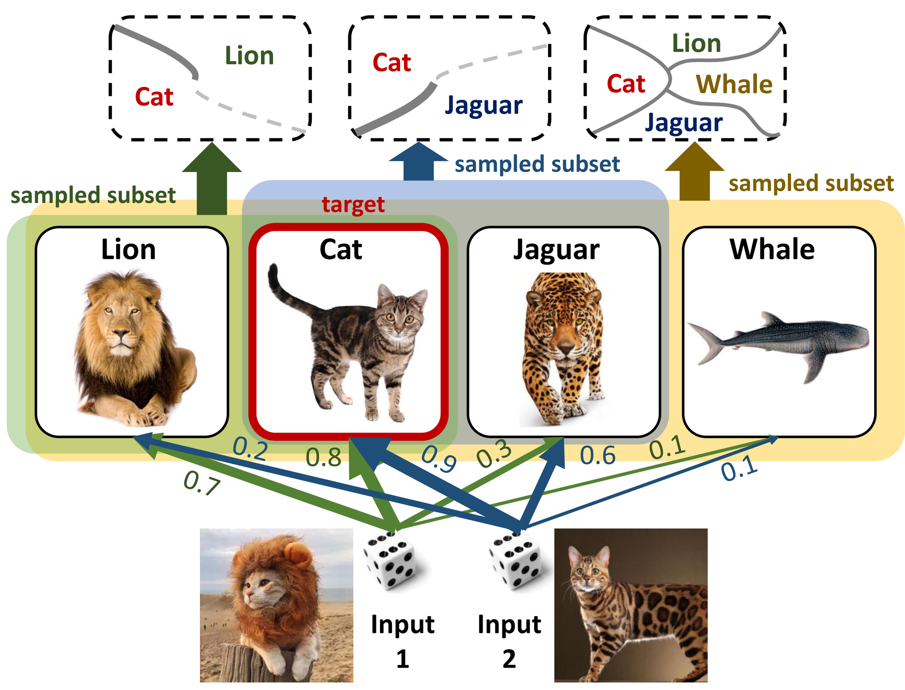
DropMax: Adaptive Variational Softmax

Hae Beom Lee [1 2] Juho Lee [3 2], Saehoon Kim [2], Eunho Yang [1 2], and Sung Ju Hwang [1 2]
KAIST [1], AItrics [2], University of Oxford [3]

Motivation

What if we apply **dropout** to **softmax** function?



- **Ensemble learning** with exponentially many different **sub-classification** problems.
- Stochastic classifier can consider **confusing classes** more often than others in **input-adaptive** manner.

The goal of this work is to figure out if such way of **noise injection** can improve the **generalization performance** of the classifier.

Approach

Softmax function:

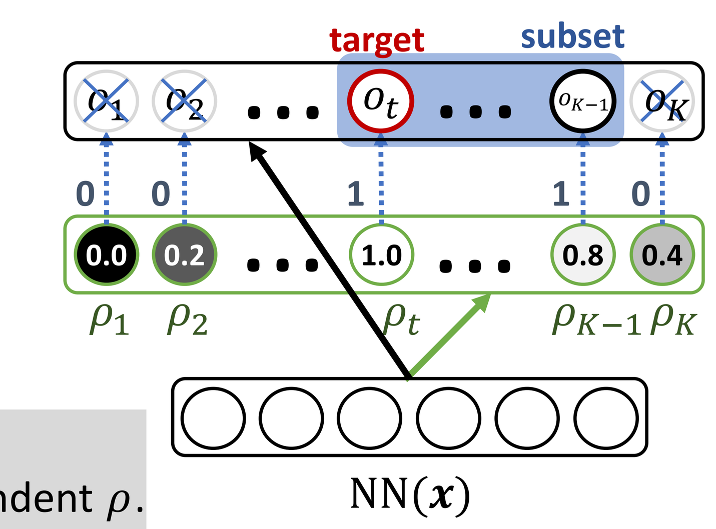
$$p(y_t=1|\mathbf{x}) = \frac{\exp(o_t(\mathbf{x};\theta))}{\sum_k \exp(o_k(\mathbf{x};\theta))}$$

Define DropMax:

$$z_k|\mathbf{x} \sim \mathrm{Ber}(z_k;\rho_k(\mathbf{x};\theta))$$

$$p(y_t=1|\mathbf{x},\mathbf{z};\theta) = \frac{(z_t+\varepsilon)\exp(o_t(\mathbf{x};\theta))}{\sum_k (z_k+\varepsilon)\exp(o_k(\mathbf{x};\theta))}$$

$z_k \sim \mathrm{Ber}(\rho_k(\boldsymbol{x}))$, $\quad \frac{z_t \exp(o_t(\boldsymbol{x}))}{\sum_k z_k \exp(o_k(\boldsymbol{x}))}$



- Class k is completely excluded when $z_k=0$.
- Target class t should not be dropped → input dependent ρ.
- The method can be applied to any neural network.

