and accuracy decline. We consider two cases of training datasets. Fig. 3a follows the original set-up of Hernández-Lobato & Adams (2015). Fig. 3b is a harder variant of the first problem (Foong et al., 2019) and a typical failure case of many uncertainty-aware methods. Wilson & Izmailov (2020) show that many methods end up being overconfident in the area between the clusters of the training datapoints. We provide details for the dataset generation and the training setup in Appendix E.1.

Table 2: **OOD performance of baselines trained on the CIFAR-10 dataset.** We consider two OOD datasets for distinguishing from CIFAR-10 with varying levels of difficulty: SVHN (far OOD dataset) and CIFAR-100 (near OOD dataset). In bold are top results (within standard error). The horizontal line separates ensembles from DUMs. Only methods with the same background color can be directly compared with each other. The performance of all models is averaged over 10 random seeds. DAB outperforms all baselines in all tasks with respect to all metrics. DA stands for distance aware. R indicates whether model has been/ can be applied to regression tasks. PR indicates whether method can be applied to a pre-trained network.

Method	DA	R	PR	SVHN		CIFAR-100	
				AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
Deep Ensemble of 5 (Lakshminarayanan et al., 2017)	✗	✓	–	0.97 $\pm$ 0.004	0.984 $\pm$ 0.003	0.916 $\pm$ 0.001	0.902 $\pm$ 0.002
Deterministic (Zagoruyko & Komodakis, 2016)	✗	✓	–	0.956 $\pm$ 0.004	0.976 $\pm$ 0.004	0.892 $\pm$ 0.002	0.88 $\pm$ 0.002
DDU (Mukhoti et al., 2023)	✓	✗	✗	0.981 $\pm$ 0.002	0.966 $\pm$ 0.003	0.894 $\pm$ 0.001	0.901 $\pm$ 0.001
DUQ (Van Amersfoort et al., 2020)	✓	✗	✗	0.940 $\pm$ 0.003	0.956 $\pm$ 0.006	0.817 $\pm$ 0.012	0.826 $\pm$ 0.006
DUE (Van Amersfoort et al., 2021)	✓	✓	✗	0.958 $\pm$ 0.005	0.968 $\pm$ 0.015	0.871 $\pm$ 0.011	0.865 $\pm$ 0.011
SNGP (Liu et al., 2020; 2023)	✓	✓	✗	0.971 $\pm$ 0.003	0.987 $\pm$ 0.001	0.908 $\pm$ 0.003	0.907 $\pm$ 0.002
vanilla VIB (Alemi et al., 2018)	✓	✓	✓	0.715 $\pm$ 0.081	0.869 $\pm$ 0.039	0.663 $\pm$ 0.045	0.701 $\pm$ 0.034
DAB (ours)	✓	✓	✓	**0.986 $\pm$ 0.004**	**0.994 $\pm$ 0.002**	**0.922 $\pm$ 0.002**	**0.915 $\pm$ 0.002**

Table 3: **Accuracy and model size of OOD baselines**. Although we use a narrow bottleneck (8-dimensional latent variables), the accuracy of our model is not compromised compared to other deterministic uncertainty baselines. This is because 10 distributional codes can sufficiently represent the training dataset without diminishing the regularization effect and distance awareness of the rate-distortion constraint. More importantly, DAB can inject uncertainty awareness into the model with a minor model size overhead.

Method	Accuracy ↑	# Trainable Parameters ↓
Deep Ensemble of 5 (Lakshminarayanan et al., 2017)	**96.6%**	182, 395, 970
Deterministic (Zagoruyko & Komodakis, 2016)	96.2%	**36,479,194**
DDU (Mukhoti et al., 2023)	95.9%	**36,479,194**
DUQ (Van Amersfoort et al., 2020)	94.9%	40,568,784
DUE (Van Amersfoort et al., 2021)	95.6%	36,480,314
SNGP (Liu et al., 2020; 2023)	95.9%	36,483,024
vanilla VIB (Alemi et al., 2018)	95.9%	36,501,042
DAB (ours)	95.9%	36,501,114

6.2. DAB for Out-of-Distribution Detection

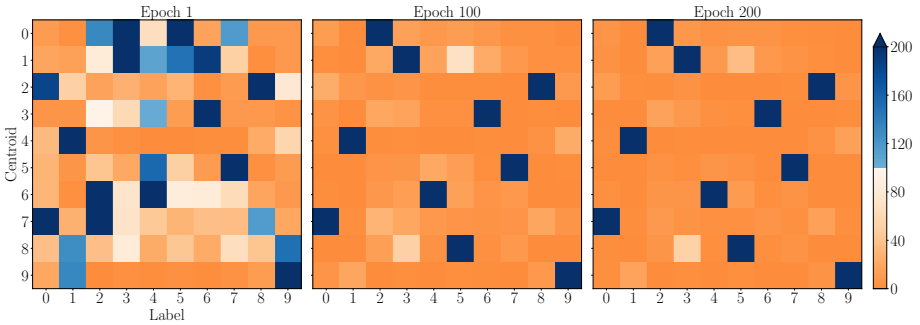
To compare the uncertainty quality of different models, we evaluate their performance in distinguishing between the test sets of CIFAR-10 and OOD datasets. We consider two OOD datasets of increasing difficulty: SVHN (Netzer et al., 2019) (far OOD/ easy task) and CIFAR-100 (near OOD/ difficult task). We compare DAB against a deterministic baseline, an ensemble baseline, a VIB with a mixture marginal trained with gradient descent, and the most competitive DUMs. All approaches do not require auxiliary OOD datasets either to train the models or to tune hyperparameters. In Table 2, we also outline some high-level properties of these models. For all methods, we use Wide ResNet 28-10 (Zagoruyko & Komodakis, 2016) as the base network. DAB and VIB are inserted right before output's dense layer. For both, we use 8-dimensional latent features. For DAB, we consider $k = 10$ centroids. For VIB, we consider a mixture with 10 components. We use the Kullback-Leibler divergence as the distortion function in Eq. 14. For fair comparisons, we train the IB and the Gaussian Process models with a single sample. Further training and evaluation configurations are given in Appendix E.2.

As shown in Table 2, DAB outperforms all baselines in terms of AUPRC and AUROC (the positive label in the binary OOD classification corresponds to the OOD images). We confirm that distances in distribution space are more informative compared to Euclidean distances. In Table 3, we report the accuracy and the size of the baselines. We note here that the accuracy of our model is on par with that of other DUMs. Importantly, DAB only minimally increases the single network's size while rendering it uncertainty-aware. The additional parameters correspond to centroids' parameters and DAB's dense layers implementing the head of the encoder.

In Figure 1, we visualize distances from the learned codebook. The in-distribution, test datapoints that are correctly classified lie within the statistical balls (Fig. 2c) defined by codebook's centroids and $\mathcal{D}_{\text{train}}$. The in-distribution, misclassified test datapoints are clearly separated from the training support but closer to the codebook than the near OOD. This, as we will see in the next section, qualitatively justifies DAB's strong calibration (Tables 4, 5). Lastly, near OOD datapoints are closer to the codebook than far OOD datapoints.

To qualitatively inspect the learning algorithm of Section 5.2, in Fig. 4 we plot the number of test datapoints per class repre-

Figure 4: **Qualitative evaluation of encoders' codebook.** We visualize the number of CIFAR-10 test data points per class assigned to each centroid during training. We assign a data point to the centroid with the smallest statistical distance from its encoder. Each centroid progressively attracts data points of the same class. Moreover, all centroids are assigned a non-zero number of test datapoints. Therefore, the centroids are useful for better explaining both train and previously unseen, test data points.



sented by each centroid $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$ in different training phases. We note that the class counts refer to the *true* and not the *predicted* class. As training proceeds, DAB learns similar latent features for datapoints that belong to the same class and pushes them closer to the same centroid. Certain centroids, however, conflate test datapoints of different classes. For example, a small number of test datapoints of class 3 (cat) are assigned to (are closest to) the centroid whose majority class is 5 (dog). Assignment to the wrong centroid presages model's misprediction for these datapoints. In contrast, we observed in an analogous figure using the training data points that they are completely separated by the centroids (the colormap displays only blue squares). However, this might not be the case in training datasets containing corrupted labels (Northcutt et al., 2021).

6.3. DAB for Misclassification Prediction

To further assess the quality of the proposed distance score (Eq. 14), we evaluate DAB's performance on misclassification prediction (Corbière et al., 2019; Zhu et al., 2022). Misclassification prediction is formulated as a binary classification task with the positive label indicating a classifier's mistake. We report the *Calibration AUROC* that was introduced by Kivlichan et al. (2021) and later used by Postels et al. (2022). As pointed out by Postels et al. (2022), the ECE (Expected Calibration Error) is not the appropriate metric for DUMs since their uncertainty scores are not directly reflected to the probabilistic forecast. Another benefit of Calibration AUROC compared to ECE is that it cannot be trivially reduced using post hoc calibration heuristics such as temperature scaling (Guo et al., 2017). In contrast, Calibration AUROC focuses on the intrinsic ability of the model to distinguish its correct from incorrect predictions and the ranking performance of its uncertainty score, i.e., whether high uncertainty predictions are wrong. In Table 4, we first evaluate DUMs' performance in predicting misclassified CIFAR-10 images. Here, we note that DAB bridges the gap between baselines and costly ensembles.

To illustrate scalability, we focus on the large-scale ImageNet dataset (Russakovsky et al., 2015) for the rest of this section. We observe that previous DUMs either exhibit training instability issues when scaled to larger datasets or fall behind in calibration (Postels et al., 2022). For this experiment, we use the ResNet-50 architecture. For DAB, we instantiate the backbone network with the publicly available, pre-trained weights (excluding the last dense layer of the classifier). The ResNet-50 features are passed through three fully connected dense layers that produce DAB's input. We consider two cases. First, we further fine-tune ResNet-50 alongside DAB. Next, we consider a setup similar to that of Alemi et al. (2017) where gradients are not backpropagated to the backbone network. This substantially decreases the training time and the number of trainable parameters. In both cases, we train DAB for 70 epochs. DAB uses a codebook with 1000 entries and 80-dimensional latent features. The implementation details are deferred to Appendix E.3.

