PREDICTIVE UNCERTAINTY THROUGH QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

High-risk domains require reliable confidence estimates from predictive models. Deep latent variable models provide these, but suffer from the rigid variational distributions used for tractable inference, which err on the side of overconfidence. We propose Stochastic Quantized Activation Distributions (SQUAD), which imposes a flexible yet tractable distribution over discretized latent variables. The proposed method is scalable, self-normalizing and sample efficient. We demonstrate that the model fully utilizes the flexible distribution, learns interesting non-linearities, and provides predictive uncertainty of competitive quality.

1 INTRODUCTION

In high-risk domains, prediction errors come at high costs. Luckily such domains often provide a failsafe in the form of rejecting a prediction: self-driving cars perform an emergency stop, doctors can run another diagnostic test, and industrial processes can be temporarily halted. For deep learning models, this can be achieved via a predetermined threshold on predicted confidence scores. As such, a low error rate can be guaranteed at the cost of rejecting some predictions. However, estimating confidence scores from neural networks that create well-ordered rankings of correct and incorrect predictions remains an active area of research.

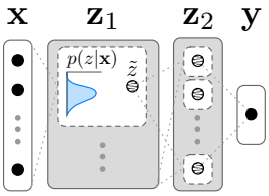

Figure 1: DLVMs have layers of stochastic latent variables.

Deep Latent Variable Models (DLVMs, fig. 1) approach this by postulating latent variables $\mathbf{z}$ for which the uncertainty in $p(\mathbf{z}|\mathbf{x})$ influences the confidence in the target prediction. Recently, efficient inference algorithms have been proposed in the form of variational inference, where an inference neural network is optimized to predict parameters of a variational distribution that approximates an otherwise intractable distribution (Kingma & Welling (2013); Rezende et al. (2014); Alemi et al. (2016); Achille & Soatto (2016)).

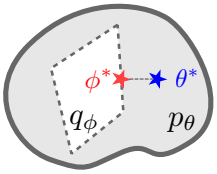

Figure 2: A distribution q_ϕ is optimized to approximate p_{θ^*}.

Variational inference relies on a tractable class of distributions that can be optimized to closely resemble the true distribution (fig. 2), and it's hypothesized that more flexible classes lead to more faithful approximations and thus better performance (Jordan et al. (1999)). To explore this hypothesis we propose a novel tractable class of highly flexible variational distributions. Considering that neural networks with low-precision activations exhibit good performance (Holi & Hwang (1993); Hubara et al. (2016)), we make the modeling assumption that latent variables can be expressed under a strong quantization scheme, without loss of predictive fidelity. If this assumption holds, it becomes tractable to model a scalar latent variable with a flexible multinomial distribution over the values (fig. 3).

By repositioning the variational distribution from a potentially limited description of moments – as found in exponential families – to a direct expression of probabilities per value, a variety of benefits arise. As the output domain is constrained, the method becomes self-normalizing, relieving the model from poorly scalable batch normalization techniques (Ioffe & Szegedy (2015)). More interesting priors can be explored outside the space of exponential families and the model is able to learn non-linearities per neuron.

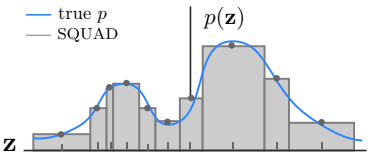

Figure 3: SQUAD quantizes the domain of z to model a flexible and tractable variational distribution.

More concretely, the contributions of this work are as follows:

- We propose a novel variational inference method by leveraging multinomial distributions on quantized latent variables.
- We show that the emerging predicted distributions are multimodal, motivating the need for flexible distributions.
- We demonstrate that the proposed method applied to the information bottleneck objective computes competitive uncertainty over the predictions and that this manifests in better accuracy under strong risk guarantees.

2 BACKGROUND

In this work, we explore deep neural networks for regression and classification. We have data-points $(\mathbf{x}, \mathbf{y})$ in a dataset $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i) \mid i \in [1, ..., N]\}$ and postulate latent variables $\mathbf{z}$ that represent the data. We focus on the Information Bottleneck (IB) perspective: first proposed by Tishby et al. (2000), the information bottleneck objective $I(\mathbf{y}, \mathbf{z}; \theta) - \beta I(\mathbf{x}, \mathbf{z}; \theta)$ is optimized to maximize the mutual information between $\mathbf{z}$ and $\mathbf{y}$, whilst minimizing the mutual information between $\mathbf{z}$ and $\mathbf{x}$. The objective can be efficiently optimized using a variational inference scheme as shown concurrently by both Alemi et al. (2016) and Achille & Soatto (2016). Under the Markov assumption $P(\mathbf{z}, \mathbf{x}, \mathbf{y}) = p(\mathbf{z}|\mathbf{x})p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$, they derive the following lower bound:

$$I(\mathbf{y}, \mathbf{z}; \theta) - \beta I(\mathbf{x}, \mathbf{z}; \theta) \geq \mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{p(\mathbf{z}|\mathbf{x}_n)}[\log q(\mathbf{y}_n|\mathbf{z})] - \beta D_{\mathrm{KL}}[p(\mathbf{z}|\mathbf{x}_n) \| r(\mathbf{z})], \tag{1}$$

where $\mathbb{E}_{p(\mathbf{z}_n|\mathbf{x}_n)}$ is commonly estimated using a single Monte Carlo sample and $r(\mathbf{z})$ is a variational distribution to the marginal distribution of $\mathbf{z}$. In practice $r(\mathbf{z})$ is fixed to a simple distribution such as a spherical Gaussian. Alemi et al. (2016) and Achille & Soatto (2016) continue to show that the Variational Auto Encoder (VAE) Evidence Lower Bound (ELBO) proposed in Kingma & Ba (2014); Rezende et al. (2014) is a special case of the IB bound when $\mathbf{y} = \mathbf{x}$ and $\beta = 1$:

