β-CapsNet: learning disentangled representation for CapsNet by information bottleneck

Ming-fei Hu1 · Jian-wei Liu1

Received: 25 November 2021 / Accepted: 12 August 2022 / Published online: 30 August 2022
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

Abstract
We present a framework for learning disentangled representation of CapsNet by information bottleneck constraint that distills information into a compact form and motivates to learn an interpretable capsule. In our β-CapsNet framework, the hyperparameter β is utilized to trade off disentanglement and other tasks, and variational inference is utilized to convert the information bottleneck constraint into a KL divergence term that is approximated as a constraint on the mean of the capsule. For supervised learning, class-independent mask vector is used for understanding the types of variations synthetically irrespective of the image class, and we carry out extensive quantitative and qualitative experiments by tuning the parameter β to figure out the relationship between disentanglement, reconstruction and classification performance. Furthermore, the unsupervised β-CapsNet and the corresponding dynamic routing algorithm are proposed for learning disentangled capsule in an unsupervised manner, and extensive empirical evaluations suggest that our β-CapsNet achieves state-of-the-art disentanglement performance compared to CapsNet and various baselines on several complex datasets both in supervision and unsupervised scenes.

Keywords Disentanglement · Information bottleneck · CapsNet · Representation learning

1 Introduction
The disentangled representation can be specified as ones where single latent units are sensitive to changes in single generative factors while being relatively invariant to changes in other factors [1]. If we could identify and separate these units, such representations distill information into a compact form which is often semantically meaningful and useful for standard downstream tasks such as supervised learning, transfer learning and reinforcement learning [2, 3]. There have been multiple efforts in deep learning toward learning disentangled representations, and β-VAE [4] and InfoGAN [5] are significant methods for disentangling based on variational autoencoder (VAE) [6] and generative adversarial networks (GANs) [7] framework. In addition to the generative networks, however, learning disentangled representations is difficult for some models due to the lack of effective constraints.

In this paper, we propose a framework based on capsule network (CapsNet) [8] and information bottleneck [9, 10] that learns disentangled representation both in supervised and unsupervised learning. As a promising concept, CapsNet provides comparable performance on several benchmark datasets, and it can be regarded as a special autoencoder whose representation is composed of some groups of neurons named capsule. However, it only can learn the entangled capsules that are unfavorable for most learning tasks. Therefore, we intend to leverage the information bottleneck to constrain the capsules for learning disentangled factors.

Information bottleneck constraint is the intractable mutual information between the input and the representation, and we present a variational bound to approximate the mutual information from the perspective of information theory. The variational bound of the mutual information is similar to the constraint of β-VAE, so our method is called...
\(\beta\)-CapsNet. In general, we can assume that the capsule vector in the representation is an isotropic unit Gaussian variable, the variance of the capsule in the representation is related to the dimension, model structure and data type, and therefore, we compel the mean of the capsule to 0.

In summary, we make the following contributions:

1. We introduce \(\beta\)-CapsNet, a novel approach for learning disentangled capsules constrained by information bottleneck, variational inference is used to construct an upper bound of information bottleneck constraint from the perspective of information theory, and the variational bound is tractable for most networks.

2. We proposed class-independent mask vector to replace the existing mask matrix for understanding the types of variations synthetically irrespective of the image class in a supervised manner, and a series of quantitative and qualitative experiments show that our approaches can learn more interpretable representation and grasp the relationship between disentanglement and other tasks by the trade-off parameter \(\beta\).

3. We proposed the unsupervised \(\beta\)-CapsNet and the corresponding dynamic routing algorithm. Empirical evaluations suggest that our unsupervised \(\beta\)-CapsNet achieves state-of-the-art disentanglement performance compared to unsupervised CapsNet and various baselines on several complex datasets.

2 Related work

Disentangled representations Early works to attempt to learn disentangled latent factors include punishing predictability of certain latent dimensions in autoencoder [11] and Boltzmann machine [12]. More recent works have focused on modeling the various factors of the generative models such as InfoGAN and \(\beta\)-VAE. InfoGAN maximizes the mutual information between a small subset of the latent variables and the observation [5], and our \(\beta\)-CapsNet minimizes the mutual information between the input and the representation. \(\beta\)-VAE [4] uses a modified version of the VAE objective with a larger weight \((\beta > 1)\) on the KL divergence, and our \(\beta\)-CapsNet adopts a similar objective with the hyperparameter \(\beta\) to compress the representation space. \(\beta\)-TCVAE carries out a decomposition of the variational lower bound [13] and uses the total correlation or mutual information term [14] to explain the success of \(\beta\)-VAE in learning disentanglement. Factor VAE encourages the code distribution to be factorial by using a discriminator that distinguishes whether the input was drawn from the marginal code distribution or the product of its marginals [15], and it is an ingenious combination of \(\beta\)-VAE and GAN for disentanglement. Joint-VAE learns disentangled jointly continuous and discrete representations for disentangling the factors of different categories on supervised data [16]. Different approaches have been explored for semi-supervised or supervised learning of disentangled representations [17–19]; however, there is no effective method to learn disentangled representations for other models such as CapsNet.

Information bottleneck The definition of information bottleneck is proposed in [9], and using this objective for deep neural networks is pointed out in [10, 20], but it does not include any experimental results. Deep variational information bottleneck constructs the lower bound of the information bottleneck objective in a high-dimensional continuous neural network through variational inference [21], and it has been successfully applied in deep learning for better representations [22–24]. Information dropout injects multiplicative noise in the activations of the neural network to approximate information bottleneck constraints [14], and a similar algorithm is used to limit the primary capsules of CapsNet for better performance and less computation [25]. However, to our best knowledge, learning disentangled representations through information bottleneck constraint is the first attempt.

Capsule network The research history of invariant spatial relationships between the object and its parts can be dated back to [26], the notion of capsule [27] and dynamic routing [8] package this theory into CapsNet, which is regarded as the first theoretical prototype. Each capsule represents an instance of an entity composed of several neurons, and dynamic routing is an iterative mechanism to send lower-level capsules to a higher level. Most works pay attention to novel versions of capsules [28], faster dynamic routing algorithms [29–32] and deeper layers [33, 34]; however, the algorithms for discovering disentangled capsules are the open problems.