We leverage DAB's distance awareness and consider a variant of the learning algorithm presented in Section 5.2. In particular, we modify the training objective in Eq. 11 to encourage high uncertainty for the misclassified datapoints in $\mathcal{D}_{\text{train}}$. This is achieved by adding a max-margin loss term (Eq. 35) in the objective at the gradient updates (Algorithm 1) to push the misclassified datapoints in $\mathcal{D}_{\text{train}}$ away from the codebook. The codebook is trained to represent only the correctly classified training examples.[9] We notice that:

penalizing high or small uncertainty (distance from the codebook) for the training examples according to the classification outcome improves model's calibration on the test examples.

[9] This variant has no effect for the CIFAR-10 experiments. This is because all models achieve very high training accuracy very early, eventually reaching 100%, leaving no misclassified examples for the repulsive loss term.

Table 4: **Calibration AUROC of DUMs for misclassification prediction on CIFAR-10.** We examine how well the model can predict it will be wrong from its estimated uncertainty. The problem is framed as a binary classification task with the positive label indicating a mistake. DAB closes the gap between DUMs and ensembles (Postels et al., 2022).

Method	Uncertainty Description	Calibration AUROC ↑
Deep Ensemble of 5 (Lakshminarayanan et al., 2017)	Gibbs softmax entropy	$\mathbf{0.951 \pm 0.001}$
DDU (Mukhoti et al., 2023)	Softmax entropy	0.632 ± 0.009
DUQ (Van Amersfoort et al., 2020)	Euclidean distance (l_2-norm) from centroid	0.889 ± 0.013
DUE (Van Amersfoort et al., 2021)	Posterior variance	0.856 ± 0.026
SNGP (Liu et al., 2020; 2023)	Dempster-Shafer uncertainty	0.897 ± 0.006
DAB (ours)	**Statistical distance (KL) from centroid**	$\mathbf{0.930 \pm 0.003}$

Table 5: **DAB's performance on ImageNet-1K.** DAB outperforms ensembles at predicting misclassifications. Moreover, it can better distinguish ImageNet-O from ImageNet images. More importantly, it does so with significantly fewer trainable parameters. The performance of all models is averaged over 4 random seeds.

Method	Uncertainty Description	Calibration AUROC ↑	OOD AUROC ↑ ImageNet-O	Accuracy ↑	# Trainable Parameters
Deep Ensemble of 5 (Lakshminarayanan et al., 2017)	Gibbs softmax entropy	0.861 ± 0.0004	0.642 ± 0.001	$\mathbf{78.4 \pm 0.06}\%$	$117,672,960$
DAB with fine-tuned ResNet-50 (ours)	Statistical distance (KL)	$\mathbf{0.868 \pm 0.0008}$	$\mathbf{0.743 \pm 0.004}$	$76.1 \pm 0.02\%$	$36,612,328$
DAB with pre-trained ResNet-50 (ours)	Statistical distance (KL)	0.866 ± 0.0003	0.732 ± 0.004	$74.71 \pm 0.09\%$	$\mathbf{13,077,736}$

For completeness, we also examine DAB's performance on the ImageNet vs ImageNet-O (Hendrycks et al., 2021) OOD task. For the OOD experiments, we quantize all training datapoints regardless the classification outcome. We report only AUROC which is preferred in situations of highly imbalanced OOD tasks (Pinto et al., 2022a) – ImageNet-O has only $2,000$ OOD images.

In Table 5, we report performance against ensembles which is the gold standard in calibration and OOD detection. As we see, DAB has better calibration and OOD detection than ensembles in both cases. We remark that applying DAB without ResNet-50 fine-tuning does not substantially hurt its calibration or OOD capability. The small performance gap is attributed to the fact that the largest part of the encoder is not regularized to stay close to the codebook. Finally, we see that DAB nearly reaches the initial accuracy of 74.9 achieved by the pre-trained ResNet-50 [10] like the standard VIB (Alemi et al., 2017).

Additional Experiments. Due to space constraints, we supplement the experiments in Appendix. We ablate DAB's hyperparameters in Appendix D.1. Appendix D.2 evaluates DAB on corrupted CIFAR-10. Appendix D.3 provides further qualitative evaluations of the learned codebook. In Appendix D.4, we test DAB on OOD regression problems.

7. Limitations & Future Research

The main purpose of this work is to define and analyze a more comprehensive notion of distance from the training data manifold under the auspices of information bottleneck methods. Although in the experiments we used the Kullback-Leibler divergence, the proposed framework is flexible and supports inference with alternative statistical distances (Minka, 2005; Nielsen, 2023). Evaluating the impact of diverse distance metrics on model's performance is a compelling avenue for future work.

DAB, like other DUMs, currently falls behind ensembles in terms of accuracy. As we briefly discussed in Section 5.1, it remains to be seen whether this can be fixed by redesigning DAB's decoder to make use of its distance score. In this article, DAB was demonstrated primarily on image classification tasks. Applying DAB in different settings such as natural language generation (Xiao et al., 2022) is another important application area. In this work, we did not use additional OOD datasets during training. DAB's Outlier Exposure (OE) (Hendrycks et al., 2019) by repelling OE datapoints away from the codebook could further improve OOD capability. Moreover, leveraging the majority vote among data points within each centroid (Fig. 4) can enhance the model's ability to make accurate predictions, even when faced with labels containing errors (Platanios et al., 2020). Finally, analyzing DAB in concert with data augmentation methods for enhancing the codebook for image datasets is another interesting line of future research. We intend this paper to offer a fresh perspective on uncertainty estimation and we believe its empirical findings are an important step toward future directions mentioned above.

8. Conclusion

We introduced DAB, a distance-aware framework for deep neural networks (DNNs). We framed distance awareness as a rate-distortion problem to learn a lossy compression of the training dataset via a codebook of encoders. Experimental analysis shows that DNNs equipped with distances from this codebook outperform expensive baselines at OOD tasks and are better calibrated.

[10] https://keras.io/api/applications/

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

This work was supported in part by DARPA award FA8750-17-2-0130, NSF grant 2038612, and by the U.S. Army Research Office and the U.S. Army Futures Command under contract W519TC-23-F-0045. The authors would like to thank Christos Faloutsos for insightful discussions on clustering methods that helped inform this work and Barnabás Póczos for his astute feedback on the manuscript draft.

Code Availability

Publicly available code for reproducing the experiments can be found at:

https://github.com/ifiaposto/Distance_Aware_Bottleneck

References

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. Deep variational information bottleneck. In *International Conference on Learning Representations*, 2017.

Alemi, A. A., Fischer, I., and Dillon, J. V. Uncertainty in the variational information bottleneck. *arXiv preprint arXiv:1807.00906*, 2018.

Azoury, K. S. and Warmuth, M. K. Relative loss bounds for online density estimation with the exponential family of distributions. *Machine Learning*, 2001.

Banerjee, A., Dhillon, I., Ghosh, J., and Merugu, S. An information theoretic analysis of maximum likelihood mixture estimation for exponential families. In *International Conference on Machine Learning*, 2004.

Banerjee, A., Merugu, S., Dhillon, I. S., Ghosh, J., and Lafferty, J. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 2005.

Barndorff-Nielsen, O. *Information and exponential families: in statistical theory*. John Wiley & Sons, 2014.

Berger, T. Rate distortion theory; a mathematical basis for data compression. *Prentice-Hall*, 1971.

Berger, T. and Gibson, J. D. Lossy source coding. *IEEE Transactions on Information Theory*, 1998.

Blahut, R. Computation of channel capacity and rate-distortion functions. *IEEE Transactions on Information Theory*, 1972.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International Conference on Machine Learning*, 2015.

Bregman, L. M. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 1967.

Brekelmans, R. and Nielsen, F. Rho-tau bregman information and the geometry of annealing paths. *arXiv preprint arXiv:2209.07481*, 2022.

Charpentier, B., Zhang, C., and Günnemann, S. Training, architecture, and prior for deterministic uncertainty methods. In *ICLR 2023 Workshop on Pitfalls of limited data and computation for Trustworthy ML*, 2023. URL https://openreview.net/forum?id=iYA80086YH.

Corbière, C., Thome, N., Bar-Hen, A., Cord, M., and Pérez, P. Addressing failure prediction by learning model confidence. *Advances in Neural Information Processing Systems*, 32, 2019.

Cover, T. M. *Elements of information theory*. John Wiley & Sons, 1999.

Csiszár, I. Information geometry and alternating minimization procedures. *Statistics and Decisions*, 1984.

Csiszár, I. Generalized projections for non-negative functions. In *IEEE International Symposium on Information Theory*, 1995.

Davis, J. and Dhillon, I. Differential entropic clustering of multivariate Gaussians. *Advances in Neural Information Processing Systems*, 2006.

Foong, A. Y., Li, Y., Hernández-Lobato, J. M., and Turner, R. E. 'in-between'uncertainty in bayesian neural networks. *arXiv preprint arXiv:1906.11537*, 2019.

Frigyik, B. A., Srivastava, S., and Gupta, M. R. Functional Bregman divergence and Bayesian estimation of distributions. *IEEE Transactions on Information Theory*, 2008.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2018.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2016.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Hendrycks, D., Mazeika, M., and Dietterich, T. Deep anomaly detection with outlier exposure. *Proceedings of the International Conference on Learning Representations*, 2019.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. AugMix: A simple data processing method to improve robustness and uncertainty. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *CVPR*, 2021.

Hernández-Lobato, J. M. and Adams, R. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, 2015.