$$I(z, x) - \beta I(z, i) \geq \mathbb{E}_{p(\mathbf{z}|\mathbf{x}_n)}[\log q_\theta(\mathbf{x}_n|\mathbf{z})] - D_{\mathrm{KL}}[p(\mathbf{z}|\mathbf{x}_n) \| r(\mathbf{z})], \tag{2}$$

where i represents the identity of data-point $\mathbf{x}_i$. Interestingly, the VAE perspective considers the bound to optimize a variational distribution $q(\mathbf{z}|\mathbf{x})$, whilst the IB perspective prescribes that $q(\mathbf{z}|\mathbf{x})$ in the ELBO is not a variational posterior but the true encoder $p(\mathbf{z}|\mathbf{x})$, and instead $p(\mathbf{y}|\mathbf{z})$ and $p(\mathbf{z})$ are the distributions approximated with variational counterparts.

From yet another perspective, equation 1 can be interpreted as a domain-translating beta-VAE (Higgins et al. (2016)), where an input image is encoded into a latent space and decoded into the target domain. The Lagrange multiplier β then controls the trade-off between rate and distortion, as argued by Alemi et al. (2017).

In this work, we follow the IB interpretation of the bound in equation 1 and leave the evaluation of our proposed variational inference scheme in other models such as the VAE for further work.

3 METHOD

At the heart of our proposal lies the assumption that neuron networks can be effective under strong activation quantization schemes. For clarity, we start with presenting the derivation for a single layer information bottleneck. We model the encoder using a mean-field approximation $p(\mathbf{z}|\mathbf{x}) = \prod_k p(z_k|\mathbf{x})$. We individually quantize the *domain* of each of the K scalar latent variables z_k into a small set $\mathbf{v} = v_1, \ldots, v_C$ consisting of a small number of C different real values (e.g. $C = 11$). We use a neural network $f_\theta(\cdot)$ with a softmax output over the quantization bins to model the distributions over the values for z:

$$p_\theta(z = v_c|\mathbf{x}) = f_\theta(\mathbf{x})_c = \mathrm{softmax}(\mathbf{W}_c\mathbf{x} + \mathbf{b}_c). \tag{3}$$

To optimize the parameters θ with Stochastic Gradient Descent (SGD), we need to derive a fully differentiable loss for our model. Providing figure 4 for visual reference, we derive the bound as

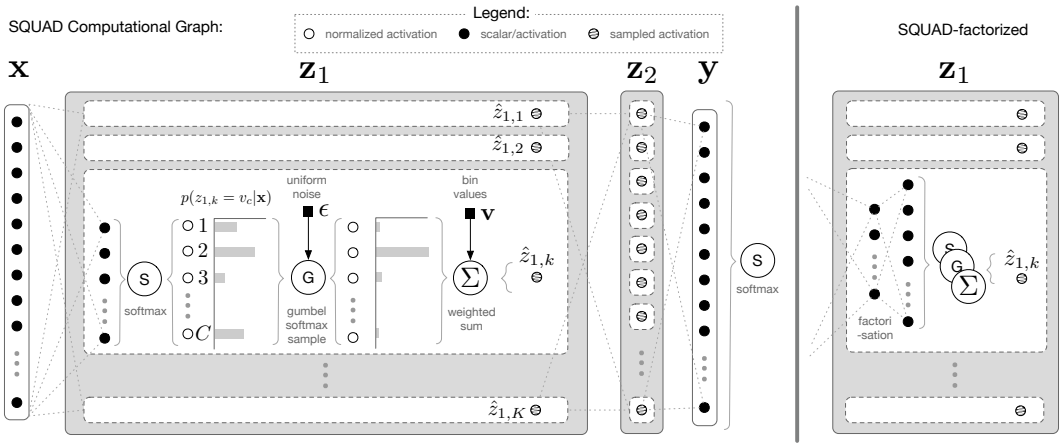

Figure 4: The left diagram visualizes the computational graph of SQUAD at training time, providing a detailed view on how an individual latent variable is sampled. The right diagram visualizes how the proposed matrix-factorization variant improves the parameter efficiency of the model.

follows, starting with the single-data-point loss in equation 1:

$$\mathcal{L} = \mathbb{E}_{p(\mathbf{z}|\mathbf{x})}[\log q(\mathbf{y}|\mathbf{z})] - \beta D_{\mathrm{KL}}[p(\mathbf{z}|\mathbf{x}) \| r(\mathbf{z})]. \tag{4}$$

We re-parametrize $\mathbf{z}$ as a set of bin-indexing variables $s_k \in \{1, \ldots, C\}$:

$$p(s_k = c|\mathbf{x}) \triangleq p(z_k = v_c|\mathbf{x}). \tag{5}$$

We model $q_\theta(\mathbf{y}|\mathbf{z})$ with a softmax layer $f_\theta(\mathbf{z}) = \mathrm{softmax}(\mathbf{W}\mathbf{z} + \mathbf{b})$ (*abusing notation to indicate element-wise vector indexing with* $\mathbf{v}[\mathbf{s}]$):

$$\mathcal{L} = \mathbb{E}_{\mathbf{s} \sim p}[\log f_\theta(\mathbf{v}[\mathbf{s}])] - \beta D_{\mathrm{KL}}[\ldots].$$

