Entropy Voting Between Capsules

Anonymous authors
Paper under double-blind review

Abstract

Capsule networks offer a promising solution in computer vision by addressing the limitations of convolutional neural networks (CNNs), such as data dependency and viewpoint challenges. Unlike CNNs, capsules reduce the need for data augmentation by enhancing generalization from limited training data. We explore capsules from the perspective of information theory, viewing them as continuous random variables. We use marginal differential entropy to measure the information content of capsules, and relative entropy to model the agreement between lower-level and higher-level capsules. The proposed entropy voting method aims to maximize marginal capsule entropies and to minimize their relative entropy. We show through an ablation study that such a relationship exists between the capsules. We also show that our approach performs better or comparably against state-of-the-art capsule networks while significantly improving inference time. This research highlights the synergy between capsules and information theory, providing insights into their combined potential.

1 Introduction

Capsule networks have emerged as a promising approach in computer vision, with a clear focus on alleviating the shortcomings of convolutional neural networks (CNNs). CNNs struggle with several issues, notably the need for enormous amounts of data, the vulnerability to challenges posed by varying viewpoints, scaling, occlusion, and deformation in images, and the lack of understanding of spatial relationships. To circumvent these limitations, CNNs require an extensive use of data augmentation techniques. Modern alternatives like Vision Transformers (ViT) do not solve this problem since they are even more demanding with respect of the amount of training data (Dosovitskiy et al., 2021). In contrast, capsules are designed to reduce the reliance on data augmentation, aiming to directly mitigate the need for such techniques and enhance the network's ability to generalize from limited training data.

Capsules, first proposed by Hinton et al. (2011) and later refined by Sabour et al. (2017), can be seen as vectors that are designed to capture rich and hierarchical information about objects in an image. Unlike traditional convolutional network units, capsules explicitly encode object instantiation parameters, such as texture and pose, thus enabling them to learn more robust representations without having to use data augmentation to achieve the same effect. The intuition behind capsule networks is to model objects as a composition of capsules, with lower-level capsules encoding parts of an object and higher-level capsules encoding the object as a whole, which are then linked together through a voting procedure that is commonly referred to as agreement between lower- and higher-level capsules.

The different levels of capsules provide a modular design that, through agreement, is able to learn part-whole relationships and spatial hierarchy in the image. Thus, capsules enable the network to utilize both global and local visual information at various levels of abstraction, thereby encoding a more comprehensive representation of the input image. While state-of-the-art methods mainly focus on modelling agreement between the capsules, we investigate capsule networks from the point of view of information theory, exploring how information-theoretic concepts can enhance our understanding and utilization of capsules.

Viewed through the lens of information theory, a capsule can be modeled as a continuous random variable with well-defined mean and variance, since capsules are inherently real-valued vectors. These moments are estimated using the method of moments for the log-normal distribution, allowing us to parameterize the

entropy and divergence calculations directly from data. This allows us to compute the marginal differential entropy of each capsule, which reflects the amount of information it encodes. Higher entropy indicates richer and more descriptive representations. To model the relationship between lower-level (child) and higher-level (parent) capsules, we use relative entropy, also known as the Kullback-Leibler divergence, which quantifies how well a parent capsule aligns with its corresponding children. Our proposed approach, entropy voting, is built on the dual objective of maximizing the marginal entropy of individual capsules and minimizing the relative entropy between lower- and higher-level capsules.

In our experiments, we provide empirical evidence that underscores the effectiveness of these information-theoretic principles in elevating the performance and robustness of capsule networks. This investigation illuminates the synergistic relationship between capsule networks and information theory, shedding valuable insights into their combined potential. We compare our results against three baselines, which are the state-of-the-art of capsule networks.

