

Continuous Indeterminate Probability Neural Network

Anonymous ICCV submission

Paper ID 4297

Abstract

*This paper introduces a general model called **CIPNN** – Continuous Indeterminate Probability Neural Network, and this model is based on IPNN, which is used for discrete latent random variables. Currently, posterior of continuous latent variables is regarded as intractable, with the new theory proposed by IPNN this problem can be solved. Our contributions are Four-fold. First, we derive the analytical solution of the posterior calculation of continuous latent random variables and propose a general classification model (CIPNN). Second, we propose a general auto-encoder called CIPAE – Continuous Indeterminate Probability Auto-Encoder, the decoder part is not a neural network and uses a fully probabilistic inference model for the first time. Third, we propose a new method to visualize the latent random variables, we use one of N dimensional latent variables as a decoder to reconstruct the input image, which can work even for classification tasks, in this way, we can see what each latent variable has learned. Fourth, IPNN has shown great classification capability, CIPNN has pushed this classification capability to infinity. Theoretical advantages are reflected in experimental results.*

Although recent breakthroughs demonstrate that neural networks are remarkably adept at natural language processing [29, 12, 22], image processing [15], neural networks are still black-box for human [7], cognitive scientists and neuroscientist have argued that neural networks are limited in their ability to represent variables and data structures [13, 5]. Probabilistic models are mathematical descriptions of various natural and artificial phenomena learned from data, they are useful for understanding such phenomena, for prediction of unknowns in the future, and for various forms of assisted or automated decision making [21].

Deep Latent Variable Models (DLVMs) is a probabilistic model and can refer to the use of neural networks to perform latent variable inference [19]. Currently, the posterior calculation is regarded as intractable [20, 21], and the variational inference method is used for efficient approximate

posterior inference [20, 28, 24].

IPNN – Indeterminate Probability Neural Network [1] proposed a new theory, which is used to derive the analytical solution of the posterior calculation of discrete random variables. However, IPNN need predefine the sample space of each discrete random variable (called ‘split shape’ in IPNN), it is sometimes hard to define a proper sample space for an unknown dataset. For CIPNN, the sample space of each continuous random variable is infinite, this issue will not exit in CIPNN.

The rest of this paper is organized as follows: In Sec. 1, related work of VAE and IPNN is introduced. In Sec. 2, the analytical solution of CIPNN is derived and the regularization method is discussed. In Sec. 3, CIPAE is derived and we propose a new method to visualize each latent variable. In Sec. 4, we discuss the training strategy, and two common training setups are discussed: CIPNN and CIPAE are combined together for better evaluation of classification and auto-encoder tasks. In Sec. 5, CIPNN and CIPAE are evaluated and the latent variables are visualized with our new proposed method. Finally, we put forward some future research ideas and conclude the paper in Sec. 6.¹

1. Related Work

1.1. VAE

Modern machine learning and statistical applications require large scale inference in complex models, the inference model are regarded as intractable and either Markov Chain Monte Carlo (MCMC) [25] or variational Bayesian inference [18] are used as approximate solutions [28]. VAE [20] proposes an estimator of the variational lower bound for efficient approximate inference with continuous latent variables. DARN method is generative auto-encoder capable of learning hierarchies of distributed representations from data, and their method applies to binary latent variables [14]. In concurrent work of VAE, two later independent papers proposed equivalent algorithms [28, 24], which provides an additional perspective on VAE and the latter work applies also the same reparameterization method. Two

¹Source code: see supplementary materials.

methods proposed by VAE are also used to realize our analytical solution: the reparameterization trick for making the model differentiable and the KL divergence term for regularization.

VAEs have been used for many tasks such as image generation [23], anomaly detection [31] and de-noising tasks [17] [6]. The drawback of auto-encoder is its strong tendency to over-fit [27], as it is solely trained to encode and decode with as little loss as possible regardless of how the latent space is organized [32], VAE has been developed as an effective solutions [27, 3], e.g. VAEs has been used in EEG classification tasks to learn robust features [33, 2, 3, 4].

The framework of our CIPAE is almost the same as that of VAE, the only difference is that VAE uses neural network as the approximate solution of decoder, while CIPAE uses probabilistic model as the analytical solution of decoder.

1.2. IPNN

Let $X \in \{x_1, x_2, \dots, x_n\}$ be training samples (x_k is understood as ID of random experiment – select one train sample) and $Y \in \{y_1, y_2, \dots, y_m\}$ consists of m discrete labels (or classes), $P(y_l | x_k) = y_l(k)$ describes the label of sample x_k . For prediction, the posterior of the label for a given new input sample x_t is formulated as $P^Z(y_l | x_t)$, superscript Z stands for the medium – model outputted N -dimensional random variables $Z = (z^1, z^2, \dots, z^N)$, via which we can infer label $y_l, l = 1, 2, \dots, m$.