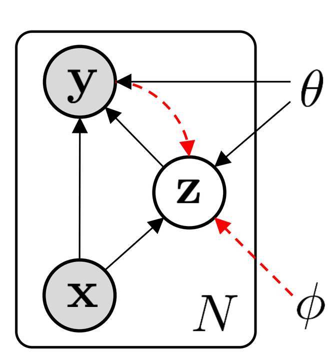
Related to Adaptive dropout **[1]**, but our focus is on the output layer and we train in more principled way using reparameterization trick with concrete relaxation:

$$z_k = \mathrm{sgm}\left\{\frac{1}{\tau}\left(\log\frac{\rho_k(\mathbf{x};\theta)}{1-\rho_k(\mathbf{x};\theta)} + \log\frac{u}{1-u}\right)\right\}, \quad u \sim \mathcal{U}(0,1)$$

Standard Variational Inference

We maximize the evidence lower bound (ELBO).

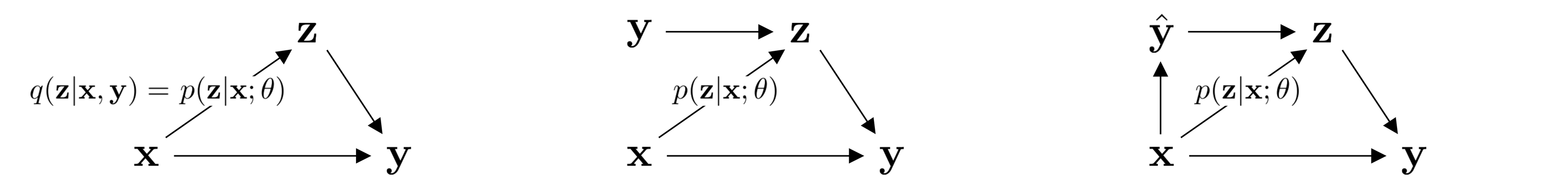
$$\log p(\mathbf{Y}|\mathbf{X};\theta)$$
$$= \log \int p(\mathbf{Y},\mathbf{Z}|\mathbf{X};\theta)d\mathbf{Z}$$
$$\geq \int q(\mathbf{Z}|\mathbf{X},\mathbf{Y};\phi)\log\frac{p(\mathbf{Y}|\mathbf{X},\mathbf{Z};\theta)p(\mathbf{Z}|\mathbf{X};\theta)}{q(\mathbf{Z}|\mathbf{X},\mathbf{Y};\phi)}d\mathbf{Z}$$
$$= \sum_{i=1}^{N}\left\{\mathbb{E}_{q(\mathbf{z}_i|\mathbf{x}_i,\mathbf{y}_i;\phi)}\left[\log p(\mathbf{y}_i|\mathbf{z}_i,\mathbf{x}_i;\theta)\right] - \mathrm{KL}\left[q(\mathbf{z}_i|\mathbf{x}_i,\mathbf{y}_i;\phi)\|p(\mathbf{z}_i|\mathbf{x}_i;\theta)\right]\right\}$$



Graphical model

Approximate Posterior

In modeling the approximate posterior $q(\mathbf{z}|\mathbf{x},\mathbf{y};\phi)$, how to utilize the label $\mathbf{y}$ is not a straightforward matter.



Ignoring label	Simple concatenation	Recurrent connection **[2]**
→ Inaccurate posterior	→ Label is ignored	→ Too much computational cost

Structural form of true posterior

- The relationship between $\mathbf{z}$ and $\mathbf{y}$ is relatively simple in DropMax.
- We encode this property into the approximate posterior.