To enable sampling from the discrete variables $\mathbf{s}$, we use the Gumbel-Max trick (Gumbel (1954)), denoted $\mathrm{gumb}()$, re-parameterizing the expectation $\mathbb{E}_{\mathbf{s} \sim p}$ with uniform noise $\epsilon \sim U(0, 1)$:

$$= \mathbb{E}_\epsilon \left[\log f_\theta \left(\mathbf{v} \left[\arg\max_c \mathrm{gumb}(p(\mathbf{s}|\mathbf{x}), \epsilon) \right] \right) \right] - \beta D_{\mathrm{KL}}[\ldots]. \tag{6}$$

As the argmax is not differentiable, we approximate this expectation using the Gumbel-Softmax trick (Maddison et al. (2016); Jang et al. (2016)), which generates samples that smoothly deform into one-hot samples as the softmax temperature τ approaches 0:

$$\approx \mathbb{E}_\epsilon \left[\log f_\theta \left(\mathbf{v} \cdot \operatorname*{softmax}_c \mathrm{gumb}(p(\mathbf{s}|\mathbf{x}), \epsilon) \right) \right] - \beta D_{\mathrm{KL}}[\ldots]. \tag{7}$$

In practice, we anneal τ from 1.0 to 0.5 during the training process, as proposed by Yang et al. (2017) to reduce gradient variance initially, at the risk of introducing bias.

For the remainder of this work, we will refer to the latent variables as $\mathbf{z}$ in lieu of $\mathbf{s}$, for clarity. To conclude our derivation, we use a fixed SQUAD distribution to model the marginal $p(\mathbf{z}) := r(\mathbf{z})$ as shown in figure 5. We can then derive the KL term analytically following the definition for discrete distributions. Using the fact that the KL divergence is additive for independent variables:

$$\mathcal{L} = \mathbb{E}_\epsilon \left[\log f_\theta \left(\mathbf{v} \cdot \operatorname*{softmax}_c \mathrm{gumb}(p(\mathbf{s}|\mathbf{x}), \epsilon) \right) \right] - \beta \sum_{k=1}^{K} \sum_{c=1}^{C} p(z_k = v_c|\mathbf{x}) \log \frac{p(z_k = v_c|\mathbf{x})}{p_\theta(z_k = v_c)} \tag{8}$$

The derived lower bound is fully differentiable, allowing it to be optimized with common SGD-based methods.

At test time, we can approximate the predictive function $p(y^*|\mathbf{x}^*)$ for a new data-point $\mathbf{x}^*$ by taking T samples from the latent variables $\mathbf{z}$ i.e. $\hat{\mathbf{z}}_t \sim p(\mathbf{z}|\mathbf{x}^*)$, and averaging the predictionss for y^*:

$$p_\theta(y^*|\mathbf{x}^*) \approx \int q_\theta(y^*|\mathbf{z}) p_\theta(\mathbf{z}|\mathbf{x}^*) d\mathbf{z} \approx \frac{1}{T} \sum_{t=1}^{T} q_\theta(y^*|\hat{\mathbf{z}}_t). \tag{9}$$

We improve the flexibility of the proposed model by creating a hierarchical set of latent variables. The joint distribution of L layers of latents is then:

$$p_\theta(\mathbf{z}_1, \ldots, \mathbf{z}_L|\mathbf{x}) = p_\theta(\mathbf{z}_L|\mathbf{z}_{L-1}) \cdots p_\theta(\mathbf{z}_1|\mathbf{x}), \tag{10}$$

and is straightforwardly implemented with a simple ancestral sampling scheme. Finally, we have $q_\theta(\mathbf{y}|\mathbf{z}_1, \ldots, \mathbf{z}_L) = q_\theta(\mathbf{y}|\mathbf{z}_L)$.

Interestingly, the strong quantization proposed in our method can itself be considered an additional information bottleneck, as it exactly upper-bounds the number of bits per latent variable. Such bottlenecks are theorized to have a beneficial effect on generalization (Tishby et al. (2000); Achille & Soatto (2016); Alemi et al. (2017; 2016)), and we can directly control this bottleneck by varying the number of quantization bins.

The computational complexity of the method, as well as the number of model parameters θ, scale linearly in C, i.e. $O(C)$ (with C the number of quantization bins). It is thus suitable for large-scale inference. We would like to stress that the proposed method differs from work that leverages the Gumbel-Softmax trick to model categorical latent variables: our proposal models continuous scalar latent variables by quantizing their domain and modeling belief with a multinomial distribution. Categorical latent variable models would incur a much larger polynomial complexity penalty of $O(C^2)$.

The method is easily integrated into existing deep neural network architectures, and SQUAD layer implementations are provided at `github.com/anonymized`.

Matrix-factorization variant To improving the tractability of using a large number of quantization bins, we propose a variant of SQUAD that uses a matrix factorization scheme to improve the parameter efficiency. Formally, equation equation 3 becomes:

$$p(z_k = v_c|\mathbf{x}) = \mathrm{softmax}(\mathbf{w}''_{k,c}(\mathbf{w}'_k \mathbf{x} + \mathbf{b}'_k) + \mathbf{b}''_{k,c}),$$

with full layer weights $\mathbf{W}''$ and $\mathbf{W}'$ respectively of shape (K, B, C) and $(|X|, K, B)$, where K denotes the number of neurons, B the number of factorization, C number of quantization bins and $|X|$ the input dimensionality. The right side of figure 4 provides visual clarification of this variant. To improve the parameter efficiency, we can learn $\mathbf{W}'$ per layer as well, resulting in shape $(|X|, 1, B)$, which is found to be beneficial for large C in the hyper-parameter search presented in section 5. We depict this alternative model on the right side of figure 4 and will refer to it as SQUAD-factorized. We leave further extensions such as Network-in-Network(Lin et al. (2013)) for future work.