The analytical solution of the posterior is as bellow [1]:

$$P^Z(y_l | x_t) = \int_Z \left(P(y_l | Z) \cdot \prod_{i=1}^N P(z^i | x_t) \right) \quad (1)$$

Where,

$$P(y_l | Z) = \frac{\sum_{k=1}^n \left(P(y_l | x_k) \cdot \prod_{i=1}^N P(z^i | x_k) \right)}{\sum_{k=1}^n \prod_{i=1}^N P(z^i | x_k)} \quad (2)$$

2. CIPNN

2.1. Continuous Indeterminate Probability

Figure 1 shows CIPNN model architecture, the neural network is used to output the parameter θ of some prior distribution of continuous random variable $z^i, i = 1, 2, \dots, N$. All the random variables together form the N -dimensional joint sample space, marked as $Z = (z^1, z^2, \dots, z^N)$, and the whole joint sample space are fully connected with all labels $Y \in \{y_1, y_2, \dots, y_m\}$ via conditional probability $P(y_l | z^1, z^2, \dots, z^N)$.

For each continuous random variable z^i , the indeterminate probability is formulated as:

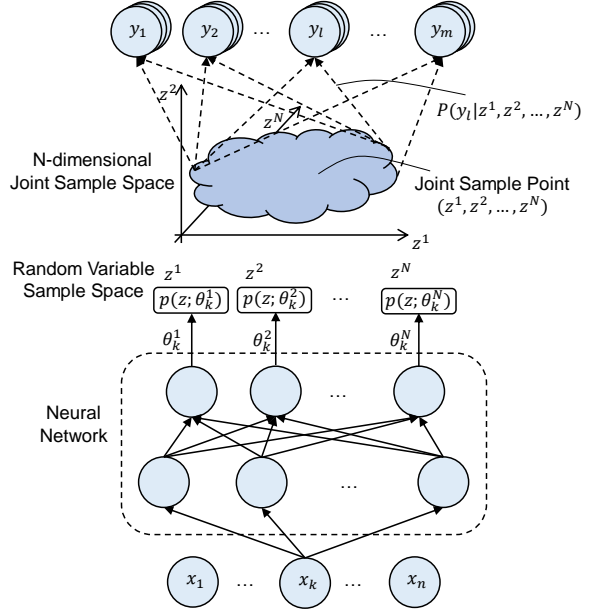


Figure 1: CIPNN – model architecture. Where $P(y_l | z^1, z^2, \dots, z^N)$ is statistically calculated, not model weights.

$$P(z^i | x_k) = p(z; \theta_k^i), i = 1, 2, \dots, N. \quad (3)$$

Where z is generated from some prior distribution with parameter θ_k^i , and $p(z; \theta_k^i)$ is also the density function of $P(z^i | x_k)$.

Substitute $P(y_l | x_k) = y_l(k)$ and Eq. (3) into Eq. (1):

$$\begin{aligned} P^Z(y_l | x_t) &= \int_Z \left(\frac{\sum_{k=1}^n \left(y_l(k) \prod_{i=1}^N p(z; \theta_k^i) \right)}{\sum_{k=1}^n \left(\prod_{i=1}^N p(z; \theta_k^i) \right)} \prod_{i=1}^N p(z; \theta_t^i) \right) \\ &= \mathbb{E}_{z \sim p(z; \theta_t^i)} \left(\frac{\sum_{k=1}^n \left(y_l(k) \cdot \prod_{i=1}^N p(z; \theta_k^i) \right)}{\sum_{k=1}^n \left(\prod_{i=1}^N p(z; \theta_k^i) \right)} \right) \end{aligned} \quad (4)$$

As the integration over Z is complicated, $P^Z(y_l | x_t)$ is rewritten as expectation, we can then use Monte Carlo method [25] to make an approximate estimation. However, a directly sampling z from distribution $p(z; \theta_t^i)$ will make the exception not differentiable. Hence, we use the reparameterization trick [20]: let $\varepsilon \sim p(\varepsilon)$ be some random noise, and define a mapping function $z = g(\varepsilon, \theta)$, so $p(z; \theta_k^i)$ can be rewritten as $p(g(\varepsilon, \theta); \theta_k^i)$.

Therefore, together with Monte Carlo method, the above function can be further formulated as:

$$\begin{aligned}
P^Z(y_l | x_t) &= \mathbb{E}_{\varepsilon \sim p(\varepsilon)} \left(\frac{\sum_{k=1}^n (y_l(k) \prod_i^N p(g(\varepsilon, \theta_t^i); \theta_k^i))}{\sum_{k=1}^n \left(\prod_i^N p(g(\varepsilon, \theta_t^i); \theta_k^i) \right)} \right) \\
&\approx \frac{1}{C} \sum_{c=1}^C \left(\frac{\sum_{k=1}^n (y_l(k) \cdot \prod_i^N p(g(\varepsilon_c, \theta_t^i); \theta_k^i))}{\sum_{k=1}^n \left(\prod_i^N p(g(\varepsilon_c, \theta_t^i); \theta_k^i) \right)} \right)
\end{aligned} \quad (5)$$

Where $\varepsilon_c \sim p(\varepsilon)$.