True posterior is decomposed as :

$$p(\mathbf{z}|\mathbf{x},\mathbf{y}) = \underbrace{p(\mathbf{z}|\mathbf{x})}_{\mathcal{A}} \times \underbrace{p(\mathbf{y}|\mathbf{z},\mathbf{x})/p(\mathbf{y}|\mathbf{x})}_{\mathcal{B}}$$

without label — with label

$$\mathbf{g}(\mathbf{x};\phi) = \mathrm{sgm}(\overline{\mathbf{W}}_\theta^\top\mathbf{h} + \bar{\mathbf{b}}_\theta + \mathbf{r}(\mathbf{x};\phi)), \quad \mathbf{r}(\mathbf{x};\phi) = \mathbf{W}_\phi^\top\mathbf{h} + \mathbf{b}_\phi$$

Encoding label information

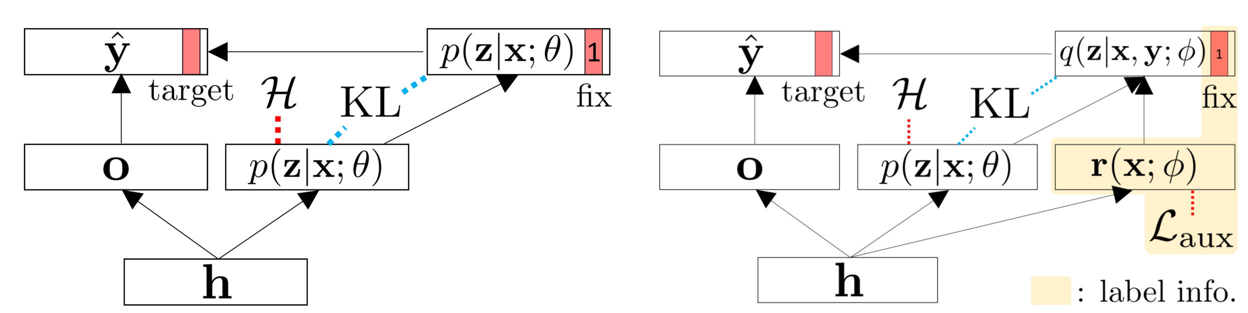
- **[Obs 1]** $\mathbf{z}_t$ is positively and $\mathbf{z}_{k\neq t}$ is negatively correlated with $\mathbf{y}$.
- **[Obs 2]** With mean field approx., $p(z_t=1|\mathbf{x},\mathbf{y})=1$, excluding $z_1=\cdots=z_K=0$.

From [Obs 1], we simply regress $\mathrm{sgm}(\mathbf{r})$ to $\mathbf{y}$:

$$\mathcal{L}_{\mathrm{aux}}(\phi) = -\sum_i\sum_k\left\{y_{i,k}\log\mathrm{sgm}(r_k(\mathbf{x}_i;\phi)) + (1-y_{i,k})\log(1-\mathrm{sgm}(r_k(\mathbf{x}_i;\phi)))\right\}$$

From [Obs 2], we have

$$q(\mathbf{z}|\mathbf{x},\mathbf{y};\phi) = \mathrm{Ber}(z_t;1)\prod_{k\neq t}\mathrm{Ber}(z_k;g_k(\mathbf{x};\phi))$$



DropMax (q=p) (Train) — **DropMax (Train)** — **Test**

Regularized Variational Inference

$p(\mathbf{z}|\mathbf{x};\theta)$ collapses into $q(\mathbf{z}|\mathbf{x},\mathbf{y};\phi)$ too easily, as $p(\mathbf{z}|\mathbf{x};\theta)$ is parametric with input $\mathbf{x}$. We add entropy regularizer to $p(\mathbf{z}|\mathbf{x};\theta)$.

$$\mathcal{H}(p(\mathbf{z}|\mathbf{x};\theta)) = \sum_k \rho_k\log\rho_k + (1-\rho_k)\log(1-\rho_k)$$