Hoffman, M. D., Riquelme, C., and Johnson, M. J. The β-VAE's implicit prior. In *Workshop on Bayesian Deep Learning, Advances in Neural Information Processing Systems*, 2017.

Jaeger, P. F., Lüth, C. T., Klein, L., and Bungert, T. J. A call to reflect on evaluation practices for failure detection in image classification. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=YnkGMIh0gvX.

Jiang, K., Kulis, B., and Jordan, M. Small-variance asymptotics for exponential family Dirichlet process mixture models. *Advances in Neural Information Processing Systems*, 2012.

Kivlichan, I., Liu, J., Vasserman, L. H., and Lin, Z. Measuring and improving model-moderator collaboration using uncertainty estimation. In *Workshop on Online Abuse and Harms*, 2021.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 2017.

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. In *International Conference on Machine Learning*, 2021.

Liu, J., Lin, Z., Padhy, S., Tran, D., Bedrax Weiss, T., and Lakshminarayanan, B. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 2020.

Liu, J. Z., Padhy, S., Ren, J., Lin, Z., Wen, Y., Jerfel, G., Nado, Z., Snoek, J., Tran, D., and Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. *Journal of Machine Learning Research*, 24(42):1–63, 2023.

Markelle Kelly, Rachel Longjohn, K. N. Uci repository of machine learning databases. In *http://www.ics.uci.edu/ mlearn/MLRepository.html*, 1998.

Matz, G. and Duhamel, P. Information geometric formulation and interpretation of accelerated Blahut-Arimoto-type algorithms. In *Information theory workshop*. IEEE, 2004.

Minka, T. Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research, 2005.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Mukhoti, J., Kirsch, A., van Amersfoort, J., Torr, P. H., and Gal, Y. Deep deterministic uncertainty: A new simple baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24384–24394, June 2023.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. The Street View House Numbers (SVHN) dataset. *http://ufldl.stanford.edu/housenumbers*, 2019.

Nielsen, F. Fisher-Rao and pullback Hilbert cone distances on the multivariate Gaussian manifold with applications to simplification and quantization of mixtures. In *Annual Workshop on Topology, Algebra, and Geometry in Machine Learning*, 2023.

Nielsen, F., Boissonnat, J.-D., and Nock, R. Bregman Voronoi diagrams: Properties, algorithms and applications. *Extended abstract appeared in ACM-SIAM Symposium on Discrete Algorithms 2007. INRIA Technical Report RR-6154, arXiv preprint arXiv:0709.2196*, 2007.

Northcutt, C., Jiang, L., and Chuang, I. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.

Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., and Yokota, R. Practical deep learning with Bayesian principles. *Advances in Neural Information Processing Systems*, 2019.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. Deep exploration via bootstrapped DQN. *Advances in Neural Information Processing Systems*, 2016.

Osband, I., Wen, Z., Asghari, S. M., Dwaracherla, V., Ibrahimi, M., Lu, X., and Van Roy, B. Epistemic neural networks. *arXiv preprint arXiv:2107.08924*, 2021.

Pinto, F., Torr, P. H., and K. Dokania, P. An impartial take to the cnn vs transformer robustness contest. In *European Conference on Computer Vision*, pp. 466–480. Springer, 2022a.

Pinto, F., Yang, H., Lim, S. N., Torr, P., and Dokania, P. Using mixup as a regularizer can surprisingly improve accuracy & out-of-distribution robustness. *Advances in Neural Information Processing Systems*, 35:14608–14622, 2022b.

Platanios, E. A., Al-Shedivat, M., Xing, E., and Mitchell, T. Learning from imperfect annotations. *arXiv preprint arXiv:2004.03473*, 2020.

Postels, J., Segù, M., Sun, T., Sieber, L. D., Van Gool, L., Yu, F., and Tombari, F. On the practicality of deterministic epistemic uncertainty. In *International Conference on Machine Learning*, 2022.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems*, 2007.

Rose, K. A mapping approach to rate-distortion computation and analysis. *IEEE Transactions on Information Theory*, 1994.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115: 211–252, 2015.

Tishby, N., Pereira, F. C., and Bialek, W. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, 2009.

Van Amersfoort, J., Smith, L., Teh, Y. W., and Gal, Y. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, 2020.

Van Amersfoort, J., Smith, L., Jesson, A., Key, O., and Gal, Y. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

Wilson, A. G. and Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in Neural Information Processing Systems*, 2020.

Wu, Y., Zhai, S., Srivastava, N., Susskind, J. M., Zhang, J., Salakhutdinov, R., and Goh, H. Uncertainty weighted actor-critic for offline reinforcement learning. In *International Conference on Machine Learning*, 2021.

Xiao, Y., Liang, P. P., Bhatt, U., Neiswanger, W., Salakhutdinov, R., and Morency, L.-P. Uncertainty quantification with pre-trained language models: A large-scale empirical analysis. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *British Machine Vision Conference*, 2016.

Zhu, F., Cheng, Z., Zhang, X.-Y., and Liu, C.-L. Rethinking confidence calibration for failure prediction. In *European Conference on Computer Vision*, pp. 518–536. Springer, 2022.

A. Preliminaries

A.1. Definitions

Definition A.1 (***Bregman Divergence***)**.** Let $f : \mathcal{S} \to \mathbb{R}$ be a differentiable, strictly convex function of Legendre type on a convex set $\mathcal{S} \subseteq \mathbb{R}^d$. The *Bregman divergence* $D_f : \mathcal{S} \times \mathcal{S} \to [0, \infty)$ for any two points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}$ is defined as (Bregman, 1967):

$$D_f(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) - f(\boldsymbol{y}) - \langle \boldsymbol{x} - \boldsymbol{y}, \nabla f(\boldsymbol{y}) \rangle, \tag{15}$$

where $\nabla f(\boldsymbol{y}) \in \mathbb{R}^d$ denotes the gradient vector of f evaluated at $\boldsymbol{y}$.

Definition A.2 (***Dual Bregman Form of Exponential Family***)**.** Each probability density function for $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ in the exponential family $\mathcal{F}_\psi = \{p_\psi(\cdot; \boldsymbol{\phi}) \mid \boldsymbol{\phi} \in \Phi\}$, where $\Phi = \mathrm{dom}(\psi) \subseteq \mathbb{R}^p$, has the form:

$$p_\psi(\boldsymbol{x}; \boldsymbol{\phi}) = \exp(\langle \boldsymbol{t}(\boldsymbol{x}), \boldsymbol{\phi} \rangle - \psi(\boldsymbol{\phi}))\, h_0(\boldsymbol{x}). \tag{16}$$

$\boldsymbol{t}(\boldsymbol{x})$ is the natural statistic of the family. $\boldsymbol{\phi}$ is called the natural parameter and Φ the natural parameter space. $\psi(\boldsymbol{\phi})$ is the log-partition function of the family that normalizes the density function. $h_0(\boldsymbol{x})$ is a non-negative function that does not depend on $\boldsymbol{\phi}$.

If $\boldsymbol{t}(\boldsymbol{x})$ is minimal, i.e., $\nexists$ non-zero $\boldsymbol{\alpha} \in \mathbb{R}^p$ such that $\langle \boldsymbol{\alpha}, \boldsymbol{t}(\boldsymbol{x}) \rangle = c$ (a constant) $\forall \boldsymbol{x} \in \mathcal{X}$, and Φ is open, i.e., $\Phi = \mathrm{int}(\Phi)$, then $\mathcal{F}_\psi$ is called *regular exponential family*. In this case, it can be shown (Barndorff-Nielsen, 2014) that Φ is a non-empty convex set in $\mathbb{R}^d$ and that ψ is a convex function. From Theorem 4 by Banerjee et al. (2005), the density of Eq. 16 can be written as:

$$p_\psi(\boldsymbol{x}; \boldsymbol{\phi}) = \exp\left(-D_{\psi^*}\left(\boldsymbol{t}(\boldsymbol{x}), \hat{\boldsymbol{t}}(\boldsymbol{\phi})\right)\right) f_{\psi^*}(\boldsymbol{x}), \tag{17}$$

where ψ^* is the Legendre-conjugate of ψ and D_{ψ^*} the corresponding Bregman divergence (def. A.1). $\hat{\boldsymbol{t}}(\boldsymbol{\phi})$ is the expectation of the sufficient statistic:

$$\hat{\boldsymbol{t}}(\boldsymbol{\phi}) \triangleq \mathbb{E}_X[\boldsymbol{t}(\boldsymbol{x})]. \tag{18}$$

By differentiating $\int p_\psi(\boldsymbol{x}; \boldsymbol{\phi}) d\boldsymbol{x} = 1$ with respect to $\boldsymbol{\phi}$ and by making use of Eq. 16 and Eq. 18, it can be proved that:

$$\hat{\boldsymbol{t}}(\boldsymbol{\phi}) = \nabla \psi(\boldsymbol{\phi}). \tag{19}$$

Finally, $f_{\psi^*}(\boldsymbol{x})$ is a non-negative function that does not depend on $\boldsymbol{\phi}$:

$$f_{\psi^*}(\boldsymbol{x}) = \exp\left(\psi^*(\boldsymbol{t}(\boldsymbol{x}))\right) h_0(\boldsymbol{x}). \tag{20}$$

Therefore, when we train by Maximum Likelihood Estimation (MLE) to learn $\boldsymbol{\phi}$, this term can be omitted from the objective function. Eq. 17 is called the *Bregman form* of the exponential family (Eq. 16) and provides a convenient way to parametrize the exponential family distribution with its expectation parameter (Eq. 18).