4 Related Work

Outside the realm of DLVMs, other methods have been explored for predictive uncertainty. Lakshminarayanan et al. (2017) propose deep ensembles: straightforward averaging of predictions from a small set of separately adversarially trained DNNs. Although highly scalable, this method requires retraining a model up to 10 times, which can be inhibitively expensive for large datasets.

Gal & Ghahramani (2015b) propose the use of dropout (Srivastava et al. (2014)) at test time and present a Bayesian neural network interpretation of this method. A follow-up work by Gal et al. (2017) explores the use of Gumbel-Softmax to smoothly deform the dropout noise to allow optimization of the dropout rate during training. A downside of MC-dropout is the limited flexibility of the fixed bimodal delta-peak distribution imposed on the weights, which requires a large number of samples for good estimates of uncertainty. van den Oord et al. (2017) propose the use of vector quantization in variational inference, quantizing a multi-dimensional embedding, rather than individual latent variables, and explore this in the context of auto-encoders.

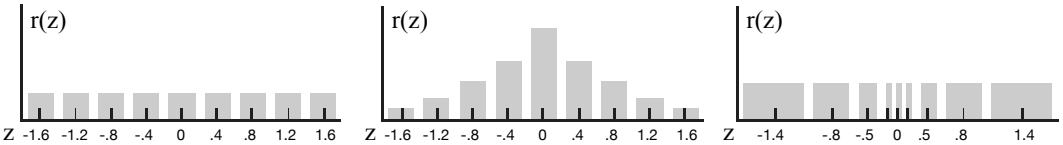

Figure 5: In the IB bound, the marginal $p(z)$ is approximated with a fixed distribution $r(z)$. Using our proposed SQUAD distribution we can we can impose a variety of interesting forms for $r(\mathbf{z})$ via the spacing $\mathbf{v}$ and weighting $r(\mathbf{z} = v_c)$ of the quantization bins. For the values $\mathbf{v}$, we compare linearly spaced bins **(left)** versus bins with equal probability mass under a normal distribution **(right)** . Furthermore, we explore the effect of allowing the bin values to be optimized with SGD on a per neuron or per layer basis, to allow the model to optimize the quantization scheme with the highest fidelity. For the prior probabilities, we explore a uniform prior **(left)** and probability mass of the bins under a normal distribution **(middle)**.

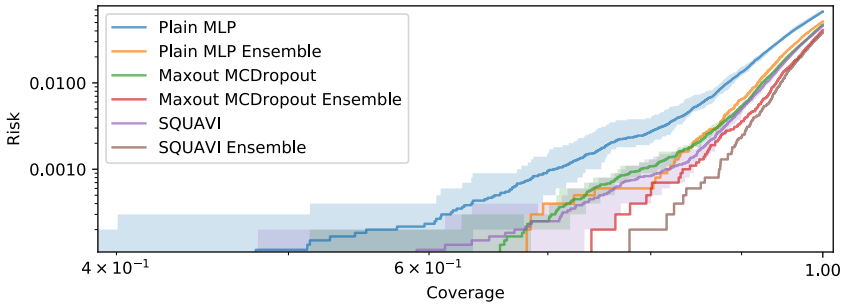

Figure 6: Risk/coverage curve (with log-axes for discernibility) of 2-layer models on notMNIST. Lines closer to the lower-right are better. As the selective classifier lowers the confidence threshold, coverage increase at the cost of greater classification risk. Area around curves represents 90% confidence bounds computed using 10 initializations/splits.

In the space of learning non-linearities, Su et al. (2017) explore a flexible non-linearity that can assume the form of most canonical activations. More flexible distributions have been explored for distributional reinforcement learning by Dabney et al. (2017) using quantile regression, of which can be seen as a special case of SQUAD where the bin values are learned but have fixed uniform probability. Categorical distributions on scalar variables have been used to model more flexible Bayesian neural network posteriors as by Shayer et al. (2017). The use of a mixture of diracs distribution to approximate a variety of distributions was proposed by Schrempf et al. (2006).

5 Results

Quantifying the quality of uncertainty estimates of models remains an open problem. Various methods have been explored in previous works, such as relative entropy Louizos & Welling (2017); Gal & Ghahramani (2015a), probability calibration, and proper scoring rules Lakshminarayanan et al. (2017). Although interesting in their own right, these metrics do not directly measure a good ranking of predictions, nor indicate applicability in high-risk domains. Proper scoring rules are the exception, but a model with good ranking ability does not necessarily exhibit good performance on proper scoring rules: any score that provides relative ordering suffices and does not have to reflect true calibrated probabilities. In fact, well-ranked confidence scores can be re-calibrated (Niculescu-Mizil & Caruana (2005)) after training to improve performance on proper scoring rules and calibration metrics.

In order to evaluate the applicability of the model in high-risk fields such as medicine, we want to quantify how models perform under a desired risk requirement. We propose to use the *selection*

Table 1: SGR coverage results on Fashion MNIST. We present coverage percentage of the test dataset for three pre-determined risk-guarantees (one per column), where higher coverage is better. The results indicate that SQUAD provides competitive uncertainty, especially at low-risk guarantees. Bayesian approximations via deep ensembles improve coverage all over the board, and for SQUAD in particular. *95% confidence bounds shown in parentheses, optimal results in* ***bold***. The 0% coverage for the Gaussian model is a result from too many incorrect predictions having high confidence meaning the 0.995 risk level can not be guaranteed.