The KL divergence and the final objective is

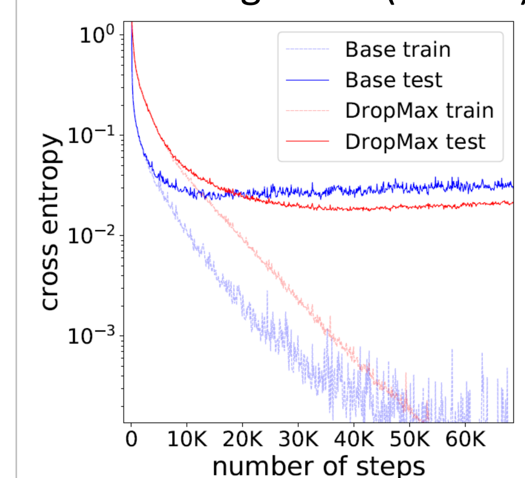
$$\mathrm{KL}[q(\mathbf{z}|\mathbf{x},\mathbf{y};\phi)\|p(\mathbf{z}|\mathbf{x};\theta)] = \log\frac{1}{\rho_t} + \sum_{k\neq t} g_k\log\frac{g_k}{\rho_k} + (1-g_k)\log\frac{1-g_k}{1-\rho_k}$$

$$\mathcal{L}(\theta,\phi) = \sum_{i=1}^{N}\left\{\frac{1}{S}\sum_{s=1}^{S} -\log p(\mathbf{y}_i|\mathbf{x}_i,\mathbf{z}_i^{(s)};\theta) + \mathrm{KL}[q(\mathbf{z}|\mathbf{x},\mathbf{y};\phi)\|p(\mathbf{z}|\mathbf{x};\theta)] - \mathcal{H}\right\} + \mathcal{L}_{\mathrm{aux}}$$

Experiments

1. Generalization Performance

Learning curve (MNIST)



Classification error (%)

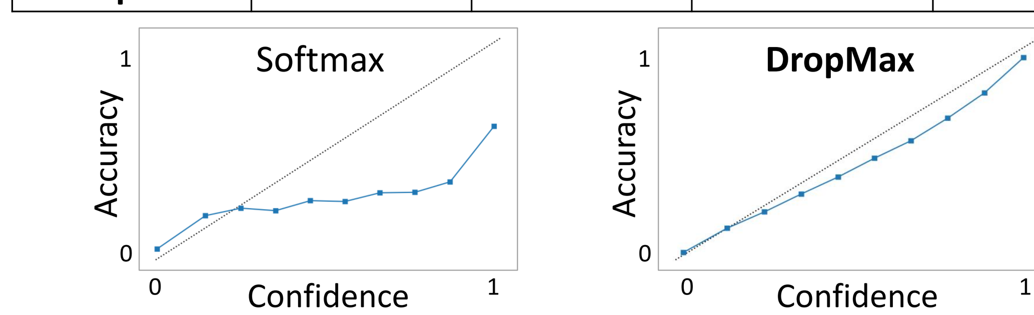
Method	MNIST-1K	CIFAR-100	AWA	CUB
Softmax	7.09	30.60	30.29	48.84
Sparsemax **[3]**	6.57	31.41	36.06	64.41
Sampled Softmax **[4]**	7.36	30.87	29.81	49.90
Class Dropout	7.19	30.78	31.11	48.87
Deterministic Attention	6.91	30.60	30.98	49.97
DropMax (q=p)	7.52	29.98	29.27	42.08
DropMax	**5.32**	**29.87**	**26.91**	**41.07**

- Class Dropout : Randomly drops out non-target classes with a predefined probability.
- Deterministic (sigmoid) Attentions : are multiplied to the softmax exponentiations.
- DropMax (q=p) : A variant of DropMax where we let $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y}) = p(\boldsymbol{z}|\boldsymbol{x};\theta)$.

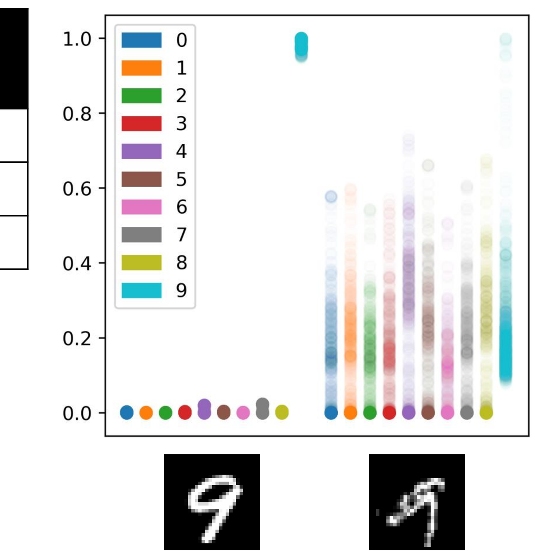
2. Reliability / Calibration

Classification error / Expected calibration error (ECE)