Take, for example, the univariate Gaussian case: let $P(z^i | x_k) = \mathcal{N}(z; \mu_k^i, \sigma_k^i)$, and $\varepsilon_c \sim \mathcal{N}(0, 1)$, the reparameterization mapping function is $z = \mu + \sigma\varepsilon$, the above function can be rewritten as:

$$\begin{aligned}
P^Z(y_l | x_t) &\approx \frac{1}{C} \sum_{c=1}^C \left(\frac{\sum_{k=1}^n (y_l(k) \cdot \prod_i^N \mathcal{N}(g(\varepsilon_c, \theta_t^i); \theta_k^i))}{\sum_{k=1}^n \left(\prod_i^N \mathcal{N}(g(\varepsilon_c, \theta_t^i); \theta_k^i) \right)} \right)
\end{aligned} \quad (6)$$

Where $\theta_k^i := (\mu_k^i, \sigma_k^i)$, $g(\varepsilon_c, \theta_t^i) = \mu_t^i + \sigma_t^i \cdot \varepsilon_c$ and $\varepsilon_c \sim \mathcal{N}(0, 1)$.

We use cross entropy as main loss function:

$$\mathcal{L}_1 = -\sum_{l=1}^m (y_l(t) \cdot \log P^Z(y_l | x_t)) \quad (7)$$

2.2. Regularization

The sufficient and necessary condition of achieving global minimum is already proved in IPNN [1], which is also valid for continuous latent variables:

Proposition 1 For $P(y_l | x_k) = y_l(k) \in \{0, 1\}$ hard label case, CIPNN converges to global minimum only when $P(y_l | z^1, z^2, \dots, z^N) \rightarrow 1$, for $\prod_i^N p(z; \theta_k^i) > 0$.

In other word, each N -dimensional joint sample area (collection of adjacent joint sample points) corresponds to a unique category. However, a category can correspond to one or more joint sample areas.

According to above proposition, the reduction of training loss will minimize the overlap between conditional joint distribution $\prod_i^N p(z; \theta_k^i)$ of each category. For Gaussian distribution, the variance will be close to zero, and the conditional joint distribution of each category will be far away from each other. This will cause over-fitting problem [32, 27], we have follow up assumption to avoid over-fitting problem:

Assumption 1 For Gaussian distribution, if the distance between the centers of any two adjacent categories' conditional joint distribution $\prod_i^N \mathcal{N}(z; \mu_k^i, \sigma_k^i)$ is equal to e.g. 6σ , then the over-fitting problem can be avoided.

In this way, the 6σ distance will lead to a very small overlap between each category, but the conditional joint distribution of each category will be closely connected with each other. Although the N -dimensional joint sample space is infinite, the effective conditional joint distributions are in a very small space.

VAE uses an additional regularization loss to avoid the over-fitting problem [20, 21], and there are follow up works which has proposed to strengthen this regularization term, such as β -VAE [16, 8], β -TCVAE [9], etc. In order to realize the above assumption, we have a modification of VAE regularization loss:

$$\begin{aligned}
\mathcal{L}_2 &= \sum_{i=1}^N (D_{KL}(\mathcal{N}(z; \mu_k^i, \sigma_k^i) || \mathcal{N}(z; \gamma \cdot \mu_k^i, 1))) \\
&= \frac{1}{2} \sum_{i=1}^N ((1 - \gamma) \cdot \mu_k^i)^2 + (\sigma_k^i)^2 - \log((\sigma_k^i)^2) - 1
\end{aligned} \quad (8)$$

Where N is the dimensionality of Z , regularization factor $\gamma \in [0, 1]$ is a hyperparameter and is used to constrain the conditional joint distribution of each category to be closely connected with each other, impact analysis of regularization factor γ see Figure 8.

The overall loss is:

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 \quad (9)$$

3. CIPAE

For image auto-encoder task, we firstly transform the pixel value to $[0, 1]$ (Bernoulli distribution), and let $Y^j \in \{y_1^j, y_2^j\}_{j=1}^J$, where J is the number of all pixels of one image. $P(y_1^j | x_k) = p_1^j(k) \in [0, 1]$, which describes the pixel value of image x_k at j^{th} position, and $P(y_2^j | x_k) = p_2^j(k) = 1 - p_1^j(k)$.