Fashion MNIST	cov@risk .5%	cov@risk 1%	cov@risk 2%
Plain MLP	29.1 (08.3, 49.8)	45.9 (41.9, 49.9)	60.4 (57.2, 63.6)
Maxout MCdropout	41.9 (32.0, 51.7)	56.5 (54.2, 58.8)	**69.9** (68.5, 71.4)
Gaussian	00.0 (00.0, 00.0)	33.5 (31.1, 35.9)	47.0 (45.6, 48.5)
SQUAD	**42.9** (35.7, 50.1)	**58.3** (55.3, 61.4)	69.5 (68.0, 71.1)
Deep Ensemble	cov@risk .5%	cov@risk 1%	cov@risk 2%
Plain MLP Ensemble	58.3 (-, -)	70.2 (-, -)	84.6 (-, -)
Maxout MCdropout Ensemble	59.1 (-, -)	72.2 (-, -)	**86.7** (-, -)
Gaussian Ensemble	34.3 (-, -)	47.8 (-, -)	68.4 (-, -)
SQUAD Ensemble	**61.6** (-, -)	**73.1** (-, -)	86.1 (-, -)

with guaranteed risk (SGR) method[1] introduced by Geifman & El-Yaniv (2017) to measure this. In summary, the SGR method defines a selective classifier using a trained neural network and can guarantee a user-selected desired risk with high probability (e.g. 99%), by selectively rejecting data points with a predicted confidence score below an optimal threshold.

To limit the influence of hyper-parameters on the comparison, we use the automated optimization method TPE (Bergstra et al. (2011)) for both baselines and our models. The hyper-parameters are optimized for coverage at 2% risk ($\delta = 0.01$) on fashionMNIST, and sub-sequentially evaluated on notMNIST. Larger models are evaluated on SVHN.

We compare[2] our model against plain MLPs, MCDropout using Maxout activations[3] (Goodfellow et al. (2013); Chang & Chen (2015)) and an information bottleneck model using mean-field Gaussian distributions. We evaluate the complementary deep ensembles technique (Lakshminarayanan et al. (2017)) for all methods.

5.1 Main results

We start our analysis by comparing the predictive uncertainty of 2-layer models with 32 latent variables per layer. In figure 6 we visualize the risk/coverage trade-off achieved using the predicted uncertainty as a selective threshold, and present coverage results in table 1. Overall, we find that SQUAD performs significantly better than plain MLPs and deep Gaussian IB models, and we tentatively attribute this to the increased flexibility of the multinomial distribution. Compared to a Maxout MCdropout model with a similar number of weights, SQUAD appears to have a slight —though not significant— advantage, despite the strong quantization scheme, especially at low risk. Deep ensembles improve results for all methods, which fits the hypothesis that ensembles integrate over a form of weight uncertainty.

[1] We deviate slightly from Geifman & El-Yaniv (2017) in that we use Softmax Response (SR) – the probability taken from the softmax output for the most likely class – as the confidence score for all methods. Geifman & El-Yaniv (2017) instead proposed to use the variance of the probabilities for MCdropout, but our experiments showed that SR paints MCdropout in a more favorable light.

[2] All models are optimized with ADAM (Kingma & Ba (2014)), weight initialization as proposed by He et al. (2015), a weight decay of 10^{-5} and adaptive learning rate decay scheme — 10x reduction after 10 epochs of no validation accuracy improvement— and use early stopping after 20 epochs of no improvement.

[3] We found this baseline to perform stronger in comparison to conventional ReLU MCdropout models, under equal number of latent variables.

Table 2: SQUAD exhibits strong perfromance on a held-out dataset without hyperparameter tuning.

Heldout dataset (notMNIST)	cov@risk .5%	cov@risk 1%	cov@risk 2%
Plain MLP	77.4 (72.2, 82.6)	85.5 (83.6, 87.5)	90.3 (89.4, 91.2)
Maxout MCdropout	85.7 (84.5, 86.8)	90.6 (90.0, 91.2)	94.2 (93.9, 94.6)
SQUAD	**87.1** (85.5, 88.7)	**91.1** (90.2, 92.0)	**94.5** (94.0, 95.0)
Plain MLP Ensemble	85.9 (-, -)	90.6 (-, -)	93.5 (-, -)
Maxout MCdropout Ensemble	88.5 (-, -)	92.8 (-, -)	95.7 (-, -)
SQUAD Ensemble	**90.7** (-, -)	**93.5** (-, -)	**96.1** (-, -)

Table 3: Results on SVHN indicate that the quantization scheme imposed by SQUAD models might hinder performance, but that this is effectively compensated by the SQUAD-factorized variant using a larger amount of bins. Even with $T = 4$ MC samples at test time, SQUAD performs well.

MLP K=256 (SVHN)	cov@risk .5%	cov@risk 1%	cov@risk 2%
Plain MLP	00.0 (00.0, 00.0)	00.0 (00.0, 00.0)	36.3 (33.5, 39.2)
Maxout MCdropout T=100	00.0 (00.0, 00.0)	50.7 (49.2, 52.1)	65.0 (62.6, 67.3)
SQUAD T=100	01.7 (-05.1, 08.4)	42.8 (40.6, 45.1)	59.3 (57.9, 60.7)
SQUAD-factorized T=100	**0.23** (-0.094, 0.526)	**53.2** (49.8, 56.6)	**66.5** (64.0, 68.9)
Maxout MCdropout T=4	00.0 (00.0, 00.0)	38.5 (34.7, 42.4)	**57.6** (55.0, 60.3)
SQUAD T=4	00.0 (00.0, 00.0)	38.0 (36.1, 39.9)	55.6 (54.8, 56.5)
SQUAD-factorized T=4	**03.9** (-11.7, 19.5)	**40.4** (35.7, 45.1)	56.4 (53.2, 59.6)

5.2 Natural Images

To explore larger models trained on natural image datasets, we lightly tune hyper-parameters on 256-latent 2-layer models over 100 TPE evaluations. As SVHN contains natural images in color, we anticipate a need for a higher amount of information per variable. We thus explore the effect of the matrix-factorized variant.