In this paper, we find the information bottleneck can help to learn disentangled representation of CapsNet in both supervised and unsupervised manners. To deal with the intractable information bottleneck, we use variational inference to construct a KL divergence term, which can be approximated as a constraint on the mean of the capsule. In addition, we introduce class-independent mask vector for understanding the types of variations synthetically; then, the unsupervised structure of \(\beta\)-CapsNet and dynamic routing algorithm is proposed for learning disentangled capsule.
3 Learning disentangled capsules by information bottleneck

In this section, we would embark on a discussion of learning disentangled capsules by an information bottleneck constraint. Firstly, we recall the conceptions of capsule, dynamic routing and reconstruction network briefly, and we propose the idea of constraining the representation of CapsNet by information bottleneck that encourages the network to learn disentangled factors. Secondly, we introduce the information bottleneck algorithm and discuss how to use it as an additional information loss to constrain the representation of CapsNet, and therefore, a novel framework $\beta$-CapsNet is proposed. We attend to address the intractable mutual information in the loss, and then, variational inference derivation is presented to construct a variational upper bound. We assume that the prior is centered isotropic multivariate Gaussian, and we parameterize the posterior by a factorized Gaussian which mean and variance depend on the CapsNet’s representation, and then, the information loss can be integrated analytically in a simple form. Lastly, class-independent mask vector is proposed for understanding the types of variations synthetically irrespective of the image labels. Our mask vector only sends the correct capsule to the decoder, forcing the network to learn jointly disentangled representation with the same parameters. In addition, our decoder network consisting of deconvolutional layers can capture more spatial relationships from the complex input images and reconstruct clear images.

3.1 A brief conception about CapsNet

A capsule consisting of a group of neurons is the essential unit in the CapsNet that represents the instantiation parameters of a specific type of entity such as an object or an object part [8], and higher-level capsules represent more complex entities with more degrees of freedom. Figure 1 illustrates a brief architecture of CapsNet, the primary capsules provide the lower-level of multi-dimensional entities, the classified capsules which are long instantiation vectors for inputs are the highest-level representations for an object or an object part, these lengths are used to represent the probability, and the one with the longest length is predicted result for the classification task.

The lower-level capsules are sent to higher-level classified capsules by a very different type of computation named dynamic routing. As an iterative routing-by-agreement mechanism on supervised datasets, the routing algorithm assigns capsules depending on whose activity vectors have a big scalar product. Routing mechanism ensures that the classified capsules can predict the input’s class label by length, and the corresponding representation can obtain enough information from features for reconstructions; however, a novel routing is needed for unsupervised learning.

The reconstruction network utilizes the regularization method to alleviate the overfitting and boost the accuracies of some basic classification datasets. During training, the mask matrix is used to mask out all with zeros, but the capsule of the correct label and these new classified capsules are flattened to a vector as the representations of CapsNet. The reconstruction network can reconstruct images from the representations while keeping important details.

Unfortunately, the numerous factors in representations generated by the above approaches are highly complex interactions with others. We guess the reason CapsNet’s entangled representation could be that no suitable constraints are incorporated to restrict the generating process for the representation. In order to retain all the valuable information, CapsNet has no feature compression process.
such as the pooling layer, which causes the learned representation does not identify the salient features underlying objects with significant differentiation. Therefore, imposing appropriate constraints on the generating process for representation without damaging any valuable information underlying objects is the key point to learning disentangled representations of CapsNet.

To discover the independent latent factors of variation and explore the relationship between disentangled representations, reconstructions and classification accuracies of CapsNet, we introduce the information bottleneck constraint, an additional term in loss function to abstract the relevant information of representation and encourage the network to discover disentangled factors.

### 3.2 The loss function of $\beta$-CapsNet

In this subsection, we would focus on formalizing the ideas of learning disentangled capsules by information bottleneck. Given the input data $x$ and its representation $z$ which has some desirable properties in task $y$, information bottleneck suggests that constraining the mutual information between $x$ and $z$ can able to compress representation space and enhance the interpretability of the representations in the perspective of information theoretic concepts, it is equivalent to solve the optimization problem:

$$\min \ I(x; z)$$

s.t. \ $I(x; y) = I(z; y)$  \ (1)

where $I(\cdot; \cdot)$ denotes the mutual information. The constraint in Eq. (1) is that the mutual information between input $x$ and task $y$ is equal to the mutual information between representation $z$ and task $y$, and it means that the desirable properties from the input $x$ should remain in the representation $z$ as much as possible. Therefore, we can substitute $I(x; z)$ for the constraint, and the corresponding Lagrangian dual formulation for Eq. (1) is written as follows in the literature [9]:

$$\mathcal{L}_{IB} = -I(z; y) + \beta I(x; z)$$  \ (2)

where $\beta$ is a positive constant. Most researchers use the second term as a regularization term because the first term can be replaced by the loss function from the original problem. Mutual information is fundamental quantity leverage for measuring the relevant information between two variables:

$$I(x; z) = \int \int p(x, z) \log \frac{p(x, z)}{p(x)p(z)} \, dx \, dz$$  \ (3)

where $p(x, z)$ is the joint probability distribution, $p(x)$ and $p(z)$ are the marginals, and any one of them is difficult to compute for our model, so it is necessary to devise some simple methods to estimate it. Here, we use variational inference to estimate the mutual information while constructing a variational bound formulation. In information theory, mutual information can be seen as the uncertainty in $x$ given $z$:

$$I(x; z) = H(z) - H(z|x)$$

$$= - \int p(z) \log p(z) \, dz + \int \int p(x, z) \log p(z|x) \, dx \, dz$$  \ (4)

where $H(\cdot)$ denotes the Shannon entropy and $H(x|z)$ denotes the conditional entropy. Directly computing the marginal distribution $p(z) = \int p(z|x)p(x) \, dx$ is difficult, so let $q(z)$ be a variational approximation to this marginal:

$$\int p(z) \log p(z) \, dz \approx \text{KL}(p(z)||q(z)) \geq 0$$  \ (5)