Substitute $P(y_l^j | x_k)$ into Eq. (6), we will get $P^Z(y_l^j | x_t)$, $l = 1, 2$. In this way, the reconstructed image is formulated as:

$$\text{reconstructed image} := \left\{ P^Z(y_l^j | x_t) \right\}_{j=1}^J \quad (10)$$

In addition, with one (or part) of N dimensional latent variables we can also reconstruct the input image, the reconstructed image is:²

²The details of applying the superscript z^i are discussed in IPNN [1].

$$\text{reconstructed feature} := \left\{ P^{z^i} \left(y_1^j \mid x_t \right) \right\}_{j=1}^J \quad (11)$$

Where $i = 1, 2, \dots, N$. In this way, we can see what each latent variable has learned.

Substitute Eq. (10) into Eq. (7), we can get a binary cross entropy loss:

$$\mathcal{L}_1 = -\frac{1}{J} \sum_{j=1}^J \sum_{l=1}^2 \left(p_l^j(t) \cdot \log P^Z \left(y_l^j \mid x_t \right) \right) \quad (12)$$

And substitute the above loss into Eq. (9), we get the overall loss for auto-encoder training.

4. Training

In this section, we will focus on the training strategy of gaussian distribution.

4.1. Training Strategy

Given an input sample x_t from a mini batch, with a minor modification of Eq. (6):

$$P^Z(y_l \mid x_t) \approx \frac{1}{C} \sum_{c=1}^C \left(\frac{\max(H(\varepsilon_c), \epsilon)}{\max(G(\varepsilon_c), \epsilon)} \right) \quad (13)$$

Where stable number ϵ on the denominator is to avoid dividing zero, ϵ on the numerator is to have an initial value of 1. Besides,

$$H(\varepsilon_c) = \sum_{k=t_0}^{t_1} \left(y_l(k) \prod_i^N \mathcal{N}(\mu_t^i + \sigma_t^i \cdot \varepsilon_c; \mu_k^i, \sigma_k^i) \right) \quad (14)$$

$$G(\varepsilon_c) = \sum_{k=t_0}^{t_1} \left(\prod_i^N \mathcal{N}(\mu_t^i + \sigma_t^i \cdot \varepsilon_c; \mu_k^i, \sigma_k^i) \right) \quad (15)$$

Where $t_0 = \max(0, t_1 - T)$, t_1 is the number of input samples, $\varepsilon_c \sim \mathcal{N}(0, 1)$. Hyperparameter T is for forgetting use, i.e., $P^Z(y_l \mid x_t)$ are calculated from the recent T samples. The detailed algorithm implementation is shown in Algorithm (1).

4.2. Training Setups

By comparing CIPNN and CIPAE, we can see that they can share the same neural network for a training task. As shown in Figure 2, the latent variables of a classification task can be visualized with CIPAE, and we can also use CIPNN to evaluate the performance of an auto-encoder task.

Algorithm 1 CIPNN or CIPAE training

Input: A sample x_t from mini-batch

Parameter: Latent variables dimension N , forget number T , Monte Carlo number C , regularization factor γ , stable number ϵ , learning rate η .

Output: $P^Z(y_l \mid x_t)$

- 1: Declare Θ as a recorder.
- 2: **for** $k = 1, 2, \dots$ **Until Convergence do**
- 3: Use Θ to record: $y_l, \mu_k^i, \sigma_k^i, i = 1, 2, \dots, N$.
- 4: **if** $\text{len}(\Theta) > T$ **then**
- 5: Forget: Reserve recent T elements from Θ
- 6: **end if**
- 7: Compute posterior with Eq. (13): $P^Z(y_l \mid x_t)$
- 8: Compute loss with Eq. (9): $\mathcal{L}(W)$
- 9: Update model parameter: $W = W - \eta \nabla \mathcal{L}(W)$
- 10: **end for**
- 11: **return** model and the posterior

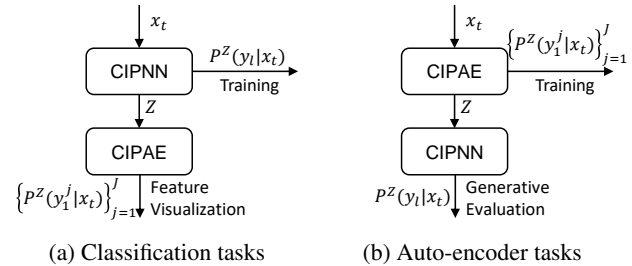


Figure 2: Training setups for classification and auto-encoder tasks. (a) CIPNN is used to do supervised classification tasks and CIPAE is used to reconstruct the input image to see what each latent variable has learned. (b) CIPAE is used to do auto-encoder task and CIPNN is used to evaluate its performance.