	MNIST-1K Random Background		MNIST-1K Background & Rotation	
	Error (%)	**ECE (%)**	**Error (%)**	**ECE (%)**
Softmax	12.89	1.52	52.06	7.69
DropMax	**10.86**	**0.683**	**48.81**	**2.38**

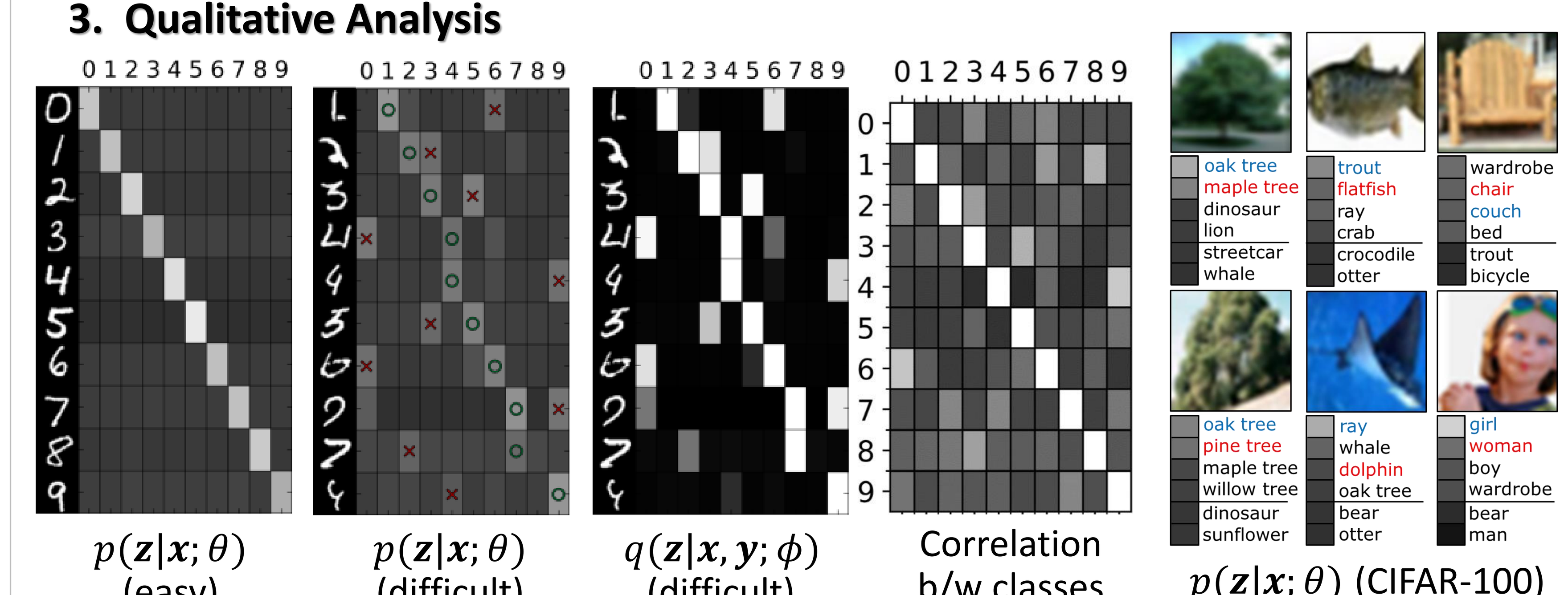


Predictive distribution



- DropMax improves calibration of output probabilities.

3. Qualitative Analysis



$p(\boldsymbol{z}|\boldsymbol{x};\theta)$ (easy) — $p(\boldsymbol{z}|\boldsymbol{x};\theta)$ (difficult) — $q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y};\phi)$ (difficult) — Correlation b/w classes — $p(\boldsymbol{z}|\boldsymbol{x};\theta)$ (CIFAR-100)

- When easy, $p(\boldsymbol{z}|\boldsymbol{x};\theta)$ pre-classifies with high confidence.
- When difficult, $p(\boldsymbol{z}|\boldsymbol{x};\theta)$ selects multiple relevant classes for each instance differently.

References

[1] J. Ba and B. Frey. Adaptive dropout for training deep neural networks. In *NIPS*, 2013.
[2] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *NIPS*, 2015.
[3] A. F. T. Martins and R. Fernandez Astudillo. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *ICML*, 2016.
[4] S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On Using Very Large Target Vocabulary for Neural Machine Translation. In ACL, 2015.