Then, we have the following variational upper bound:

$$I(x; z) = \int \int p(x, z) \log \frac{p(z|x)}{q(z)} \, dx \, dz$$

$$\leq \int p(x) \int p(z|x) \log \frac{p(z|x)}{q(z)} \, dz \, dx$$

$$\approx \frac{1}{N} \sum_{n=1}^{N} \text{KL}(p(z|x_n)||q(z))$$  \ (6)

Naturally, we can assume the prior is a normal distribution $p(z) \sim N(0, I)$, where $I$ denotes an identity matrix. The posterior has the form $p(z|x) = N(\mu, \sigma^2)$ where $\mu$ and $\sigma^2$ denote mean and variance, respectively, and these parameters can be constructed from the encoder of CapsNet. Then combining all the facts derived, we have:

$$I(x; z) \approx \frac{1}{N} \sum_{n=1}^{N} \text{KL}(p(z|x_n)||q(z))$$

$$= \int N(\mu, \sigma^2) \log N(\mu, \sigma^2) \, dz - \int N(\mu, \sigma^2) \log N(0, I) \, dz$$

$$= \frac{1}{2} (\mu^2 + \sigma^2 - \log(\sigma^2) - 1)$$  \ (7)

In this case, the KL divergence term controls the degree of disentanglement by twiddling the coefficient $\beta$, and it is similar to the $\beta$-VAE’s loss function that has evaluated the regularization term on unsupervised learning, so we call our model $\beta$-CapsNet. $\beta$-CapsNet limits the relevant features of the representation and forces the representation to
complete the reconstructions with fewer but more general features; therefore, our disentangled representation would remove the complex details and highly entangled latent factors.

Then, we can get the loss function of supervised $\beta$-CapsNet as follows:

$$L = \sum L_k + \alpha L_{\text{mse}} + \beta I(x; z)$$

(8)

where $L_k$ denotes classification loss, and $L_{\text{mse}}$ denotes reconstruction loss. Each capsule in the representation contains several neurons which represent a long instantiation vector if and only if it is occurred in the input, and the candidate capsule with the longest length indicates the predicted result. Therefore, the multi-classification loss $L_k$ for intending to classify capsule $v_k$ depends on the two terms:

$$L_k = T_k \max (0, m^+ - ||v_k||)^2 + \lambda (1 - T_k) \max (0, ||v_k|| - m^-)^2$$

(9)

where $T_k = 1$ if the predicted ground truth class is true, otherwise $T_k = 0$. $m^+$ is set to 0.9 which denotes the lower bound for the correct capsule and $m^-$ is set to 0.1 which denotes the upper bound for other capsules, and their values depend on the certain tasks. $\lambda$ denotes a trade-off parameter used to control the effect of the loss for absent classified capsules, and we set $\lambda = 0.5$ to stop shrinking the lengths of all the classified capsules at the initial training phase of gradient backpropagation.

The goal of the reconstruction network is to reconstruct the input images, and Euclidean distance is used to measure reconstruction loss $L_{\text{mse}}$:

$$L_{\text{mse}} = ||x - \hat{x}||_2^2$$

(10)

where $\hat{x}$ denotes the reconstructions.

### 3.3 Class-independent mask vector for supervised data

The decoder in CapsNet, which is consisted of three fully connected layers, is class dependent. For supervised learning, we assume that the mask matrix $M \in \mathbb{R}^{a \times b}$ denotes the activity matrix of the mask for all classes, where $a$ is the number of classes and $b$ is the capsule dimension. As illustrated in Fig. 2, classified capsules are masked by activity vector $M$ with label $y$ and other capsules are masked by zeros, which results in $M$ as shown in the following:

$$m_i = \begin{cases} p_i & i = y \\ 0 & i \neq y \end{cases}$$

where $i$ is $i$-th class of capsules. After being masked by matrix $M$ that provides class information to the decoder indirectly, the decoder becomes class dependent, and then, the capsules are flattened as a one-dimensional vector and fed into the decoder network.

However, a significant limitation of class-dependent mask is that the latent factors captured by instantiation parameters lack controllability and interpretability. For example, if one disentangled factor for a given class causes some style variable, there is no guarantee that the same dimension would cause the same style in other classes, because capsules in different classes maybe have different positions and parameters. As a result, it is a challenge for learning disentangled latent representations.

Hence, we propose a class-independent mask vector for disentangled capsules and better reconstruction images. Instead of building a mask matrix for all classes, we only send the correct one to the decoder as shown in Fig. 3. Let $M' \in \mathbb{R}^{1 \times b}$ denote the activity vector of mask for class label, the mask vector can force the decoder to learn disentangled representation jointly within a constrained space and same parameters, and the instantiation parameters and interpretable latent factors of all classes are learned from the same distribution and the same dimension of the input vector.

Our mask vector helps us to understand the types of variations synthetically irrespective of the image class. Furthermore, it can also learn more interpretable properties such as a variation from one class to another. In addition, to obtain clearer and sharper reconstructions, we replaced the fully connected layers of the decoder with some deconvolutional layers.

### 4 The specific implementation of $\beta$-CapsNet

In this section, we would describe the specific implementation of $\beta$-CapsNet on supervised learning and unsupervised learning setup, respectively. Firstly, we introduce $\beta$-CapsNet in three parts: encoder, decoder and representation; then, we demonstrate that the variance of the representations is related to the dimension and a new constraint algorithm that limits the mean to 0 is presented to replace the KL term. Secondly, we show how to use $\beta$-CapsNet to learn disentangled representation in a supervised manner, where class-independent mask vector is used to capture more controllable and interpretable latent factors. Lastly, we introduce a novel dynamic routing algorithm for unsupervised learning, and then, we describe the specific implementation of unsupervised $\beta$-CapsNet.
4.1 The structure of encoder, decoder and representation

If the CapsNet is viewed as an autoencoder, the convolutional layers and capsule layers can be regarded as an encoder, the reconstruction network is corresponding to the decoder, and the classified capsule vector after masking operation is the representation. In the following, we will discuss the encoder, the decoder and the representation shown in Fig. 4, respectively.