Definition A.3 (***Scaled Exponential Family***)**.** Given an exponential family $\mathcal{F}_\psi$ with natural parameter $\boldsymbol{\phi}$ and log-partition function $\psi(\boldsymbol{\phi})$ (Eq. 16), a *scaled exponential family* (Jiang et al., 2012) $\mathcal{F}_\psi^\alpha$ with $\alpha > 0$ has natural parameter $\tilde{\boldsymbol{\phi}} = \alpha \boldsymbol{\phi}$ and log-partition function $\tilde{\psi}(\tilde{\boldsymbol{\phi}}) = \alpha \psi(\tilde{\boldsymbol{\phi}}/\alpha) = \alpha \psi(\boldsymbol{\phi})$. In case $\mathcal{F}_\psi$ is a regular exponential family, the Bregman form of the scaled family is (Jiang et al., 2012):

$$p_{\tilde{\psi}}(\boldsymbol{x}; \tilde{\boldsymbol{\phi}}) = \exp\left(-\alpha D_{\psi^*}\left(\boldsymbol{t}(\boldsymbol{x}), \hat{\boldsymbol{t}}(\boldsymbol{\phi})\right)\right) f_{\alpha\psi^*}(\boldsymbol{x}), \tag{21}$$

where ψ^* is the Legendre-conjugate of ψ. $f_{\alpha\psi^*}$ is defined in Eq. 20 where we scale ψ^* by α. Finally, the mean $\hat{\boldsymbol{t}}(\boldsymbol{\phi})$ of $\mathcal{F}_\psi^\alpha$ is the same with that of $\mathcal{F}_\psi$ and is given in Eq. 18, Eq. 19.

A.2. Variational Information Bottleneck

Alemi et al. (2017) derive efficient variational estimates of the mutual information terms in Eq. 1. The accuracy term is:

$$I(Z, Y; \boldsymbol{\theta}) = \int \log \frac{p(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})}{p(\boldsymbol{y})} p(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\theta}) d\boldsymbol{z} d\boldsymbol{y}.^{11} \tag{22}$$

The decoder $p(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})$ in Eq. 22 is fully defined:

$$p(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta}) = \int \frac{p(\boldsymbol{y}|\boldsymbol{x}) p(\boldsymbol{z}|\boldsymbol{x}; \boldsymbol{\theta}) p(\boldsymbol{x})}{p(\boldsymbol{z}; \boldsymbol{\theta})} d\boldsymbol{x}. \tag{23}$$

Generally, Eq. 23 cannot be computed in closed-form. Moreover, it contains the intractable marginal $p(\boldsymbol{z}; \boldsymbol{\theta})$:

$$p(\boldsymbol{z}; \boldsymbol{\theta}) = \int p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}) p(\boldsymbol{x}) d\boldsymbol{x}. \tag{24}$$

Similarly, the regularization term is analytically intractable since:

$$I(Z, X; \boldsymbol{\theta}) = \int \log \frac{p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})}{p(\boldsymbol{z}; \boldsymbol{\theta})} p(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta}) d\boldsymbol{z} d\boldsymbol{x}. \tag{25}$$

Variational estimates in a distributional family $m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})$ [12] and $q(\boldsymbol{z}; \boldsymbol{\phi})$ of Eq. 23, Eq. 24 minimize the Kullback-Leibler divergences $D_{\mathrm{KL}}(p(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta}), m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta}))$ and $D_{\mathrm{KL}}(p(\boldsymbol{z}; \boldsymbol{\theta}), q(\boldsymbol{z}; \boldsymbol{\phi}))$, respectively. Non-negativity of the Kullback-Leibler divergence yields a lower bound of Eq. 22 and an upper bound of Eq. 25. Substituting these variational bounds in Eq. 1 gives us the Variational Information Bottleneck (VIB) minimization loss:

$$\mathcal{L}_{\mathrm{VIB}} = \mathbb{E}_{X,Y,Z}[-\log m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})] + \beta \mathbb{E}_X[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}), q(\boldsymbol{z}; \boldsymbol{\phi}))]. \tag{26}$$

B. Learning Algorithm (Section 5.2 continued.)

DAB's concrete learning algorithm is given in Algorithm 1. Each epoch (outer loop in Algorithm 1) consists of the four alternating minimization steps presented in Section 5.2.

To render the update of π (Eq. 13) amenable to mini-batch optimization, we maintain i) a non-trainable tensor that holds the current π ii) a moving average of the mini-batch marginals (Eq. 13). The moving average is updated at step 4 in Algorithm 1 such that at batch t of size B:

$$\pi^0(\kappa) = 1/k, \ \pi^t(\kappa) = \gamma \pi^{t-1}(\kappa) + (1-\gamma) \frac{1}{B} \sum_{i=1}^{B} \pi_{\boldsymbol{x}_i}(\kappa). \tag{27}$$

$0 \leq \gamma \leq 1$ is the momentum of the moving average. At the onset of step 4, the moving average is reset to the uniform distribution. At the end of step 4, π is set to its moving average and is kept fixed throughout the rest of the steps, i.e., all training datapoints use the same π.

We maintain two optimizers for the gradient updates $\nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathrm{DAB}}$ and $\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{DAB}}$. The gradient descent updates in Algorithm 1 are written using constant learning rates $\eta_{\boldsymbol{\theta}}$, $\eta_{\boldsymbol{\phi}}$. In practice, we can use any optimizer with adaptive learning rates. To make sure that the gradients are not propagated through $\pi_{\boldsymbol{x}}$ (Eq. 12), we apply a `tf.stop_gradient` operator when $\mathcal{L}_{\mathrm{DAB}}$ is computed.

In this work, we use multivariate Gaussian distributions for centroids and encoders. In this case, the centroids' parameters $\boldsymbol{\phi}$ correspond to the means and covariance matrices: $\boldsymbol{\phi} = \{\boldsymbol{\mu}_\kappa, \boldsymbol{\Sigma}_\kappa\}_{\kappa=1}^k$ and the optimal solution has a closed form (Davis & Dhillon, 2006). We empirically observed that using the closed-form update for the covariance matrix and gradient descent for the means facilitates optimization and speeds up convergence. To make use of the closed-form solution for the covariance matrix, we maintain non-trainable tensors holding current $\boldsymbol{\Sigma}_\kappa$ along with their moving averages. At the beginning of the training, the centroids's covariances are initialized to the identity matrix. The moving averages are updated in a way similar to that of π (Eq. 27). On the onset of step 3 in Algorithm 1, the moving average is reset to the zero matrix and is updated during the gradient updates $\nabla_{\boldsymbol{\phi}} \mathcal{L}_{\mathrm{DAB}}$. At the end of step 3, the codebook covariances are set to their moving averages computed during this step.

[11] Note that in our problem, $p(\boldsymbol{y}), p(\boldsymbol{x}), p(\boldsymbol{y} \mid \boldsymbol{x})$ refer to our training dataset. Therefore, they are independent of $\boldsymbol{\theta}$.

[12] As in the main paper, we use $\boldsymbol{\theta}$ to denote the joint set of parameters of both encoder and variational decoder.

Algorithm 1: Optimization of Distance Aware Bottleneck

Inputs:
training data: $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$
codebook size: k
statistical distance: D
hyper-parameters:
regularization coefficient $\beta \geq 0$ (Eq. 11)
temperature $\alpha \geq 0$ (Eq. 12)

Outputs:
optimal parameters of encoder and decoder: $\boldsymbol{\theta}^*$
optimal codebook parameters: $\boldsymbol{\phi}^*$
marginal assignment probabilities: $\pi*$

Initialize:
encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$
decoder $m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})$
codebook $\{q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})\}_{\kappa=1}^k$
π, $\pi_{\boldsymbol{x}_i}$ to uniform distribution

while *not converged* **do**
  step 1:
  Update decoder $m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})$, encoder $p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta})$: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{DAB}}$ ($\mathcal{L}_{\text{DAB}}$ in Eq. 11)
  step 2:
  for $i = 1, 2, \ldots, N$ **do**
    for $\kappa = 1, 2, \ldots, k$ **do**
      $\pi_{\boldsymbol{x}_i}(\kappa) = \frac{\pi(\kappa)}{\mathcal{Z}_{\boldsymbol{x}_i}(\alpha)} \exp\left(-\alpha D(p(\boldsymbol{z} \mid \boldsymbol{x}; \boldsymbol{\theta}), q_\kappa(\boldsymbol{z}; \boldsymbol{\phi}))\right)$ (Eq. 12)
    end
  end
  step 3:
  Update codes $q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})$: $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \eta_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\text{DAB}}$ ($\mathcal{L}_{\text{DAB}}$ in Eq. 11)
  step 4:
  for $\kappa = 1, 2, \ldots, k$ **do**
    $\pi(\kappa) = \frac{1}{N} \sum_{i=1}^N \pi_{\boldsymbol{x}_i}(\kappa)$ (Eq. 13)
  end
end

C. VIB for Euclidean Clustering of Latent Codes

One way we can use the set of distributions $\{q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})\}_{\kappa=1}^k$ is to consider a mixture of k distributions for the marginal $q(\boldsymbol{z}; \boldsymbol{\phi})$ and trivially train it by gradient descent (Alemi et al., 2018). To better understand the role of each $q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})$ during optimization, we associate a discrete random variable $\hat{Z}$ with Z. The value of $\hat{Z}$ indicates the assignment of Z to a component $q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})$ of the mixture. We rewrite the upper bound of Eq. 2 in terms of $\hat{Z}$. The resulting decomposition of Proposition 1 shows that the regularization term in the VIB (Eq. 28) encloses the objective of a fixed-cardinality rate-distortion function (Eq. 5) under some assumptions. However, computation of Eq. 28 requires Monte-Carlo samples of Z to assign an encoder to the mixture components. The regularization terms of VIB and DAB are identical for $k = 1$. The rate-distortion formulation of Eq. 28 motivates the DAB objective (Section 5.1). It also serves as a conceptual step towards the definition of a rate-distortion function acting directly on probability densities.