5. Experiments and Results

To evaluate the effectiveness of the proposed approach, we conducted experiments on MNIST [11], Fashion-MNIST [30] and Dogs-vs.-Cats-Redux [10] datasets.

Besides, VAE validated that Monte Carlo number C can be set to 1 as long as the batch size is high enough (e.g. 100) [20], we will set batch size to 64, Monte Carlo number $C = 2$ and forget number $T = 3000$ for all the experiments in this paper.

5.1. Results of Classification Tasks

In this section, we use train setup in Figure 2a to perform different classification tasks in order to reconstruct the latent variable to see what they have learned.

The results from the work [1] show that IPNN prefers to put number 1,4,7,9 into one cluster and the rest into another cluster. We also get a similar interesting results in

CIPNN, as shown in Figure 3, with stable number $\epsilon = 1$, the reconstructed image with 1-D latent space shows a strong tendency to sort the categories into a certain order and the number 1,4,7,9 stays together in the latent space. Similar results are also found with 2-D latent space, see Figure 5. Unfortunately, we currently do not know how to evaluate this sort tendency numerically.

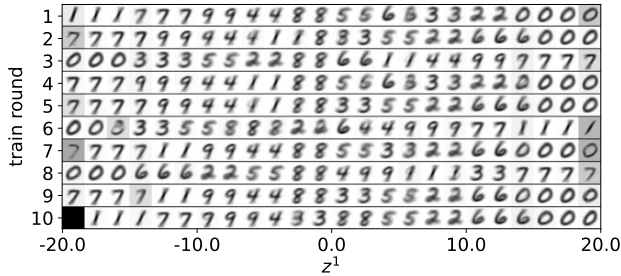


Figure 3: Reconstructed image with 1-D latent space for classification of MNIST: test accuracy is $93.3 \pm 0.5\%$, $\gamma = 0.95$, $\epsilon = 1$. The training is repeated for 10 rounds with different random seeds.

With the visualization method proposed in Eq. (11), we can see what each latent variable has learned in the 10-D latent space. As shown in Figure 4, each latent variable focuses on mainly one or two different categories.

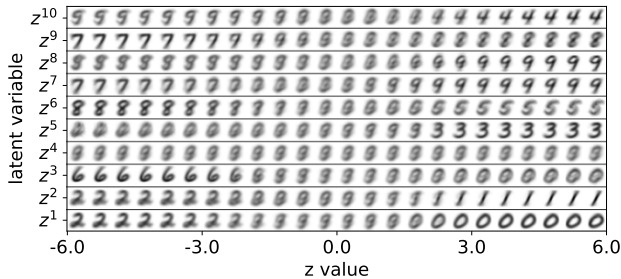


Figure 4: Classification results of 10-D latent space: test accuracy is 97.1% , $\gamma = 0.8$, $\epsilon \approx 0$. Images are reconstructed with one latent variable z^i , $i = 1, 2, \dots, 10$, see Eq. (11).

As shown in Figure 5(a,d,g), with a proper regularization factor γ , the test dataset is mapped to a relative small latent space, and the over-fitting problem is avoided. Besides, in Figure 5(b,e,h) each joint sample area correspond to one unique category, this is consistent with our Proposition 1. In Figure 5(c,f,i), the reconstructed image follows the conditional joint distribution $P(y_l | z^1, z^2)$, $l = 0, 2, \dots, 9$.

5.2. Results of Auto-Encoder Tasks

In this section, we will make a comparison with our CIPAE and VAE [20] model using train setup in Figure 2b, for VAE model, we replace CIPAE part with VAE, in order

to be able evaluate it with CIPNN model. Besides, the regularization loss of VAE is switched to our proposed loss, see Eq. (8). As shown in Figure 6, the results of auto-encoder tasks between CIPAE and VAE are similar, this result further verifies that CIPAE is the analytical solution.

6. Conclusion

General neural networks, such as FCN, Resnet [15], Transformer [29], can be understood as a complex mapping function $f : X \rightarrow Y$ [26], but they are black-box for human [7]. Our proposed model can be understood as two part: $f : X \rightarrow Z$ and $P(Y | Z) : Z \rightarrow Y$, the first part is still black box for us, but the latter part is not unknown anymore. Such kind of framework may have two advantages: the first part can be used to detect the attributes of datasets and summarize the common part of different categories, as shown in Figure 3; the latter part is a probabilistic model, which may be used to build a large Bayesian network for complex reasoning tasks.

Besides, our proposed framework is quite flexible, e.g. from X to Z , we can use multiple neural networks with different structures to extract specific attributes as different random variables z^i , and these random variables will be combined in the statistical phase.

Although our proposed model is derived from indeterminate probability theory, we can see Determinate from the expectation form in Eq. (4). Finally, we'd like to finish our paper with one sentence: The world is determined with all Indeterminate.