As shown in table 3, SQUAD-factorized outperforms the non-factorized variant. Considering the computational cost at the optimum of a 4-neuron factorization ($B = 4$) with C=37 quantization bins, the model clocks 3.4 million weights. In comparison, the optimum for the presented MCdropout results has C=11, using 9.0 million weights. On an NVIDIA Titan XP, the dropout baseline takes 13s per epoch on average, while SQUAD-factorized spans just 9s.

To evaluate the sample efficiency of the methods, we compare results at $T = 4$ samples. We find that SQUAD's results suffer less from undersampling than MCdropout. We tentatively attribute the sample efficiency to the flexible approximating posterior on the activations, which is in stark contrast to the rigid approximating distribution that MCdropout imposes on the weights. In conclusion, SQUAD comes out favorably in a resource-constrained environment.

5.3 Analysis of latent variable distributions

In order to evaluate if the proposed variational distribution does not simply collapse into single mode predictions, we want to find out what type of distributions the model predicts over the latent variables. We visualize the forms of predicted distributions in figure 7a. While we capture only a small subset of potential multi-modal behavior, this demonstrates that the model indeed utilizes the distribution to its full potential. To provide an intuition on how these predicted distributions emerge, we present a figure 9 in the appendix.

In figure 8 we visualize one of the non-linearities that the method learns for a 1-dimensional input SQUAD-factorized model. The learned non-linearities resemble "peaked" sigmoid activations, which can be interpreted as a combination of an RBF kernel and sigmoid. This provides food for thought on how non-linearities for conventional neural networks can be designed, and the effect of using such a nonlinearity will be studied in further work.

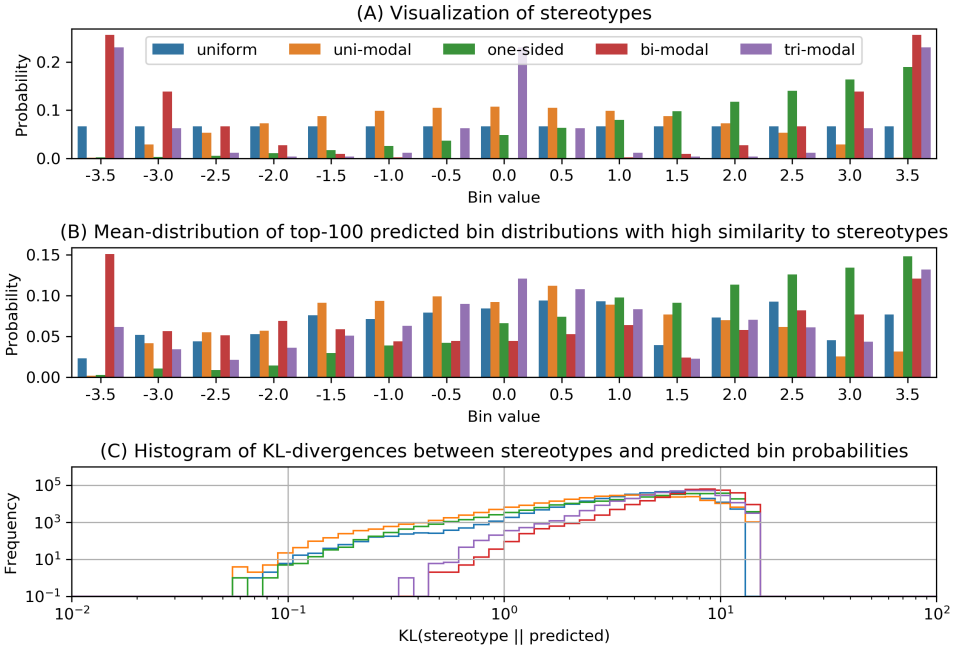

Figure 7: The model uses the flexible variational distribution p(z—x) to its full advantage. Figure (A) visualizes a subset of interesting stereotypical distributions. Figure (B) summarizes distributions predicted by the model similar to stereotypes. Figure (C) shows the frequency of distributions similar to stereotypes (lower KL is closer to stereotypes), demonstrating that these distributions occur frequently.

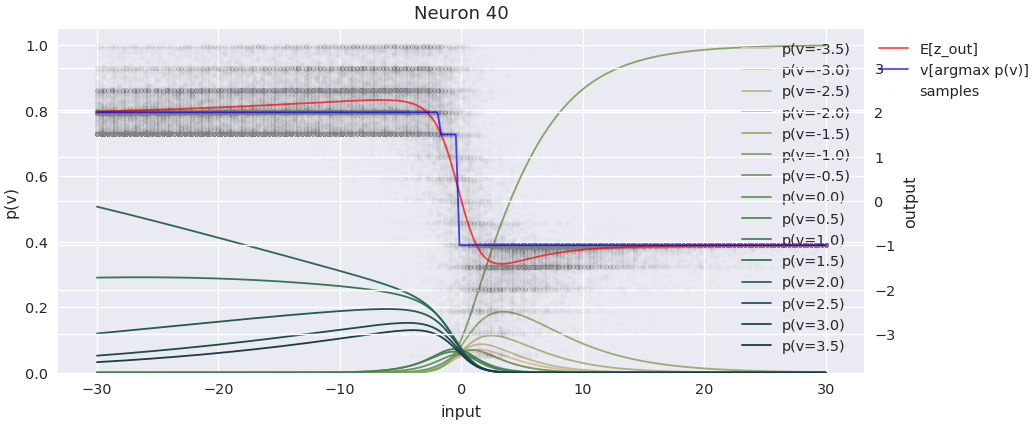

Figure 8: Predicted distribution over the quantization bins of one z as a function of the input, visualizing the stochastic non-linearity learned by the model, for a 1 dimensional input. The model exhibits peaky sigmoid activations. Left y-axis indicates probability for each bin, right y-axis indicates output value. The red line visualizes the expected value, while the blue line shows the value with maximum likelihood. The gray dots represent Gumbel-Softmax samples with $\tau = 0.5$.