Proposition 1. *Let the variational marginal $q(\boldsymbol{z}; \boldsymbol{\phi})$ of Eq. 2 be a mixture of k distributions in $\mathbb{R}^d$ that belong to the scaled regular exponential family (def. A.3) $\mathcal{F}_\psi^\alpha$ with $\alpha > 0$ and log-partition function ψ. Let $\hat{\mathbf{t}}_\kappa$ be the expected value of the minimal sufficient statistic $\mathbf{t}(Z)$ of the family when $Z \sim q_\kappa(\boldsymbol{z}; \boldsymbol{\phi})$. Let $\hat{Z}$ be a (latent) categorical random variable following distribution $q(\hat{\boldsymbol{z}})$. We assume $\hat{Z}$ is conditionally independent of X given Z, i.e., $P(X, Y, Z) = P(X)P(Z \mid X)P(\hat{Z} \mid Z)$.*

The upper bound of the VIB in Eq. 2 can be decomposed as:

$$\mathbb{E}_X\left[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q(\boldsymbol{z};\boldsymbol{\phi}))\right] = \\ -H(Z \mid X;\boldsymbol{\theta}) - \mathbb{E}_{X,Z}[\log f_{\psi^*}(\boldsymbol{z})] + \alpha\mathbb{E}_{X,Z,\hat{Z}}[D_{\psi^*}(\boldsymbol{t}(\boldsymbol{z}),\hat{\boldsymbol{t}}_{\hat{z}}(\boldsymbol{\phi}))] + \mathbb{E}_{X,Z}[D_{\mathrm{KL}}(q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi}), q(\hat{\boldsymbol{z}}))], \tag{28}$$

where D_{ψ^*} is the Bregman divergence of $\mathcal{F}_\psi$, i.e., the Bregman divergence defined by the Legendre-conjugate function ψ^* of ψ. f_{ψ^*} is a non-negative function that does not depend on the natural parameter $\boldsymbol{\phi}$.

Proof. We expand the upper bound in Eq. 2:

$$\mathbb{E}_X\left[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q(\boldsymbol{z};\boldsymbol{\phi}))\right] = \\ \int p(\boldsymbol{x})p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})\log p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{z}d\boldsymbol{x} - \int p(\boldsymbol{x})p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})\log q(\boldsymbol{z};\boldsymbol{\phi})d\boldsymbol{z}d\boldsymbol{x}. \tag{29}$$

The first term of Eq. 29 is the negative conditional differential entropy of the encoder, i.e., $-H(Z \mid X;\boldsymbol{\theta})$. We will focus on the second term of Eq. 29. For a fixed $\boldsymbol{z}$:

$$\begin{aligned}\log q(\boldsymbol{z};\boldsymbol{\phi}) &= \mathbb{E}_{\hat{Z}|\boldsymbol{z}}[\log q(\boldsymbol{z};\boldsymbol{\phi})] \\ &= \mathbb{E}_{\hat{Z}|\boldsymbol{z}}[\log q(\boldsymbol{z};\boldsymbol{\phi}) + \log q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi}) - \log q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi})] \\ &= \mathbb{E}_{\hat{Z}|\boldsymbol{z}}[\log q(\boldsymbol{z},\hat{\boldsymbol{z}};\boldsymbol{\phi}) - \log q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi})] \\ &= \mathbb{E}_{\hat{Z}|\boldsymbol{z}}[\log q(\boldsymbol{z} \mid \hat{\boldsymbol{z}};\boldsymbol{\phi}) + \log q(\hat{\boldsymbol{z}}) - \log q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi})].\end{aligned} \tag{30}$$

We first analyze the first term in Eq. 30. By definition of $\hat{Z}$, $q(\boldsymbol{z} \mid \hat{\boldsymbol{z}};\boldsymbol{\phi}) = q_{\hat{\boldsymbol{z}}}(\boldsymbol{z};\boldsymbol{\phi})$. Let $\hat{\boldsymbol{t}}_\kappa(\boldsymbol{\phi})$ be the expected value of $\boldsymbol{t}(Z)$ when Z is sampled from the κ–th component of the mixture: $Z \sim q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$. Since $q_\kappa(\boldsymbol{z};\boldsymbol{\phi})$ belongs to the regular exponential family, its Bregman form (Eq. 17) is:

$$q_\kappa(\boldsymbol{z};\boldsymbol{\phi}) = \exp\left(-D_{\psi^*}(\boldsymbol{t}(\boldsymbol{z}),\hat{\boldsymbol{t}}_\kappa(\boldsymbol{\phi}))\right) f_{\psi^*}(\boldsymbol{z}), \tag{31}$$

where ψ^* is the conjugate of the log-partition function ψ of the family, D_{ψ^*} is the Bregman divergence defined by ψ^*, and f_{ψ^*} given in Eq. 20. In general, we can consider a scaled exponential family with Bregman form (see def. A.3):

$$q_\kappa(\boldsymbol{z};\boldsymbol{\phi}) = \exp\left(-\alpha D_{\psi^*}(\boldsymbol{t}(\boldsymbol{z}),\hat{\boldsymbol{t}}_\kappa(\boldsymbol{\phi}))\right) f_{\alpha\psi^*}(\boldsymbol{z}),\ \alpha > 0. \tag{32}$$

We now look at the last two terms of Eq. 30:

$$\mathbb{E}_{\hat{Z}|\boldsymbol{z}}[\log q(\hat{\boldsymbol{z}}) - \log q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi})] = -D_{\mathrm{KL}}(q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi}), q(\hat{\boldsymbol{z}})). \tag{33}$$

By taking expectation of Eq. 30 with respect to $p(\boldsymbol{x}), p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})$ and using Eq. 32 and Eq. 33, we can rewrite Eq. 29:

$$\mathbb{E}_X\left[D_{\mathrm{KL}}(p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta}), q(\boldsymbol{z};\boldsymbol{\phi}))\right] = \\ -H(Z \mid X;\boldsymbol{\theta}) - \mathbb{E}_{X,Z}[\log f_{\psi^*}(\boldsymbol{z})] + \alpha\mathbb{E}_{X,Z,\hat{Z}}\left[D_{\psi^*}(\boldsymbol{t}(\boldsymbol{z}),\hat{\boldsymbol{t}}_{\hat{z}}(\boldsymbol{\phi}))\right] + \mathbb{E}_{X,Z}\left[D_{\mathrm{KL}}(q(\hat{\boldsymbol{z}} \mid \boldsymbol{z};\boldsymbol{\phi}), q(\hat{\boldsymbol{z}}))\right]. \tag{34}$$

□

When minimizing Eq. 34 with respect to $\boldsymbol{\theta}$, the Bregman term encourages encoder $p(\boldsymbol{z} \mid \boldsymbol{x};\boldsymbol{\theta})$ that generates samples $\boldsymbol{z}$ whose sufficient statistics are close to one of the means $\hat{\boldsymbol{t}}_\kappa$ in terms of D^*_ψ. This term, in turn, encourages:

① encoders that collapse to a single atom $\hat{\boldsymbol{t}}_\kappa$: $q(\kappa \mid \boldsymbol{z};\boldsymbol{\phi}) \rightsquigarrow 1$. This is counterbalanced by the KL term of Eq. 34.

② low-entropy encoders that generate almost deterministic sufficient statistics for its samples: $\boldsymbol{t}(\boldsymbol{z}) \rightsquigarrow \hat{\boldsymbol{t}}_\kappa$. The negative entropy term in Eq. 34 helps avoid such degenerate solutions. A similar observation for the special case of a single Gaussian $q(\boldsymbol{z};\boldsymbol{\phi})$ (note the KL term in Eq. 34 vanishes and $\hat{Z}$ can be dropped in the second expectation in this case) following, however, an entirely algebraic route, is also made by Hoffman et al. (2017). Here, we present an information-theoretic perspective of this trade-off.

Keeping everything but $\boldsymbol{\phi}$ fixed, minimizing Eq. 29 over a finite number of sampled latent codes $\boldsymbol{z}$ is equivalent to MLE with a mixture distribution. In the case of a Gaussian mixture, this is equivalent to soft K-means clustering in the latent space in $\mathbb{R}^d$. For distributions for which $\boldsymbol{t}(\boldsymbol{z}) = \boldsymbol{z}$[13], minimizing Eq. 28 with respect to $\boldsymbol{\phi}$ amounts to computing the Rate Distortion Finite Cardinality (RDFC) function (Eq. 5) with the Bregman distortion D_{ψ^*} (Banerjee et al., 2004). The support $\hat{\mathcal{Z}}$ of $\hat{Z}$ to be learned has cardinality k and corresponds to the sufficient statistic means $\hat{\mathcal{Z}} = \{\hat{\boldsymbol{t}}_\kappa\}_{\kappa=1}^k$[14]. In our case, the log-likelihood of latent codes sampled by the encoder is maximized instead. Moreover, the source (encoder) is not apriori known but its parameters $\boldsymbol{\theta}$ are trainable during optimization. Using the decomposition of Eq. 34, the first two terms can be ignored since $H(Z \mid X; \boldsymbol{\theta})$ and $\log f_{\psi^*}(\boldsymbol{z})$ do not depend on $\boldsymbol{\phi}$.

D. Additional Experiments

D.1. Ablation Studies on CIFAR-10

In Table 6, we compare the OOD performance of DAB models when using other commonly-used OOD metrics. As expected, the proposed distortion score, that is *explicitly* minimized for the training datapoints via the loss function in Eq. 11, yields better OOD detection performance.

Table 6: **DAB performance with alternative OOD scores.** D_{KL} refers to the Kullback-Leibler distortion of Eq. 14. H refers to the entropy of the decoder's classifier: $H \triangleq \mathbb{E}_{Y,Z|\boldsymbol{x}}[-\log m(\boldsymbol{y} \mid \boldsymbol{z}; \boldsymbol{\theta})]$. Finally, p_{max} refers to the maximum probability of the classifier: $p_{\max} \triangleq \arg\max_c \mathbb{E}_{Z|\boldsymbol{x}}[m(Y = c \mid \boldsymbol{z}; \boldsymbol{\theta})]$. $p_{\max}$ and H are approximated by Monte Carlo with a single sample of Z. The Kullback-Leibler divergence from the learned centroids is more sensitive to input variations rendering the distortion of Eq. 14 a better indicator of an OOD input. Moreover, it is Monte Carlo sample-free for Gaussian encoders and centroids.