A. Visualization

Figure 7 shows classification results of 20-D latent space on Dogs-vs.-Cats-Redux dataset, we can see that each latent variable focuses on both two different categories.

Impact analysis of regularization factor γ is discussed in Figure 8.

References

- [1] Authors. Indeterminate probability neural network. International Conference on Machine Learning 2023 Submission ID 577, Supplied as additional material ipnn.pdf. 1, 2, 3, 4
- [2] David Bethge, Philipp Hallgarten, Tobias Alexander Große-Puppenthal, Mohamed Kari, L. Chuang, Ozan Ozdenizci, and Albrecht Schmidt. Eeg2vec: Learning affective eeg representations via variational autoencoders. 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 3150–3157, 2022. 2
- [3] Luzheng Bi, Jingwei Zhang, and Jinling Lian. Eeg-based adaptive driver-vehicle interface using variational autoencoder and pi-tsvm. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 27(10):2025–2033, 2019. 2

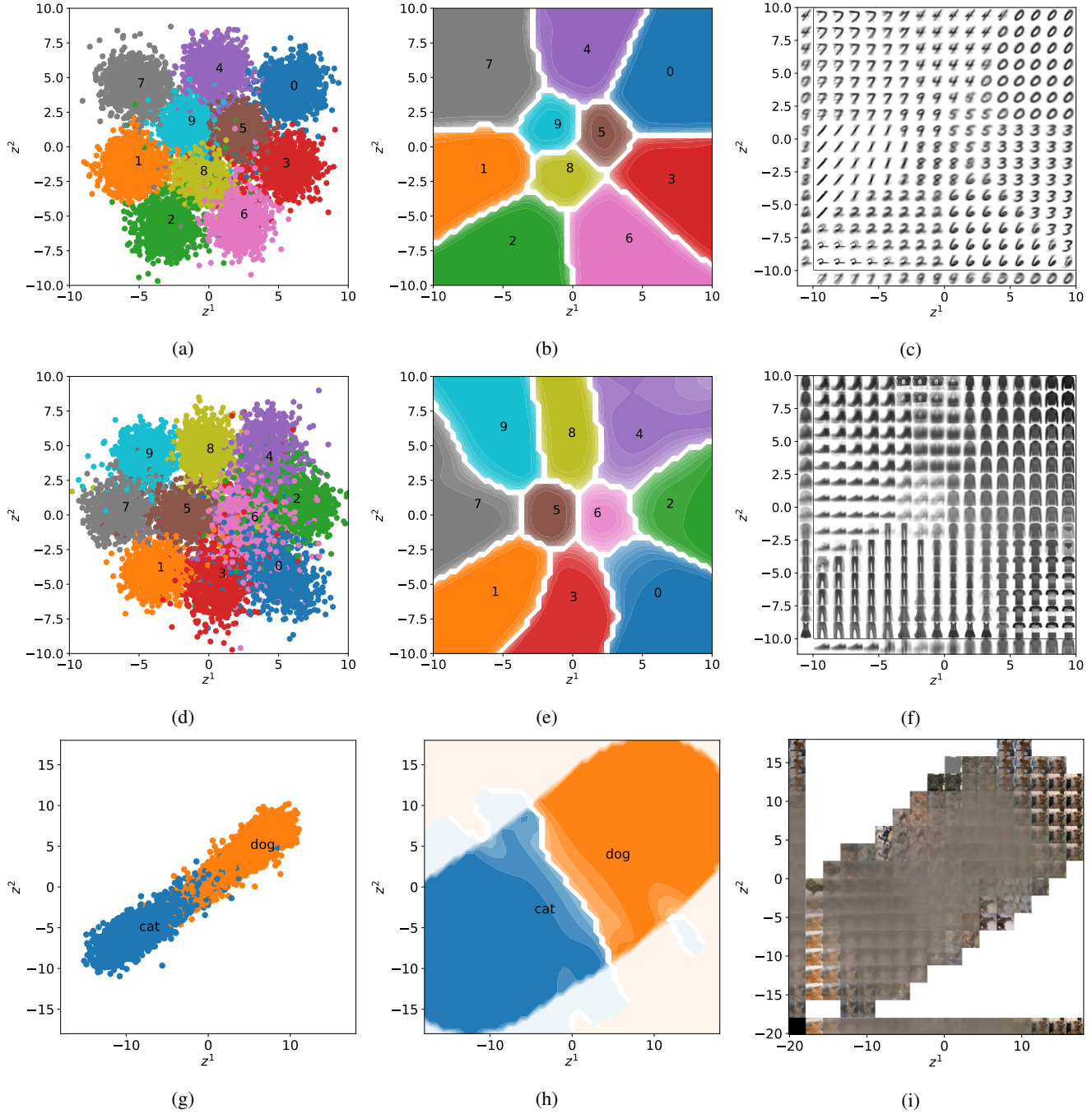


Figure 5: Classification Results of 2-D latent space on MNIST, Fashion-MNIST and Dogs-vs.-Cats-Redux: test accuracy is 96.1%, 87.6% and 95%, respectively. $\gamma = 0.9$ for MNIST and Fashion-MNIST, $\gamma = 0.9999$ for Dogs-vs.