**Encoder** As shown in Fig. 4a, some convolutional layers are used to convert pixel intensities to the activities of local features from the input without pooling layers. These features named blocks are divided into \( m \) primary capsules, and each primary capsule is an 8D vector. The final layer has one 16D classified capsule per class, and each of them receives input from all primary capsules through dynamic routing. The size of weight matrix \( W \) in routing is \( \left( m, 8, 16 \times n \right) \), where \( n \) denotes class, and this weight can achieve a better initialization of the routing and change the dimensionality from primary capsules to classified capsules.

**Decoder** As shown in Fig. 4b, the existing decoder consists of three fully connected layers that are suited for simple datasets such as MNIST. We replace the decoder with a deconvolutional network which is better at reconstructing spatial relationships and instantiated entities when the input images are complex. We find that the batch normalization will affect disentanglement in CapsNet, and it is different with VAE such as \( \beta \)-TCVAE [13].

**Representation** The construction process of disentangled representation is shown in Fig. 4c. Classified capsules have two roles in supervised learning: The first one is classification; according to the Euclid norm of the classified capsules, we can get the length vectors, and then, we use the vectors to calculate margin loss. The second function is reconstruction, classified capsules masked by class-independent mask vector are our representation, and then, we send the representation to the decoder for reconstruction. In an unsupervised manner, we do not need to consider the impact of the class label, and our mask vector should be removed because there is only one capsule vector in the representation.

In general, we can assume that the output of the encoder is the variance of \( p(z|x) \). To let the length of the classified capsule represent the probability of the entity occurrence in the current input, the dynamic routing mechanism contains a novel nonlinear function named ‘squash’ for lower-level capsules in Eq. (11) and a ‘routing softmax’ for the coupling coefficients between capsules in Eq. (12):

\[
f(v) = \frac{||v||^2}{1 + ||v||^2} \frac{v}{||v||} \quad (11)
\]

\[
c_{ij} = \frac{\exp(b_{ij})}{\sum_j (b_{ij})} \quad (12)
\]

where \( b_{ij} \) denotes the coefficient between capsule \( i \) and capsule \( j \) in higher-level layer. These functions ensure that the longest classified capsule gets shrunk to a length slightly below 1 and other lengths get shrunk to almost zero. However, constraining the variance is an unreasonable assumption for our representation due to squash activation and softmax function, and the variance is related to the dimension and \( \beta \) as shown in Fig. 5. Therefore, it is
necessary to build an alternative constraint that is more suitable for CapsNet.

To tackle the above problem, a new constraint is proposed: We assume that the output of the encoder is the mean of \( p(z|x) \) and to set the mean to approximate equal to 0, the constraint on the variance is abandoned due to its indeterminacy. It is a simpler and more appropriately choice of space compression that only the mean of the capsules needs to be limited.

\[
KL(p(z|x) || q(z)) = \frac{1}{2} (\mu^2 + \sigma^2 - \log(\sigma^2) - 1)
\]

\[
= \frac{1}{2} (\mu^2 - 1)
\]

Now, we can get the final loss function as:

\[
\mathcal{L} = \sum \mathcal{L}_k + \alpha \mathcal{L}_{mse} + \frac{\beta}{2} (v^2 - 1)
\]  

(13)

where \( v \) denotes the representation after masked matrix, \( \alpha \) is positive constant to trade off the reconstruction and other tasks, and we set it to 1 which is always a good balance.

4.2 \( \beta \)-CapsNet in a supervised manner

In supervised learning, the datasets we used are MNIST and Fashion-MNIST, so we assume that the input size of \( \beta \)-CapsNet is \((1, 28, 28)\), and its structure is shown in Fig. 6. In this case, we would follow the encoder settings of CapsNet, except that the number of filters in the second convolutional layer has been adjusted. The first convolutional layer has 256 filters, 9 kernels, 1 stride and ReLU
activation, and the second layer has 128 filters, 9 kernels and a stride of 2. A capsule block (the size is \((8, 6, 6)\)) contains the output of 8 filters, each block has 36 primary capsule vectors, and each of them is an 8D vector, and all the vectors in the same block are sharing their weights.

In total, the primary capsules layer has \(16 \times 6 \times 6\) capsules. The next layer named classified capsule has one 8D capsule per class, and each of these capsules receives input from all the primary capsules, and the size of the weight matrix in dynamic routing is \((576, 8, 80)\). After masking by our mask vector, our representation (an 8D vector) is limited by information bottleneck constraint, and the information loss between representation distribution \(q(z)\) and standard normal distribution \(p(z)\) in Eq. (13) is an additional loss item in the whole loss function.

Five deconvolutional layers are used in our decoder, and the detailed structure of \(\beta\)-CapsNet and CapsNet is exhibited in Table 1. There are two kinds of decoders in our model: A fully connected network is used for simple datasets such as MNIST, and a deconvolutional network is suitable for complex images such as Fashion-MNIST.

There are some issues worth discussing in our setting. First, there are fewer capsule blocks and primary capsules, which can greatly reduce the computational complexity without affecting the reconstruction and learning disentangled representation. Second, the classified capsules are 8D vectors instead of 16D because 8 dimensional is the most suitable setting for reconstruction and disentanglement as shown in Fig. 7. Therefore, our representation is an 8D vector and the dimension of the representation in CapsNet [8] is 160. (Most of them are masked by zeros.)

### 4.3 \(\beta\)-CapsNet for unsupervised data

CapsNet is used for the classification task in supervised learning, we need to modify the model structure and routing algorithm so that the model can handle unsupervised data, and then, we will describe the specific implementation of unsupervised \(\beta\)-CapsNet.

Unsupervised Structure Compared with CapsNet in supervised learning scene, the unsupervised model has two characteristics. Firstly, unsupervised samples have no labels, and the related calculation process can be deleted such as the softmax function and margin loss function. Secondly, we can regard unsupervised samples as supervised data with only one class, and the number of classified capsules is 1 which is the representation of unsupervised data.
CapsNet. Therefore, the mask vector we proposed should be removed because there are no more redundant capsule vectors in the representation and the representation of all samples is the same capsule vector.