6 DISCUSSION

As an interesting side effect, the method breaks the gradient flow normally present in deep neural networks. As the input correlates stronger with the learned weights, the output adjusts accordingly. With SQUAD, stronger correlation with learned weights perturbates the belief of values, rather than the values itself. As a benefit, this can potentially decrease the – arguably underdefined (Santurkar et al. (2018)) – problem of internal covariate shift, as the output domain is constrained. Furthermore, this allows the model to learn the non-linearity. As a downside, this breaks certain assumptions on which best-practice hyperparameter settings have been determined, such as weight initialization and activation functions (He et al. (2015)), and would require further study.

Conclusion In this work, we have proposed a new flexible class of variational distributions and achieve strong performance when applied to a deep variational information bottleneck model. By placing a quantization-based distribution on the activations, we can compute principled uncertainty estimates. We proposed an evaluation scheme motivated by the need in real-world domains to guarantee a minimal risk. The results presented indicate that SQUAD provides an improvement over plain neural networks, Gaussian IB and MCDropout in high-risk domains and that the flexible distribution is used to its full advantage.

REFERENCES

Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. November 2016.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. December 2016.

Alexander A Alemi, Ben Poole, Ian Fischer, Joshua V Dillon, Rif A Saurous, and Kevin Murphy. An Information-Theoretic analysis of deep Latent-Variable models. November 2017.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-Parameter optimization. In J Shawe-Taylor, R S Zemel, P L Bartlett, F Pereira, and K Q Weinberger (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2546–2554. Curran Associates, Inc., 2011.

Jia-Ren Chang and Yong-Sheng Chen. Batch-normalized maxout network in network. November 2015.

Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. October 2017.

Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. June 2015a.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. June 2015b.

Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. May 2017.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4885–4894. Curran Associates, Inc., 2017.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. February 2013.

E J Gumbel. Statistical theory of extreme values and some practical, applications, national bureau of standards, washington (1954). *MR0061342 (15: 811b)*, 1954.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034. cv-foundation.org, 2015.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. November 2016.

J L Holi and J N Hwang. Finite precision error analysis of neural network hardware implementations. *IEEE Trans. Comput.*, 42(3):281–290, March 1993.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. September 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456. jmlr.org, June 2015.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. November 2016.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, November 1999.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. December 2014.

Diederik P Kingma and Max Welling. Auto-Encoding variational bayes. December 2013.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6405–6416. Curran Associates, Inc., 2017.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. December 2013.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. March 2017.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. November 2016.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pp. 625–632, New York, NY, USA, 2005. ACM.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. January 2014.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? (no, it is not about internal covariate shift). May 2018.

O C Schrempf, D Brunn, and U D Hanebeck. Dirac mixture density approximation based on minimization of the weighted cramer-von mises distance. In *2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 512–517, 2006.

Oran Shayer, Dan Levi, and Ethan Fetaya. Learning discrete weights using the local reparameterization trick. October 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15: 1929–1958, 2014.

Qinliang Su, Xuejun Liao, and Lawrence Carin. A probabilistic framework for nonlinearities in stochastic neural networks. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4486–4495. Curran Associates, Inc., 2017.