-Cats-Redux, $\epsilon \approx 0$. (a,d,g) Results of latent variables on test dataset; (b,e,h) Conditional probability distribution of each category $P(y_l | z^1, z^2)$, $l = 0, 2, \dots, 9$. Colors represent probability value: from 1-dark to 0-light; (c,f,i) Reconstructed image with (z^1, z^2) , see Eq. (10), and image on x and y axes is reconstructed with z^1 and z^2 , respectively, see Eq. (11).

[4] Lies Bollens, Tom Francart, and Hugo Van Hamme. Learning subject-invariant representations from speech-evoked eeg using variational autoencoders. In *ICASSP 2022 - 2022*

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 1256–1260, 2022. 2

[5] Léon Bottou. From machine learning to machine reasoning.

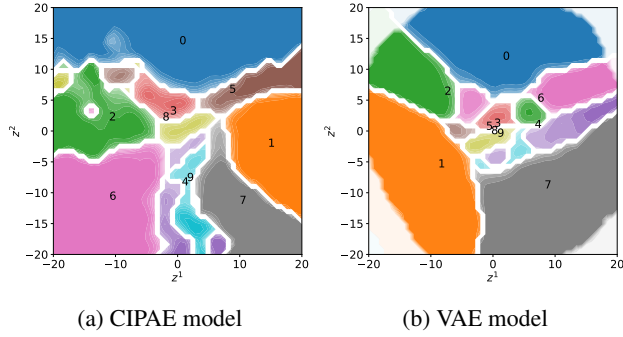


Figure 6: Auto-encoder results of 2-D latent space evaluated with CIPNN model on MNIST: test accuracy is 70.1% for CIPAE and 67.4% for VAE, $\gamma = 0.98, \epsilon \approx 0$. Conditional probability distribution of each category $P(y_l | z^1, z^2), l = 0, 2, \dots, 9$. Colors represent probability value: from 1-dark to 0-light;

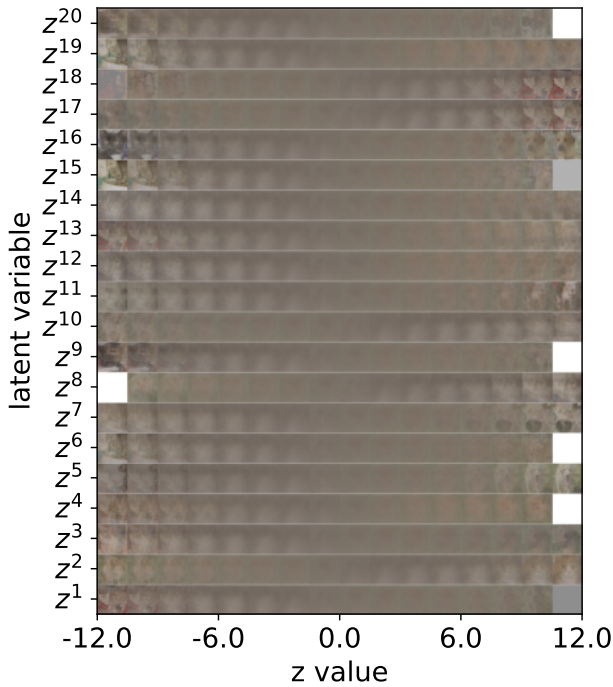


Figure 7: Classification results of 20-D latent space on Dogs-vs.-Cats-Redux: test accuracy is 95.2%, $\gamma = 0.999, \epsilon \approx 0$. Images are reconstructed with one latent variable $z^i, i = 1, 2, \dots, 20$, see Eq. (11).