**Unsupervised Dynamic Routing** In a supervised manner, dynamic routing in [8] has been proven to work well at classification tasks, and we can utilize it in our supervised $\beta$-CapsNet directly. The main goal of supervised routing is to distinguish capsules and complete the classification task, it assigns the instantiation features of capsule vectors to all capsules of the next layer, and we can use it to pass the features and information in capsule layers. In unsupervised learning, a slight modification is needed for unsupervised data to merge all capsules into the last capsule vector because the last capsule layer contains only one capsule.

We iterate through the proposed routing algorithm $r$ times which is set to 3 empirically following [8]. The inputs of unsupervised routing are capsule vectors $u_i$ in layer $l - 1$ and weight matrix $W$, $u_i$ and $W_i$ denote the $i$-th capsule vector and corresponding weight; after the iterations, the output is capsule vector $v$ in last capsule layer $l$. In the initialized procedure, $\hat{u}_i$ is a prediction vector after spatial mapping and dimensional transformation which is produced by multiplying the $i$-th capsule vector $u_i$ by the weight $W_i$, and $b_i$ and $c_i$ are the log prior probabilities and coupling coefficients between capsule $u_i$ and $v$.

**Algorithm I: Unsupervised dynamic routing algorithm**

1. **Require:** $u_i$, $W_i$
2. **Initialize:** $b_i \leftarrow 0$
3. **for** $r$ iterations **do**
   
   **for** all capsule $i$ in layer $(l-1)$: $c_i \leftarrow \text{soft max}(b_i)$
   
   $v \leftarrow \text{squash}(\sum_i c_i \hat{u}_i)$
   
   **for** all capsule $i$ in layer $(l-1)$ and capsule $v$ in layer $l$: $b_i \leftarrow b_i + \hat{u}_i \cdot v$

   **return** $v$

Figure 8 shows the structure of unsupervised $\beta$-CapsNet which contains two capsule layers and an unsupervised routing algorithm. The largest unsupervised data we used are CelebA, so we set the input size $(3, 64, 64)$ (after resizing) to analyze the specific structure. In this case, four convolutional layers are used to construct primary capsules, and more hidden layers can help to extract more advanced features and reduce the number of primary capsules. There are 576 primary capsule vectors (each vector is an 8D vector) in the first capsule layer and a weight matrix (576, 8, 16) in unsupervised routing. The output of routing is the representation which is a 16D vector, six deconvolutional layers are used to reconstruct samples that can properly capture the spatial relationships from the representations, and the detailed structure of the encoder and the decoder used in the experiments is exhibited in Table 2. We refer to the hyperparameters and settings of convolutional layers and deconvolutional layers of $\beta$-VAE and $\beta$-TCVAE, except for batch normalization layers.

### 5 Experimental results

In this section, we would discuss $\beta$-CapsNet for supervised learning and unsupervised learning, respectively. All the models were implemented using PyTorch and RTX-2070. For the training procedure, we used Adam optimizer with an initial learning rate of 0.001 and all the models are trained for 100 epochs.

| Model       | $\beta$-CapsNet | CapsNet |
|-------------|------------------|---------|
| **Input**   | $(1, 28, 28)$    | $(1, 28, 28)$ |
| **Encoder** | Conv $(256, 9 \times 9, 1)$ | Conv $(256, 9 \times 9, 2)$ |
| Primary capsule | $(576, 8)$ | $(1152, 8)$ |
| Routing matrix | $(576, 8, 80)$ | $(1152, 8, 160)$ |
| Classified capsule | $(10, 8)$ | $(10, 16)$ |
| Mask | mask vector | mask matrix |
| **Representation** | $(1, 8)$ | $(1, 160)$ |
| **Decoder** | Deconv $(256, 4 \times 4, 1)$ | FC (512) |
| Primary capsule | Deconv $(128, 4 \times 4, 2)$ | FC (1024) |
| Routing matrix | Deconv $(64, 9 \times 9, 1)$ | FC (784) |
| Classified capsule | Deconv $(32, 9 \times 9, 1)$ | |
| **Output** | $(1, 28, 28)$ | $(1, 28, 28)$ |
In the first group of experiments, we carry out several experiments to validate information bottleneck loss in a supervised manner, and we would perform a series of quantitative and qualitative experiments, showing the relationship between reconstruction fidelity, classification task and quality of disentanglement by the trade-off parameter $b$. We analyze the classification performance and upper bound on the loss of our proposed $b$-CapsNet and CapsNet with different values of $b$ and datasets. Then, we compare the effects of different $b$ on the reconstructions of the decoder, and we find the configuration for the best performance about the dimension of the representation and the value of $b$. Finally, we train $b$-CapsNet with appropriate hyperparameters on two datasets commonly used to evaluate disentangling performance on supervised learning.

In the second group of experiments, we validate the effectiveness of information bottleneck loss in an unsupervised manner and confirm qualitatively that our model discovers more disentangled factors than CapsNet and $b$-VAE while also being fairly robust to random initialization on unsupervised MNIST and Fashion-MNIST, 3D chairs and CelebA datasets.

5.1 $b$-CapsNet for supervised data

5.1.1 Supervised datasets

(1) MNIST [35]: Modified national institute of standards and technology database is a basic dataset of handwritten digits that is commonly used for computer vision tasks, it contains 60 k training images and 10 k testing images, and each of them is a $28 \times 28$ GY image. There is some interpretable semantic information between images of the same class, and therefore, it is one of the supervised data that is often used to verify disentanglement.
Fashion-MNIST [36]: As a replacement and strengthening benchmarking dataset for the original MNIST, Fashion-MNIST is a dataset of article images consisting of the same amount and size. It also contains several separate factors of variation in the data of the same class, so we adopt it as another supervised dataset.

5.1.2 Disentanglement trade-off

The parameter $\beta$ in our method is used to adjust the information bottleneck loss of the representation, and it can be seen as managing the trade-off among the disentanglement of the representation (measured by information loss), the fidelity of the reconstruction of the input from the representation (reconstruction loss) and classification accuracy (margin loss). In this subsection, we would compare our method with CapsNet baseline [8] on some standard benchmarks using different values of $\beta$.