2 Related works

The concept of capsules was first introduced by Hinton et al. (2011) as a way to encode both descriptor and probability parameters of an object or object part. However, it was the work by Sabour et al. (2017) that formulated the concept of capsule networks as it is now known, including (1) the dynamic routing mechanism, (2) a capsule-specific loss function, (3) a reconstruction regularizer. and (4) a squash activation function that converts the magnitude of each parent capsule to a probability. Since then, there have been many works on capsule networks, namely focusing on improving the underlying routing algorithm, but also some modeling capsules as matrices rather than vectors.

Hinton et al. (2018) proposed to treat capsules as 4×4 pose matrices with an activation probability. The method uses an iterative expectation-maximization algorithm that includes a transformation matrix which learns a mapping between capsules at different levels. In addition, the authors propose a new, matrix-capsule specific loss function.

Choi et al. (2019) investigate a routing mechanism based on attention and propose a modification to the squash activation function. Routing based on self-attention has also been proposed by Mazzia et al. (2021). They, too, introduced a modification to the squash activation function, similar to Choi et al. (2019).

Routing in capsule networks has also been investigated as an optimization problem by Wang & Liu (2018). The authors formulated routing as a clustering problem, and used a Kullback-Leibler divergence term for regularization. In their proposed work, capsules are squashed only after routing to stabilize the growth of capsule under the assumption that only parent capsules should have a probabilistic magnitude.

Zhang et al. (2018) investigate an alternative approach to capsule networks, based on orthogonal representation of capsule subspaces onto which feature vectors (i.e. first-level capsules) are projected. Each capsule subspace is updated until it contains input feature vectors corresponding to the associated class, i.e. parent capsule. Their proposed method does not involve any routing mechanism as capsule subspaces and projection matrices are learned through back-propagation.

Ribeiro et al. (2020) propose a routing method based on variational Bayesian inference, an interpretation of expectation-maximization. In their proposed method capsule activation is done only after routing. Another alternative routing method is proposed by Zhang et al. (2020), which is based on kernel density estimation. The authors proposed two different approaches, one based on expectation-maximization, and the other on mean shift, a feature-space analysis technique. Building on variational methods, De Sousa Ribeiro et al. (2020) investigate a routing mechanism based on variational inference that accounts for uncertainty.

Hahn et al. (2019) investigate routing as a separate network. Each capsule is routed independently by a dedicated sub-network, each of which is a single-layer perceptron (SLP), meaning that there are as many SLPs as there are higher-level capsules. The SLPs have two sets of weights, one for pose and one for routing. Rajasegaran et al. (2019) explore a routing method based on 3D convolutions, and introduce a 3D variant of the squash function.

However, Byerly et al. (2021) show that no routing is needed by using homogeneous vector capsules, which replace matrix multiplication with the Hadamard product, in order to keep capsule dimensions untangled. Their approach also leverages multiple classification branches.

One notable entropy-based approach is by Renzulli et al. (2022), which analyzes and refines part-whole relationships captured by the routing mechanism. However, the proposed method is not end-to-end trainable but, instead, it requires manual pruning of routing coefficients during training and, afterwards, quantizing the routing coupling coefficients into discrete levels and extracting parse trees based on the resulting quantized sequences. These sequences are then used to construct a dictionary of parse trees, from which entropy is computed to assess and enforce structural consistency. Furthermore, the proposed routing method is iterative.

The aforementioned approaches fall short in at least one of two aspects; computationally inefficient routing (be it iterative, attention-based, a sub-network, or other), and capsule activation done using some non-differentiable function (the squash function or a variant thereof, such as vector norm). The first aspect leads to large execution times during training and/or inference, and the second aspect often leads to instability during training. Moreover, in all cases test performance is reported using significant amounts of data augmentation. Keeping in mind that one of the key ideas of capsule networks is to reduce the necessity of data augmentation, this is at least somewhat questionable. An objective evaluation of the performance of capsule networks w.r.t. the issues they are designed to mitigate should not be based on data augmentation. We address all three aspects in our work. Furthermore, we assume that the large execution times (not only for training but especially for inference) are the main reason that capsule networks are still rarely used, despite their theoretical advantages, which is not addressed by any of the aforementioned works.