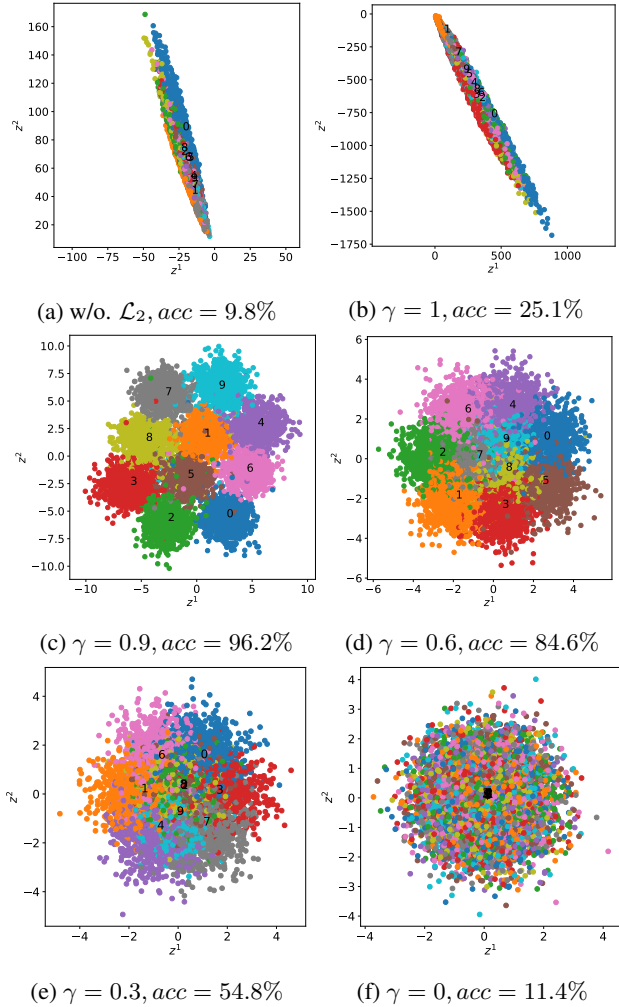


Figure 8: Impact analysis of regularization factor γ with 2-D latent space of classification results on MNIST. (a) Without regularization term, the model has a strong over-fitting problem, and the variance is near to 0, that's why it's space is not big than (b). (b) The variance is around 1, however, without constrain the distribution to be connected together, the space is very high, and it also shows over-fitting problem. (c) With proper γ value, we get an optimal effective latent space. (d,e,f) As γ value further reduces, the latent space is getting smaller, and both tain and test accuracy is reducing due to over regularization.

5

Machine Learning, 94:133–149, 2011. 1

[6] Onur Boyar and Ichiro Takeuchi. Latent reconstruction-aware variational autoencoder. *ArXiv*, abs/2302.02399, 2023. 2

[7] Vanessa Buhrmester, David Münch, and Michael Arens. Analysis of explainers of black box deep neural networks for computer vision: A survey. *ArXiv*, abs/1911.12116, 2019. 1,

[8] Christopher P. Burgess, Irina Higgins, Arka Pal, Loïc Matthey, Nicholas Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β -vae. *ArXiv*, abs/1804.03599, 2018. 3

[9] Ricky T. Q. Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett,

- editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 3
- [10] Will Cukierski. Dogs vs. cats redux: Kernels edition, 2016. 4
- [11] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 4
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019. 1
- [13] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John P. Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476, 2016. 1
- [14] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. *ArXiv*, abs/1310.8499, 2013. 1
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 5
- [16] 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. In *International Conference on Learning Representations*, 2017. 3
- [17] Daniel Jiwoong Im, Sungjin Ahn, Roland Memisevic, and Yoshua Bengio. Denoising criterion for variational auto-encoding framework. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI’17, page 2059–2065. AAAI Press, 2017. 2
- [18] 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, nov 1999. 1
- [19] Yoon Kim, Sam Wiseman, and Alexander M. Rush. A tutorial on deep latent variable models of natural language, 2018. 1
- [20] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014. 1, 2, 3, 4, 5
- [21] Diederik P. Kingma and Max Welling. 2019. 1, 3
- [22] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. 1
- [23] Ali Razavi, Aäron van den Oord, and Oriol Vinyals. *Generating Diverse High-Fidelity Images with VQ-VAE-2*. Curran Associates Inc., Red Hook, NY, USA, 2019. 2
- [24] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML’14*, page II–1278–II–1286. JMLR.org, 2014. 1
- [25] C.P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Verlag, 2004. 1, 2
- [26] Daniel A. Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory*. Cambridge University Press, 2022. <https://deeplearningtheory.com>. 5
- [27] Harald Steck. Autoencoders that don’t overfit towards the identity. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS’20*, Red Hook, NY, USA, 2020. Curran Associates Inc. 2, 3
- [28] Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1971–1979, Beijing, China, 22–24 Jun 2014. PMLR. 1
- [29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. 1, 5
- [30] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. 4
- [31] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, Jie Chen, Zhaogang Wang, and Honglin Qiao. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference, WWW ’18*, page 187–196, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. 2
- [32] Yuan Yue, Jeremiah D. Deng, Dirk De Ridder, Patrick Manning, and Divya Adhia. Variational autoencoder learns better feature representations for eeg-based obesity classification, 2023. 2, 3
- [33] Yuan Yue, Jeremiah D. Deng, Dirk De Ridder, Patrick Manning, and Divya Adhia. Variational autoencoder learns better feature representations for eeg-based obesity classification, 2023. 2