Classification Performances of Variation $\beta$

Different from the unsupervised learning scene, the input constrains some test samples in supervised learning which are not involved in the training process since there is a difference in classification performances between training and test set. Here, we only need to demonstrate the accuracies of the test set, because the accuracies of the training set are always unconsidered. We set $\beta \in (0, 0.01, 0.1, 1, 2, 3, 4, 5)$, when $\beta = 0$, we get back the original CapsNet.

The classification performances depicted in Fig. 9 confirm our intuition: When training with small values of $\beta$, the network has very little pressure to limit the information of the representation, and the classified capsules have enough information to finish precise classification, so we can expect our model to achieve better performance; on the other hand, increasing the value of $\beta$ makes a more strongly information constraint on the representation; therefore, the model tends to find more disentangled factors.

Table 2 Structure of $\beta$-CapsNet in unsupervised learning

| Unsupervised $\beta$-CapsNet |
|-----------------------------|
| **Input**               | (3, 64, 64) |
| **Encoder**             |               |
| Conv (32, 4 x 4, 2)     |               |
| Conv (64, 4 x 4, 2)     |               |
| Conv (128, 4 x 4, 2)    |               |
| Conv (64, 4 x 4, 1)     |               |
| **Primary capsule**     | (576, 8)     |
| **Routing matrix**      | (576, 8, 16) |
| **Representation**      | (1,16)       |
| **Decoder**             |               |
| Deconv (512, 1 x 1, 1)  |               |
| Deconv (64, 4 x 4, 1)   |               |
| Deconv (64, 4 x 4, 2)   |               |
| Deconv (32,4 x 4, 2)    |               |
| Deconv (32,4 x 4, 2)    |               |
| Deconv (3,4 x 4, 2)     |               |
| **Output**              | (3, 64, 64)  |
rather than other tasks during the training, and we expect the degradation in performance.

Variational Loss of Variation $\beta$ Since the $\beta$-CapsNet’s loss is an upper bounds on the standard loss function, we would like to see the effect of changing the value of $\beta$ on the
training and test set. We train several $\beta$-CapsNet using a group of different values $\beta \in (0, 0.01, 0.1, 1, 3, 5, 8, 10)$, when $\beta = 0$, we get back the original CapsNet. (Its information loss is none.) To paint a clearer picture, we aggregate total loss, information loss, reconstruction loss and margin loss to visualize the effect of the parameter $\beta$ in the training set as shown in Fig. 9a and b, and then, we visualize total loss and reconstruction loss with variation in Fig. 9c and d because the margin loss and information loss in the test set have almost the same curves as the training set.

Margin loss and information loss are in line with our expectations: When training with large values of $\beta$, the network attends to limit the information of the representation, and severe constraint on the representation leads to small information loss; meanwhile, margin loss increases due to small restriction corresponding to the accuracy decline. Although in theory, the increase of $\beta$ will increase reconstruction loss that in turn blurs the reconstructed image, we observe that the reconstructions effect in Fig. 10a, b and c are almost unaffected. Therefore, we would explore the influence of $\beta$ on reconstructed images from Fig. 11 in the next subsection, the experimental results show that the reconstructed samples of $\beta$-CapsNet are almost unaffected by our information bottleneck constraints, and this is remarkable different from $\beta$-VAE whose reconstruction image details will be blurred seriously as the constraints increase.

5.1.3 Qualitative comparisons of disentanglement

In order to qualitatively compare the disentangling performance of $\beta$-CapsNet against CapsNet on the supervised dataset, we train these models on MNIST and Fashion-MNIST. The components of capsule for CapsNet are set within the range of $[-0.2, 0.2]$, the components of the capsule for $\beta$-CapsNet are set within the range of $[-0.08, 0.08]$, and $\beta$ is set to 3. Figure 12 depicts interpretable properties in representation: Both $\beta$-CapsNet and CapsNet have shown to be capable of learning several properties including thickness, width and angle on MNIST, width and length on Fashion-MNIST. However, CapsNet always tends to learn entangled factors; for instance, digit thickness in Fig. 12a is entangled with angle, cloth length in Fig. 12b is entangled with width, and CapsNet can only perceive the width variation in two categories. In contrast, $\beta$-CapsNet learns more disentangled factors that are more interpretable.

5.2 $\beta$-CapsNet for unsupervised data

5.2.1 Unsupervised datasets

(1) Unsupervised MNIST and Fashion-MNIST Handwritten digits and clothes without labels are appropriate unsupervised datasets for learning disentangle factors, and we would like to observe the interpretability of capsule and different disentangled factors from unsupervised learning viewpoint.

(2) 3D Chairs [37]: 3D Chairs is a large dataset of many chair classes used for object category detection in
images as a type of 2D to 3D alignment problem. A chair class can be seen as a running example that contains several continuous interpretable features; therefore, it has become a dominant dataset for disentangled representation evaluation.

3. CelebA [38]: CelebFaces attributes dataset is a dataset for face attributes with more than 10 k identities and 200 k face images, and it can be employed as the training and test sets for many computer vision tasks that is one of the most dominant datasets for learning disentangled representation. As a general preprocessing step, the aligned images are center cropped to $128 \times 128$ and then downsampled to $64 \times 64$, and the center crop can remove background and make it easier for reconstructing.

4. FFHQ [39]: Flickr Faces-HQ is a high-quality dataset of human faces that covers considerably wider variation than CelebA. The dataset consists of 70,000 PNG images at $1024 \times 1024$ resolution,
containing several properties such as age, ethnicity, glasses and hair. Due to the limitation of CapsNet model capacity, the images are center cropped to 200 \times 200 and downsampled to 256 \times 256.