OOD score	**SVHN**		**CIFAR-100**	
	AUROC ↑	**AUPRC** ↑	**AUROC** ↑	**AUPRC** ↑
D_{KL}	$\mathbf{0.986 \pm 0.004}$	$\mathbf{0.994 \pm 0.002}$	$\mathbf{0.922 \pm 0.002}$	$\mathbf{0.915 \pm 0.002}$
H	0.964 ± 0.009	0.982 ± 0.005	0.891 ± 0.003	0.883 ± 0.003
$1 - p_{max}$	0.959 ± 0.009	0.978 ± 0.006	0.889 ± 0.003	0.875 ± 0.003

In the rest of this section, we study the effect of the DAB hyperparameters, also listed in Table 12, on the OOD performance of our model.

In Table 7, we do an ablation study on the RDFC cardinality k. We see that a larger number of centroids improves the quality of the uncertainty estimates. However, further increasing the codebook size with $k > 10$ yields diminishing performance benefits. Similar to Fig. 4, we sought to justify this model's behavior via visual inspection of the codebook. We noticed that when $k > 10$ some centroids are assigned to only a small number of training datapoints. This observation can serve as a recipe for choosing the codebook size: albeit a larger codebook will not harm performance, unutilized entries indicate that a smaller codebook can achieve similar quality for the model's uncertainty estimates.

Table 7: **Ablation study over codebook size k.** A single Gaussian code $q(\boldsymbol{z})$ does not discriminate well CIFAR-10 from the visually similar datapoints of CIFAR-100. As we increase the number of centroids, DAB progressively becomes better at distinguishing these datasets. DAB reaches competitive performance with a small number of 10 centroids. The performance remains roughly the same when using a larger cardinality $k > 10$.

	SVHN		**CIFAR-100**	
	AUROC ↑	**AUPRC** ↑	**AUROC** ↑	**AUPRC** ↑
$k = 10$	$\mathbf{0.986 \pm 0.004}$	$\mathbf{0.994 \pm 0.002}$	$\mathbf{0.922 \pm 0.002}$	$\mathbf{0.915 \pm 0.002}$
$k = 5$	0.968 ± 0.031	0.986 ± 0.012	0.912 ± 0.009	0.907 ± 0.007
$k = 1$ vanilla VIB (Alemi et al., 2017)	0.906 ± 0.052	0.958 ± 0.026	0.746 ± 0.023	0.764 ± 0.026

In Table 8, we study the effect of the temperature α (Eq. 12). We verify that α controls the strength of the statistical distance when comparing a datapoint with the codebook. For small values of α, the model exhibits a uniformity-tolerance for the datapoints that lie well beyond the support of the training dataset. On the other hand, the distribution $\pi_{\boldsymbol{x}}$ (Eq. 12) becomes

[13] For example, for Gaussian $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with constant and a priori known covariance matrix $\boldsymbol{\Sigma}$, $\boldsymbol{t}(\boldsymbol{z}) = \boldsymbol{z}$, $\boldsymbol{\phi} = \boldsymbol{\mu} = \hat{\boldsymbol{t}}$.

[14] We implicitly redefine $\hat{Z}$ to take values in $\hat{\mathcal{Z}} = \{\hat{\boldsymbol{t}}_\kappa\}_{\kappa=1}^k$.

sharper for larger values of α. A sharper distribution translates to a more informative centroid assignment for datapoint $\boldsymbol{x}$. Subsequently, an informative codebook helps the model to successfully mark the areas of the input distribution that is familiar with.

Table 8: **Ablation study over temperature $\boldsymbol{\alpha}$.** With small values of α, the model fails to discriminate inputs successfully, which it should be less confident about. Large values of α lead to a more concentrated assignment of the training datapoints to the centroids. This, in turn, provides the model with more effective OOD scores that sufficiently penalize large distances from the codebook.

	SVHN		**CIFAR-100**	
	AUROC ↑	**AUPRC** ↑	**AUROC** ↑	**AUPRC** ↑
$\alpha = 0.1$	0.932 ± 0.038	0.972 ± 0.018	0.756 ± 0.031	0.776 ± 0.032
$\alpha = 0.5$	0.958 ± 0.045	0.982 ± 0.019	0.878 ± 0.057	0.879 ± 0.043
$\alpha = 1.0$	$\mathbf{0.986 \pm 0.004}$	$\mathbf{0.994 \pm 0.002}$	$\mathbf{0.922 \pm 0.002}$	$\mathbf{0.915 \pm 0.002}$
$\alpha = 2.0$	0.989 ± 0.003	0.995 ± 0.001	0.924 ± 0.001	0.918 ± 0.002
$\alpha = 10.0$	0.982 ± 0.005	0.991 ± 0.002	0.923 ± 0.002	0.916 ± 0.002

In Table 9, we vary the regularization coefficient β (Eq. 11). We see that the model achieves the best performance within a range of β. For smaller values of β, the distortion term in Eq. 11 is disregarded. Therefore, the main network is not restricted to producing encoders that can be well-represented by the codebook. For larger values of β, the training datapoints get closely attached to the centroids. This results in statistical balls of small radius (Fig. 2c) effectively leaving out novel, in-distribution datapoints.

Table 9: **Ablation study over regularization coefficient $\boldsymbol{\beta}$.** The model is best performing within a range of values. Large values of β correspond to small balls around the centroids (Fig. 2c) and vice-versa. The balls should be small enough to exclude OOD inputs but large enough to include unseen, in-distribution points to which the model can generalize.

	SVHN		**CIFAR-100**	
	AUROC ↑	**AUPRC** ↑	**AUROC** ↑	**AUPRC** ↑
$\beta = 0.0001$	0.925 ± 0.429	0.965 ± 0.02	0.70 ± 0.019	0.697 ± 0.02
$\beta = 0.0005$	0.98 ± 0.009	0.99 ± 0.005	0.917 ± 0.002	0.91 ± 0.003
$\beta = 0.001$	$\mathbf{0.986 \pm 0.004}$	$\mathbf{0.994 \pm 0.002}$	$\mathbf{0.922 \pm 0.002}$	$\mathbf{0.915 \pm 0.002}$
$\beta = 0.005$	0.985 ± 0.004	0.993 ± 0.002	0.921 ± 0.002	0.914 ± 0.002
$\beta = 0.01$	0.977 ± 0.01	0.988 ± 0.005	0.914 ± 0.002	0.907 ± 0.001

In Table 10, we are sweeping the bottleneck dimension. In Table 3, we see that 8-dimensional latent features can capture the information needed for the CIFAR-10 classification task. Further increasing the bottleneck size leads to irrelevant features that have no effect. On the other hand, smaller features disregard essential aspects of the input.

Table 10: **Ablation study over bottleneck dimension.** Larger latent features improve OOD capability until a performance plateau is reached.

	SVHN		**CIFAR-100**	
	AUROC ↑	**AUPRC** ↑	**AUROC** ↑	**AUPRC** ↑
$\dim(\boldsymbol{z}) = 2$	0.748 ± 0.03	0.797 ± 0.014	0.678 ± 0.014	0.59 ± 0.008
$\dim(\boldsymbol{z}) = 4$	0.974 ± 0.01	0.98 ± 0.004	0.877 ± 0.012	0.872 ± 0.008
$\dim(\boldsymbol{z}) = 8$	$\mathbf{0.986 \pm 0.004}$	$\mathbf{0.994 \pm 0.002}$	$\mathbf{0.922 \pm 0.002}$	$\mathbf{0.915 \pm 0.002}$
$\dim(\boldsymbol{z}) = 10$	0.983 ± 0.005	0.991 ± 0.003	0.924 ± 0.002	0.915 ± 0.001

Finally, the model was not sensitive to typical values, i.e. > 0.9, for the momentum γ.

D.2. DAB for detecting CIFAR-10 with noise corruptions

Table 5 shows the AUROC scores for the DAB of Section 6.2 on test CIFAR-10 versus test CIFAR-10 with common noise corruptions (Hendrycks & Dietterich, 2019).