We selected Sabour et al. (2017), Mazzia et al. (2021), and Byerly et al. (2021) as our baselines due to the fact that they are the state-of-the-art capsule networks on MNIST, which is still the de facto dataset for capsule networks.

3 Motivation

Information theory provides a robust framework for quantifying and analyzing the information content and relationships between variables. As random variables are defined as measurable real-valued functions, we can use differential entropy to measure the information content of such functions based on their moments, thus offering insights into the intrinsic information it carries. Similarly, Kullback-Leibler divergence serves as a measure of distance between two such functions, allowing us to evaluate how closely they align with respect to their underlying distributions.

These concepts are critical for modeling relationships between random variables and can be extended to real-valued vectors, enabling the study of dependencies and information transfer between capsules in the context of capsule networks based on their moments. Thus, applying differential marginal entropy and Kullback-Leibler divergence to functions provides a powerful method to quantify information content, assess similarity, and analyze dependencies and information transfer. This is motivated by the concept of mutual information, which quantifies the shared information between two random variables.

4 Methodology

4.1 Construction of capsules

We model capsules as positive real-valued vectors. Formally, let $\Psi = \{\boldsymbol{\psi}_i\}_{i=1}^{N}$ denote the set of child capsules and $\Omega = \{\boldsymbol{\omega}_j\}_{j=1}^{M}$ the set of parent capsules, where each $\boldsymbol{\psi}_i \in \mathbb{R}_+^d$ and $\boldsymbol{\omega}_j \in \mathbb{R}_+^{d'}$. We then treat each capsule vector as a log-normally distributed random variable, such that $\boldsymbol{\psi}_i \sim \text{LogNormal}(\mu_{\psi_i}, \sigma_{\psi_i}^2)$ and $\boldsymbol{\omega}_j \sim \text{LogNormal}(\mu_{\omega_j}, \sigma_{\omega_j}^2)$, defined via a softplus activation function to ensure positivity and numerical stability. This is motivated by the central limit theorem, under which the internal activation of artificial neurons can often be regarded as approximately Gaussian. When passed through an exponential function e^x, these Gaussian-distributed variables result in a log-normal distribution, which aligns well with the desired

positive support of capsule outputs. Furthermore, this formulation allows us to quantify both the marginal entropies of individual capsules and the agreement between lower-level (child) and higher-level (parent) capsules using principles from information theory.

$$\zeta(x) = log(1 + exp(x)) \tag{1}$$

However, in our implementation we use the softplus activation function $\zeta(x)$ (Equation 1) instead of the exponential function in the capsule layers since we empirically observed it to perform better and to be more stable. Note that softplus applied to a Gaussian distribution does not only have a similar appearance of a log-normal distribution in the same co-domain $[0, +\infty)$, as can be seen in Figure 1, but that it is a good approximation of the exponential function for x being small or negative, as the quotient of $\frac{\zeta(x)}{e^x}$ quickly converges to 1 for $x \to -\infty$.

Furthermore, we assume that the reason why the exponential activation function e^x behaves unstably is because its derivative is the exponential function itself, e^x, meaning that the gradients can become exponentially large during back-propagation, leading to the problem of exploding gradients. On the other hand, the derivative of the softplus function is the sigmoid function, which is defined in $[0, 1]$ and thus behaves more stably.