5.2.2 Qualitative comparisons on unsupervised MNIST

The differences in category number of MNIST without labels are the most significant interpretable factor. Therefore, in addition to factors within the same category, disentangled learning should be able to learn to control transformations between categories of similar shapes such as 7–9. Figure 13 provides a qualitative comparison of the disentangling performance of CapsNet and $\beta$-CapsNet. The components of capsule for CapsNet are set within the range of $[-0.3, 0.3]$ or $[-0.6, 0.6]$ ($\beta$ is set to 0), the components of capsule for $\beta$-CapsNet are set within the range of $[-0.15, 0.15]$, and $\beta$ is set to 0.2. It can be seen that both models can automatically identify and learn the disentangle factors such as thickness, width and angle; however, $\beta$-CapsNet can consistently and significantly learn more disentangled latent capsules which is more obvious than the disentangling performance on the supervised dataset in Fig. 9a. For example, when learning about thickness factor, CapsNet without information bottleneck constraint entangles digit width or angle with thickness in almost all digit classes.

Fig. 13 Qualitative comparing results of disentanglement for unsupervised CapsNet and $\beta$-CapsNet

unsupervised CapsNet

unsupervised $\beta$-CapsNet

thickness

width

angle

transformations between two categories

Factor not learnt

transformations between multiple categories
Fig. 14 Disentanglement results of unsupervised $\beta$-CapsNet on 3D chair
Further, although CapsNet performs relatively well on transformations between the part of the digital category, it still struggles to learn a clean factor between multiple categories. By contrast, \( \beta \)-CapsNet can learn a variety of transformations between two categories and multiple categories covering almost all digital categories of similar shapers. This experimental result suggests that a discrete disentanglement quality can be controlled by continuing latent representation of CapsNet, and the information bottleneck constraint leads to better disentanglement both in the same category and in different categories.

5.2.3 Qualitative comparisons on 3D chairs

Manipulating latent variables on 3D chairs is often used for comparing qualitative results of disentangling performance, and Fig. 14 depicts the interpretable properties in reconstructing 3D chairs from the latent representation of \( \beta \)-VAE [4], \( \beta \)-TCVAE [13] and \( \beta \)-CapsNet. However, most properties learned by \( \beta \)-VAE and \( \beta \)-TCVAE are entangled with others; for instance, chair size is entangled with chair category, and backrest is entangled with azimuth and chair category. By contrast, the representation learned by \( \beta \)-CapsNet is disentangled with nuances. \( \beta \)-TCVAE and \( \beta \)-CapsNet are capable of learning an additional property: rotation for swivel chairs; this property is more subtle and likely requires higher mutual information (total correlation mutual information in \( \beta \)-TCVAE). The shortcoming of our model is that azimuth learned by \( \beta \)-CapsNet is not as good as \( \beta \)-TCVAE.

5.2.4 Disentangled capsule on complex datasets

Unsupervised Fashion-MNIST Figure 15 shows that width and length attributes in 4 classes are discovered by unsupervised \( \beta \)-CapsNet, and these attributes are the same as the disentangled latent factors with supervision in Fig. 12b.
Fig. 16 Disentanglement results for unsupervised $\beta$-CapsNet on CelebA
Furthermore, unsupervised \( \beta \)-CapsNet can learn some transformation factors of capsule between two categories (e.g., trouser to pullover) and multiple categories (e.g., sandal to sneaker to coat), and supervised \( \beta \)-CapsNet leads to the entangled capsule in different categories due to the mask vector.

**CelebA** Figure 16 shows that 11 attributes out of 16 dimensions are discovered by the \( \beta \)-CapsNet \((\beta = 1)\) without supervision. \( \beta \)-VAE discovers six disentangled factors only, and some of them are entangled with nuances; \( \beta \)-CapsNet discovers numerous extra factors such as bangs, masculinity and glasses. In addition, it is difficult to render complete face width or skin color for \( \beta \)-VAE and \( \beta \)-TCVAE, whereas the experimental results for \( \beta \)-CapsNet show meaningful disentanglement and extrapolation characteristics. For instance, the extrapolation of face width for \( \beta \)-CapsNet shows that it focuses more on facial lines and contours, whereas the experimental results for \( \beta \)-TCVAE are entangled with many irrelevant factors such as azimuth and gender.

Figure 17 shows 11 attributes and their ranges. We can find an obvious rule from the ranges of these factors: Most factors are only manifested in one-sided of a latent variable, except for three symmetrical attributes including gender, azimuth and hue. For instance, the traversal range of gender from \(-0.04\) to \(0.04\) denotes a conversion from man to woman, traversal of one-sided attributes such as baldness is \(-0.04\) to \(0.04\) and another side learns nothing.

**FFHQ** Figure 18a shows the reconstruction results of \( \beta \)-CapsNet \((\beta = 0.5)\) without supervision. As a special kind of autoencoder, CapsNet has limited model capacity and reconstruction capability compared with GAN, and it is able to reconstruct the basic features of the faces such as profile and color. Figure 18b shows that \( \beta \)-CapsNet discovers six disentangled attributes, and all the traversals maintain the same level of clarity. By contrast, the original CapsNet could not learn any meaningful attributes.

Compared with CelebA, the human faces in FFHQ dataset have fewer samples, fewer attributes and more details. For example, the male faces only have 30 k samples \((100 \text{ k in CelebA})\) with greater otherness. Figure 18c compares several different types of gender attribute traversal, and we can see that the sample learned various gender changes which could not be learned at toy datasets.
Fig. 18 Qualitative results for unsupervised $\beta$-CapsNet on FFHQ dataset.
6 Conclusions and future work

In this paper, we have introduced $\beta$-CapsNet, a novel method for learning disentangled representations of CapsNet through variational information bottleneck. Our $\beta$-CapsNet achieves better disentanglement than CapsNet on the MNIST and Fashion-MNIST datasets in a supervised manner. It also learns more interpretable properties than $\beta$-VAE without supervision on unsupervised MNIST and Fashion-MNIST, 3D chairs and CelebA datasets. We present class-independent mask vector, a refinement of the existing mask matrix that helps to learn disentanglement between different categories within a constrained space on supervised datasets. We also propose unsupervised $\beta$-CapsNet and the corresponding dynamic routing algorithm for unsupervised learning.