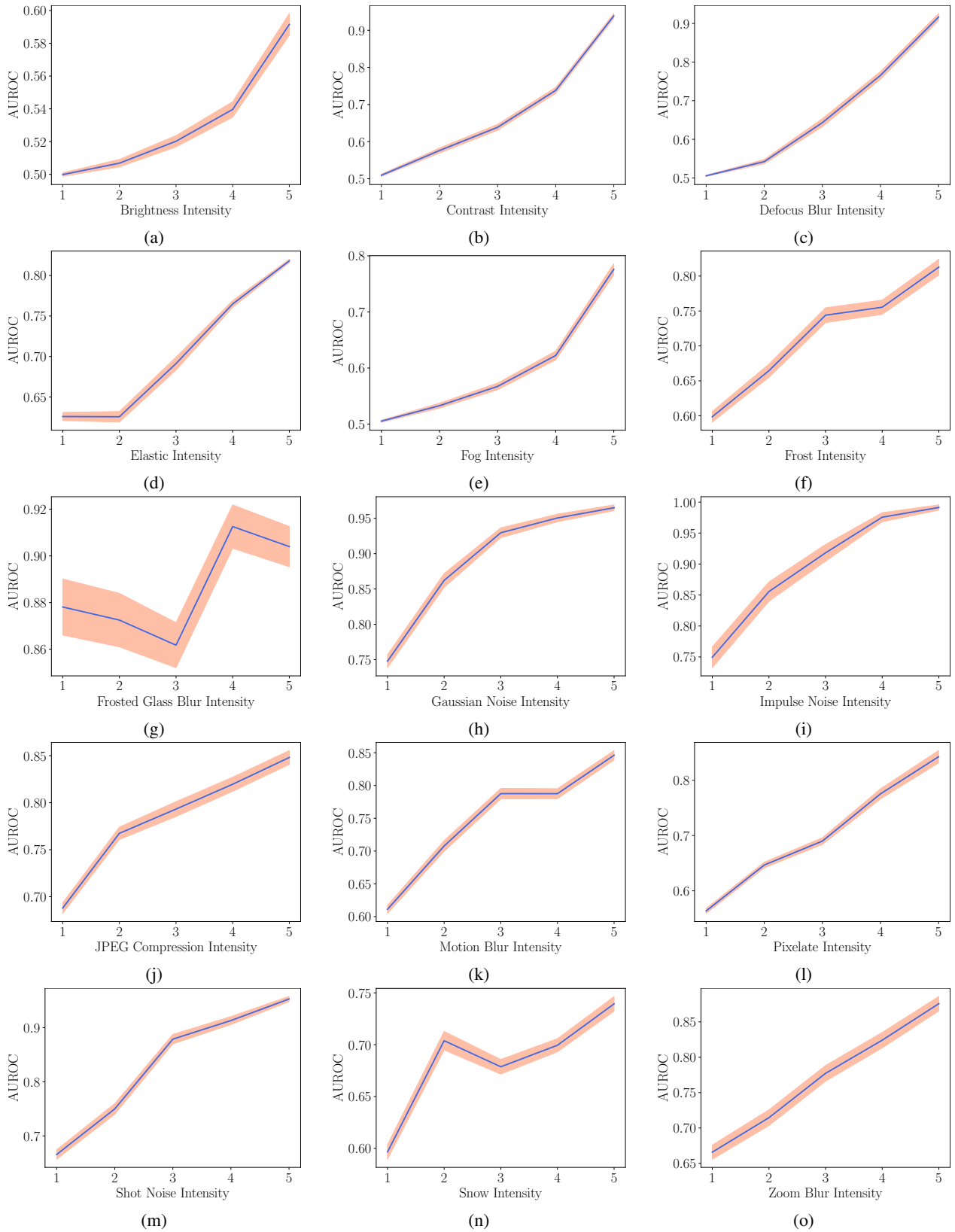


Figure 5: DAB's AUROC vs corruption intensity for common corruptions to test CIFAR. The shaded area corresponds to $+/-$ one standard deviation across 10 random seeds.

D.3. Qualitative Evaluation of DAB on CIFAR-10 (Section 6.2 continued.)

In Fig. 6 and Fig. 7, we also investigate qualitatively the rest of the IB methods examined in Sections 6.2 and D.1 (Tables 2, 7). Finally, Fig. 8 visualizes DAB's calibration demonstrating that model's accuracy negatively correlates with its uncertainty.

Figure 6: **Qualitative evaluation with 5 entries.** We visualize the number of test data points per class assigned to each centroid at the end of three (first, middle, last) iterations of our alternating minimization algorithm (Algorithm 1). We notice that semantically similar classes are assigned to the same code. For example, dogs (class 5) and cats (class 3) are both represented by centroid 3. Similar observations hold for the pair of cars (class 1)/ trucks (class 9) and airplane (class 0)/ ships (class 8).

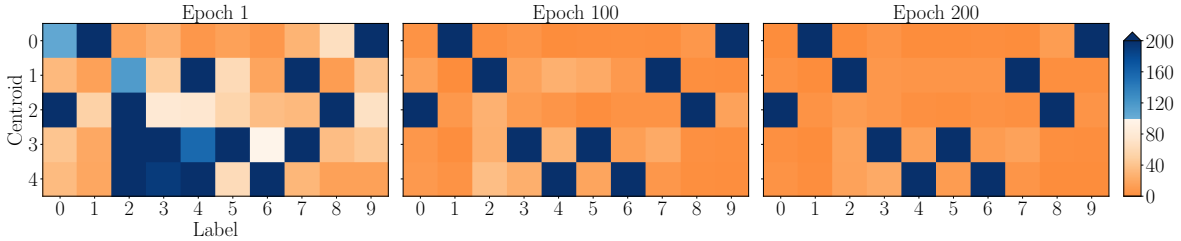
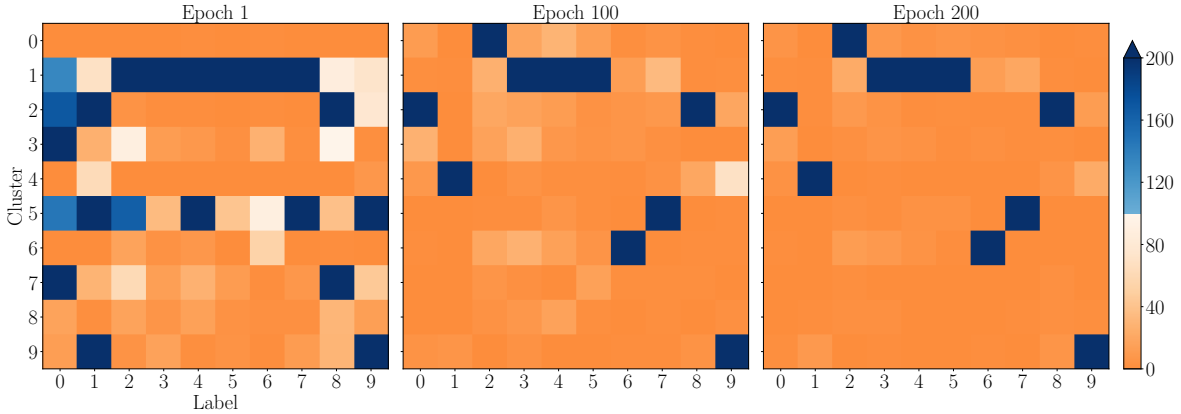


Figure 7: **Qualitative evaluation of a $10-$component mixture marginal in the VIB trained with gradient descent.** We visualize the number of test data points per class assigned to each component at the end of three (first, middle, last) epochs when the mixture variational marginal $q(\boldsymbol{z};\boldsymbol{\phi})$ and the rest of the network (encoder and decoder) are jointly trained via gradient descent (Alemi et al., 2018). We notice that gradient descent conflates features of different classes. This observation can help explain the inferior performance of the IB gradient descent method on OOD tasks (Table 2). Moreover, it justifies the need for guiding optimization through the alternating minimization steps of Algorithm 1.



We verify that misclassification of a test datapoint is signaled by its large distance from the codebook.

D.4. Out-of-Distribution Detection on UCI Regression Tasks

Currently, the bulk of uncertainty-aware methods is designed for and applied to image classification in supervised learning settings. However, as shown by Jaeger et al. (2023), a wide range of tasks and datasets should be considered when evaluating OOD methods. Moreover, there is an ongoing importance to effective uncertainty estimation for regression tasks, especially in unsupervised learning scenarios. For example, in deep reinforcement learning, uncertainty quantification for the estimated Q-values can be leveraged for efficient exploration (Lee et al., 2021). As already pointed out, DAB provides a unified notion of uncertainty for both regression and classification tasks.

In Table 11, we test the OOD capability of our model when trained on the normalized UCI, Energy Efficiency dataset (Markelle Kelly, 1998). As in the image classification tasks, the positive label corresponds to the OOD inputs. The results were averaged across 10 runs. We contrast our model with ensemble methods. We see that DAB consistently demonstrates OOD capability and outperforms 4-member ensembles on all OOD tasks (of varying difficulty). In Section E.4, we provide the experimental details. Here, we comment that all centroids were assigned a roughly equal number of datapoints indicating the need for codebook sizes larger than one (recall that DAB with a unit-size codebook corresponds to the standard VIB (Alemi et al., 2017)).

Figure 8: **Calibration plot of DAB on CIFAR-10 test data.** We qualitatively assess the proposed uncertainty score in terms of calibration. We train 10 models with different random seeds. For each model, we find the 20 quantiles of the estimated uncertainty on test data. We compute the accuracy for the datapoints whose uncertainty falls between two successive quantiles. We report the mean uncertainty and accuracy along with one standard deviation error bars across the runs. We see that the accuracy is higher in the quantile buckets of lower uncertainty.

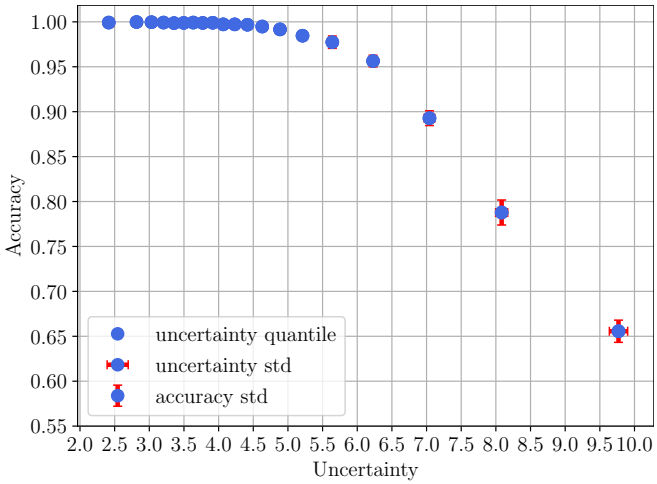


Table 11: **DAB's OOD performance on the UCI energy efficiency dataset.** DAB can consistently and competitively solve a diversity of OOD regression tasks.