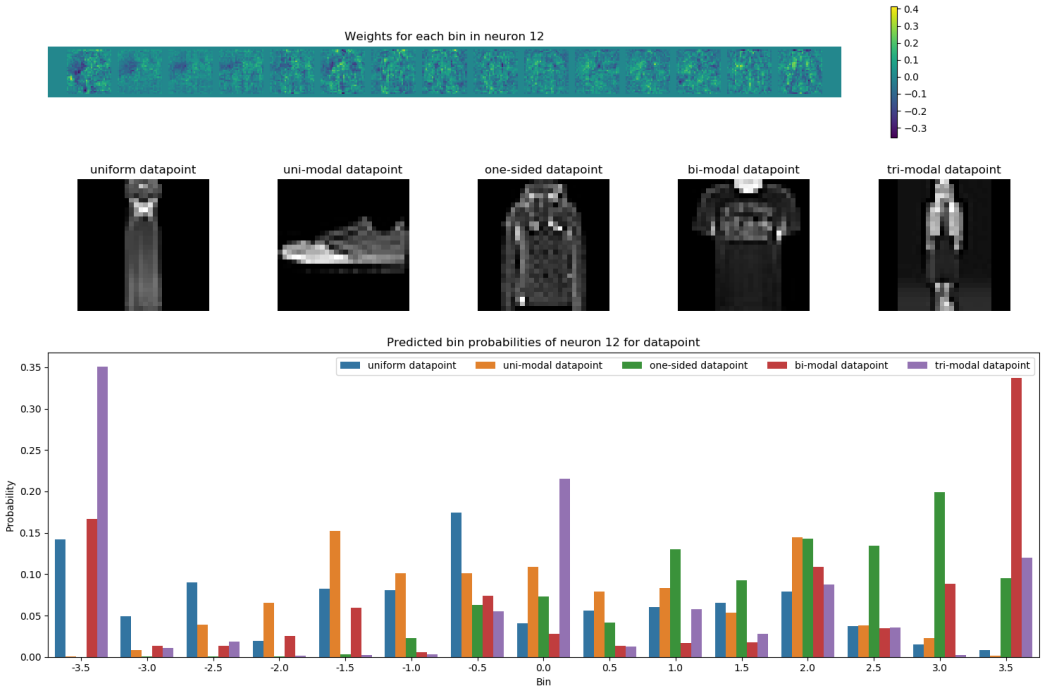

Figure 9: This figure serves to provide intuition on how a variety of distributions come about in our model. We show the set of weights used to predict the probability for the C bins of a randomly selected latent variable $\mathbf{z}_{l=1,k=12}$ from the first layer in a converged 2-layer SQUAD model (reshaped to a 28x28 squares for comparison with the data). We then present 5 data-points for which the neuron predicts a stereotypical distribution, as visualized in the last bar-plot.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. April 2000.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. November 2017.

Dong Yang, Daguang Xu, S Kevin Zhou, Bogdan Georgescu, Mingqing Chen, Sasa Grbic, Dimitris Metaxas, and Dorin Comaniciu. Automatic liver segmentation using an adversarial Image-to-Image network. July 2017.

7 Appendix

7.1 Effect of hyper-parameters on coverage:

The optimal configuration of hyper-parameters and bin priors have been determined using 700 evaluations selected using TPE. The space of parameters explored is as follows, presented in the hyperopt API for transparency:

```
# Shared
C: quniform(2, 10, 1) * 2 + 1,
dropout rate: uniform(0.01, .95),
lr: loguniform(log(0.0001), log(0.01)),
batch_size: qloguniform(log(32), log(512), 1)
# SQUAD & Gaussian
kl_multiplier: loguniform(log(1e-6), log(0.01)),
```

```
init_scale: loguniform(log(1e-3), log(20)),
# SQUAD
use_bin_probs: choice(['uni', 'gaus']),
use_bins: choice(['equal_prob_gaus',
                  'linearly_spaced']),
learn_bin_values: choice([
   'per_neuron', 'per_layer', 'fixed']),
```

In figure 10 we visualize the pairwise effect of these hyper-parameters on the coverage. The optimal configuration found in for the main SQUAD model are: batch size: 244, KL multiplier: 0.0027, learn bin values: per layer, $p(z)$: uniform, $\mathbf{v}$: linearly spread over (-3.5,3.5), lr: 0.0008, C: 15, initialization scale: 3.214.

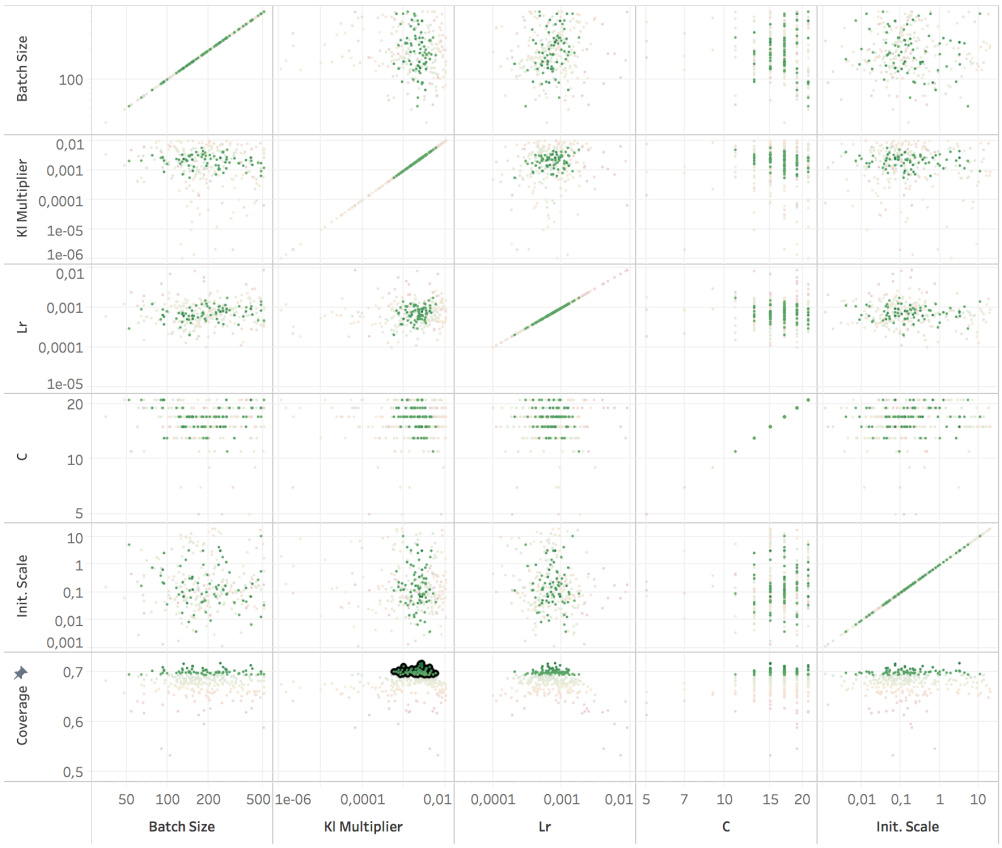

Figure 10: This figure visualizes the pairwise relationship between hyper-parameters of SQUAD and the effect on coverage. The top-60 configurations are highlighted. Green values are good, red values are bad. We have filtered on the optimal settings for bin values and prior to reduce clutter.