Variational bound of information bottleneck term in $\beta$-CapsNet is KL divergence between a standard Gaussian and the distribution of capsule representation. We approximate the KL divergence by limiting the mean of the capsule to approach 0 due to the uncertainty of the variance, which is a very simple regularization technique and is suitable for almost all networks. Therefore, it is an interesting topic to investigate whether it can help other models learn disentangled representations. In addition, the reason why restricting the mean and reducing the space of the representations is beneficial to disentangled representations may be a significant direction. Figuring out the problem
might give us some insight into the nature of disentanglement.

Data availability No new data were created during the study.

Declarations

Conflict of interest We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, and there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled ‘β-CapsNet: Learning Disentangled Representation for CapsNet by Information Bottleneck.’

References

1. Higgins I, Matthey L, Pal A, Burgess C, Glorot X, Botvinick M, Mohamed S, Lerchner A (2017) beta-vae: Learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations
2. Ridgeway K (2016) A survey of inductive biases for factorial representation-learning. arXiv preprint arXiv:1612.05299
3. Lake BM, Ullman TD, Tenenbaum JB, Gershman SJ (2016) Building machines that learn and think like people. Behav Brain Sci 40:1–101
4. Bengio Y, Courville A, Vincent P (2013) Representation learning: a view and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
5. Chen X, Duan Y, Houthooft R, Schulman J, Sutskever I, Abbeel P (2016) InfoGAN: interpretable representation learning by information maximizing generative adversarial nets. In: NIPS
6. Kingma DP, Welling M (2014) Auto-encoding variational bayes. ICLR
7. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: NIPS, Montreal, Canada, pp 2672–2680
8. Sabour S, Frost N, Hinton GE (2017) Dynamic routing between capsules. In: NIPS, Long Beach, CA pp 3856–3866
9. Tishby N, Pereira FC, Biale W (1999) The information bottleneck method. In: The 37th annual Allerton conference on communication, control, and computing, pp 368–377
10. Tishby N, Zaslavsky N (2015) Deep learning and the information bottleneck principle. In: Information theory workshop (ITW), 2015 IEEE, pp 1–5. IEEE
11. Schmidhuber J (1992) Learning factorial codes by predictability minimization. Neural Comput 4(6):863–879
12. Desjardins G, Courville A, Bengio Y (2012) Disentangling factors of variation via generative entangling. arXiv preprint arXiv:1210.5474
13. Chen TQ, Li X, Grosse RB, Duvenaud DK (2018) Isolating sources of disentanglement in variational autoencoders. In NIPS, pp 2615–2625
14. Achille A, Soatto S (2018) Information dropout: learning optimal representations through noisy computation. IEEE Trans Pattern Anal Mach Intell 40(12):2897–2905
15. Kim H, Mnih A (2018) Disentangling by factorising. arXiv preprint arXiv:1802.05983
16. Dupont E (2018) Learning disentangled joint continuous and discrete representations
17. Kulkarni TD, Whitney WF, Kohli P, Tenenbaum J (2015) Deep convolutional inverse graphics network. Adv Neural Inf Process Syst 2539–2547
18. Yang J, Reed SE, Yang M-H, Lee H (2015) Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In: NIPS, pp 1099–1107
19. Reed S, Sohn K, Zhang Y, Lee H (2014) Learning to disentangle factors of variation with manifold interaction. In: ICML, pp 1431–1439
20. Shwartz-Ziv R, Tishby N (2017) Opening the black box of deep neural networks via information. CoRR, arXiv:1703.00810
21. Alemi AA, Fischer I, Dillon JV, Murphy K (2017) Deep variational information bottleneck. In: ICLR
22. Peng XB, Kanazawa A, Toyer S, Abbeel P, Levine S (2019) Variational discriminator bottleneck: improving imitation learning, inverse RL, and GANs by constraining information flow. In: ICLR
23. Chalk M, Marre O, Tkacik G (2016) Relevant sparse codes with variational information bottleneck. In: NIPS, pp 1957–1965
24. Federici M, Dutta A, Forrê P, Kushman N, Akata Z (2020) Learning robust representations via multi-view information bottleneck. In: ICLR
25. Hu MF, Liu JW (2021) Optimal representations of CapsNet by information bottleneck. In: ICANN
26. Hinton GE (1981) A parallel computation that assigns canonical object-based frames of reference. In: Proceedings of the 7th international joint conference on artificial intelligence, IJCAI, pp 683–685
27. Hinton GE, Krizhevsky A, Wang SD (2011) Transforming auto-encoders. In: Artificial neural networks and machine learning, ICANN, pp 44–51
28. Hinton GE, Sabour S, Frost N (2018) Matrix capsules with EM routing. In: International conference on learning representations, ICLR
29. Ribeiro FDS, Leonidis G, Kollias SD (2020) Capsule routing via variational bayes. In: The thirty-fourth AAAI conference on artificial intelligence, AAAI 2020, pp 3749–3756
30. Dou Z, Tu Z, Wang X, Wang L, Shi S, Zhang T (2019) Dynamic layer aggregation for neural machine translation with routing-by-agreement. In: AAAI conference on artificial intelligence, pp 86–93
31. Wang D, Liu Q (2018) An optimization view on dynamic routing between capsules. In: International conference on learning representations, ICLR
32. Li H, Guo X, Dai B, Ouyang W, Wang X (2018) Neural network encapsulation. In: European computer vision conference, ECCV, pp 266–282
33. Kosiorek AR, Sabour S, Teh YW, Hinton GE (2019) Stacked capsule autoencoders. In: Annual conference on neural information processing systems, NeurIPS, pp 15486–15496
34. Rajasegaran J, Jayasundara V, Jayasekara S et al (2019) Deep-caps: going deeper with capsule networks. In: Computer vision and pattern recognition, CVPR, pp 10725–10733
35. LeCun Y, Cortes C, Burges CJC (1998) The mnist database of handwritten digits
36. Xiao H, Rasul K, Vollgraf R (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. CoRR
37. Aubry M, Maturana D, Efros A, Russell B, Sivic J (2014) Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In: CVPR
38. Liu Z, Luo P, Wang X, Tang X (2015) Deep learning face attributes in the wild. In: ICCV
39. Karras T, Laine S, Aila T (2019) A style-based generator architecture for generative adversarial networks. In: CVPR, pp 4401–4410

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.