OOD Dataset	Model	OOD Scores	
		AUROC ↑	AUPRC ↑
kin8nm	DAB	$\mathbf{0.982 \pm 0.008}$	$\mathbf{0.998 \pm 0.001}$
	Ensemble of 2	0.916 ± 0.025	0.992 ± 0.003
	Ensemble of 4	0.977 ± 0.008	$\mathbf{0.998 \pm 0.001}$
concrete strength	DAB	$\mathbf{0.978 \pm 0.011}$	$\mathbf{0.988 \pm 0.006}$
	Ensemble of 2	0.898 ± 0.043	0.941 ± 0.028
	Ensemble of 4	0.967 ± 0.02	0.979 ± 0.013
protein structure	DAB	$\mathbf{0.989 \pm 0.017}$	$\mathbf{1.0 \quad \pm 0.001}$
	Ensemble of 2	0.875 ± 0.059	0.998 ± 0.001
	Ensemble of 4	0.971 ± 0.018	0.999 ± 0.001
boston housing	DAB	$\mathbf{0.988 \pm 0.008}$	$\mathbf{0.988 \pm 0.007}$
	Ensemble of 2	0.888 ± 0.043	0.887 ± 0.047
	Ensemble of 4	0.969 ± 0.028	0.967 ± 0.03

E. Implementation details

E.1. Implementation details for the synthetic regression tasks (Section 6.1)

For the example of Fig. 3a, we generate 20 training data points from the uniform distribution $\mathcal{U}[-4, 4]$. The test data are evenly taken in $[-5, 5]$. The targets are $y = x^3 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 9)$. We use a single centroid to represent the whole dataset. We verify that the model's confidence and accuracy decline as we move far away from the data. In Fig. 3b, we stress test our model under a harder variant of the first problem. In this case, we create two clusters of training data points sampled from $\mathcal{U}[-5, -2]$ and $\mathcal{U}[2, 5]$. We use two codes.

We use a network with 3 dense layers. We apply DAB to the last one. The intermediate layers have 100 hidden units and ELU non-linearity. We perform 1500 training iterations. We use a single encoder sample during training. The optimizer of both the main network and the centroids are set to `tf.keras.optimizers.Adam` with initial learning rates $\eta_{\boldsymbol{\theta}} = 0.001$ and $\eta_{\boldsymbol{\phi}} = 0.01$ respectively. The rest of the hyperparameters are set to the default values of `tf.keras.keras.layers.Dense`. Regarding the parametrization and initialization of encoders and centroids, we follow the setup described in Section E.2.

In Table 12, we provide the hyperparameters related to DAB. Note that the dataset for these tasks consists of only 20 datapoints. Therefore, we can use the whole dataset at each gradient update step. In this case, there is no need to maintain moving averages for the update of the assignment probabilities and covariance matrices.

Table 12: **A summary of DAB hyperparameters for the synthetic regression tasks.**

Hyperparameter	Description	Value
β	Regularization coefficient (Eq. 11)	1.0
α	Temperature (Eq. 12)	5.0
$\dim(\boldsymbol{z})$	Dimension of latent features	8
k	Number of centroids	1 for Fig. 3a 2 for Fig. 3b
γ	Momentum of moving averages (Eq. 27)	0.0

E.2. Implementation details for the CIFAR-10 experiments (Tables 2, 4, 3, Sections D.3, D.1)

All models are trained on four 32GB V100 GPUs. The per-core batch size is 64. For fair comparisons, we train all models for 200 epochs. This number may deviate from the suggested setup of some baselines such as RegMixup (Pinto et al., 2022b) or DDU (Mukhoti et al., 2023) which are originally trained for 350 epochs. For the IB and GP methods, we backpropagate through a single sample.

DAB is interleaved between the Wide ResNet 28-10 features (right after the flattened average pooling layer) and the last dense layer of the classifier. In this experiment, we use a full-covariance multivariate Gaussian for the encoder and the centroids. The encoder's network first learns a matrix $\boldsymbol{S}$ as $\boldsymbol{S} = \boldsymbol{U}\sqrt{\boldsymbol{\Lambda}}$. $\boldsymbol{U}$ is a unitary matrix. $\boldsymbol{\Lambda}$ is a positive definite, diagonal matrix. $\boldsymbol{U}$ and $\boldsymbol{\Lambda}$ are computed from the SVD decomposition of a symmetric matrix. To enforce positive definiteness of $\boldsymbol{\Lambda}$ with small initial values, we transform its entries by $\texttt{softplus}(\lambda - 5.0)$. A similar transformation was used by Alemi et al. (2017). Finally, the covariance matrix is given by: $\boldsymbol{\Sigma} = \boldsymbol{S}\boldsymbol{S}^T$.

We train the means of the centroids using `tf.keras.optimizers.Adam(learning_rate=1e-1)`. Only for the case $k = 1$ in Table 7, we used `tf.keras.optimizers.Adam(learning_rate=1e-3)`.The centroid means are initialized with `tf.random_normal_initializer(mean=0.0,stddev=0.1)`. For the hyperparameters that are not related to the DAB, we preserve the default values used in:
https://github.com/google/uncertainty-baselines/blob/main/baselines/cifar/deterministic.py.

Table 13: **A summary of DAB hyperparameters for the CIFAR-10 classification tasks.**

Hyperparameter	Description	Value
β	Regularization coefficient (Eq. 11)	0.001
α	Temperature (Eq. 12)	1.0
$\dim(\boldsymbol{z})$	Dimension of latent features	8
k	Number of centroids	10
γ	Momentum of moving averages (Eq. 27)	0.99

E.3. Implementation details for the ImageNet-1K experiments (Table 5)

All models are trained on four 48GB RTX A6000 GPUs. The per-core batch size is 256. We initialize the network with the weights of a pre-trained ResNet-50 network[15]. We train DAB only for 70 epochs.

The ResNet-50 features (without including the fully-connected layer at the top of the network) are first passed through three fully connected layers, each with 2048 units and ReLU activation. We add a residual connection between the first and last dense layer before DAB's input. In this experiment and due to the higher dimension of the latent features, we use diagonal multivariate Gaussians for the encoder and the centroids. The encoder's scale matrix is given by $\boldsymbol{S} =$

[15] `tf.keras.applications.resnet50.ResNet50`

$\texttt{diag}(\texttt{softplus}(\boldsymbol{o} - 5.0))$, where $\boldsymbol{o}$ are the encoder's outputs corresponding to the covariance. The covariance matrix is given by: $\boldsymbol{\Sigma} = \boldsymbol{S}\boldsymbol{S}^T$. Finally, we use Eq. 9 of Davis & Dhillon (2006) to update the codebook's covariance matrices where only the diagonal entries are computed.

To improve model's calibration, we add a max margin-loss term in the objective function of Eq. 11 for the misclassified datapoints:

$$\ell(\boldsymbol{x}) = \max(0, \mathrm{U}_{lb} - \mathrm{uncertainty}(\boldsymbol{x})). \tag{35}$$

$\mathrm{uncertainty}(\boldsymbol{x})$ is defined in Eq. 14. This term encourages higher model's uncertainty for the mispredicted training datapoints. In the experiment, we set the uncertainty lower bound as $\mathrm{U}_{lb} = 100$. Moreover, only the correctly classified training examples are quantized by the codebook (Eq. 8). For the OOD experiments, we quantize all training datapoints regardless the classification outcome. Therefore, the loss term in Eq. 35 is omitted.

Table 14 provides DAB's hyperparameters when we backpropagate to ResNet-50. Table 15 provides DAB's hyperparameters when the ResNet-50 weights are frozen during training. In the first case, the main network is trained using `tf.keras.optimizers.SGD(learning_rate=1e-1)`. In the second case, encoder's dense layers and the decoder are trained using `tf.keras.optimizers.SGD(learning_rate=5e-2)`. We train centroids' means using `tf.keras.optimizers.Adam`. The centroid means are initialized with `tf.random_normal_initializer(mean=0.0,stddev=0.1)`. For the rest of the hyperparameters, we preserve the default values used in:
https://github.com/google/uncertainty-baselines/blob/main/baselines/imagenet/deterministic.py.

Table 14: **Hyperparameters for DAB with ResNet-50 fine-tuning for the ImageNet-1K classification tasks (Table 5).**

Hyperparameter	Description	Value
η_ϕ	Codebook's learning rate	0.4 for OOD 0.1 for Calibration
β	Regularization coefficient (Eq. 11)	0.005 for OOD 0.01 for Calibration
α	Temperature (Eq. 12)	2.0
$\dim(\boldsymbol{z})$	Dimension of latent features	80
k	Number of centroids	1000
γ	Momentum of moving averages (Eq. 27)	0.99

Table 15: **Hyperparameters for DAB without ResNet-50 fine-tuning for the ImageNet-1K classification tasks (Table 5).**

Hyperparameter	Description	Value
η_ϕ	Codebook's learning rate	0.4 for OOD 0.1 for Calibration
β	Regularization coefficient (Eq. 11)	0.0025 for OOD 0.02 for Calibration
α	Temperature (Eq. 12)	2.0
$\dim(\boldsymbol{z})$	Dimension of latent features	80
k	Number of centroids	1000
γ	Momentum of moving averages (Eq. 27)	0.99

E.4. Implementation details for the UCI regression experiments (Section D.4)

The optimizer of the codebook was set to `tf.keras.optimizers.Adam(learning_rate=1e-1)`. The architecture consists of an MLP network with one hidden layer of dimension 50 and ReLU nonlinearity. The exact backbone architecture along with the hyperparameters that are not related to the DAB were kept the same and can be found here: https://github.com/google/uncertainty-baselines/tree/main/baselines/uci.

As for the rest of experiments, we applied DAB between the penultimate and output layer of the architecture. Regarding the parametrization and initialization of encoders and centroids, we follow the setup described in Section E.2. Table 16 reports the DAB hyperparameters used for this task.

Table 16: **A summary of DAB hyperparameters for the UCI regression tasks.**

Hyperparameter	Description	Value
β	Regularization coefficient (Eq. 11)	0.001
α	Temperature (Eq. 12)	1.0
$\dim(\boldsymbol{z})$	Dimension of latent features	4
k	Number of centroids	2
γ	Momentum of moving averages (Eq. 27)	0.99