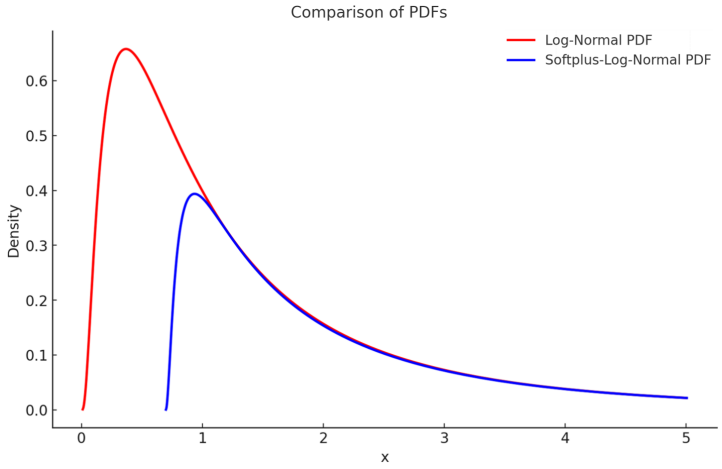

Figure 1: Difference between softplus log-normal PDF and regular log-normal PDF.

By treating the capsules as random variables, we are able generate multiple samples from the input space to approximate the true posterior distribution of object instantiation parameters.

Child capsules, Ψ, are constructed using 2D depthwise convolutions. Therefore, each child capsule $\psi \in \Psi$ is derived from just one input feature map as depthwise convolution does not mix information across input channels, which reduces the number of parameters. Furthermore, the output of the 2D depthwise convolution is a $(K, K, D \times N)$ tensor, where K is the feature map height and width respectively, D is the length of the child capsules, i.e. the depth multiplier, and N is the number of input channels. The output tensor is then reshaped to $(D, 1, 1, K \times K \times N)$ to enable the construction of parent capsules. Therefore, 2D depthwise convolution conveniently allows to move from a 3-dimensional (width, height, channels) tensor to an 4-dimensional tensor (depth, width, height, channels), thereby constructing the first-level of capsules, i.e. child capsules. We set D=8, as per Sabour et al. (2017), which acts as the depthwise multiplier in the 2D depthwise convolutional layer.

Parent capsules, Ω, are derived from child capsules through a 3D transpose convolution. Whereas each child capsule attends to a subregion of one input feature map, each parent capsule attends to all child capsules, aggregating the information the child capsules encode. The motivation behind using a 3D transpose convolution is that it enables to go from a $(D, 1, 1, N)$ tensor to a $(D', 1, 1, N')$ tensor, where $D' >> D$ and

$N' << N$. D' is 16, as defined by Sabour et al. (2017), and N' is the number of classes. The kernel size for the 3D transpose convolution is $5 \times 1 \times 1$ and the strides are (2, 1, 1) in order to double the depth dimension, i.e. capsule size.

4.2 Entropy voting

As each parent capsule, $\omega \in \Omega$, is conditioned on all child capsules, $\psi \in \Psi$, we know that there is a latent joint distribution $p(\Psi, \Omega)$, which allows to compute the agreement between child and parent capsules using information theory. The parent capsule $\omega \in \Omega$ with the highest average entropy score w.r.t. to all $\psi \in \Psi$ is considered the correct prediction.

We define the entropy voting function as:

$$f(\Psi, \Omega) = h(\Psi) + h(\Omega) - D_{\mathrm{KL}}(\Psi \parallel \Omega) \tag{2}$$

with $h(\Psi)$ and $h(\Omega)$ being the marginal differential entropy terms and $D_{KL}(\Psi||\Omega)$ being the Kullback-Leibler (KL) divergence of Ψ from Ω.

The differential entropy for a log-normal distribution is given by:

$$h(\boldsymbol{x}) = \mu_x + \frac{1}{2}\ln(2\pi e\sigma_x^2) \tag{3}$$

Thus, to obtain the capsule-wise marginal differential entropies:

$$h(\boldsymbol{\psi_i}) = \mu_{\psi_i} + \frac{1}{2}\ln(2\pi e\sigma_{\psi_i}^2) \tag{4}$$

$$h(\boldsymbol{\omega_j}) = \mu_{\omega_j} + \frac{1}{2}\ln(2\pi e\sigma_{\omega_j}^2) \tag{5}$$

To maximize $f(\Psi, \Omega)$, we want to maximize the log-normal marginal differential entropy terms (Equation 3) $h(\Psi)$ and $h(\Omega)$, while minimizing $D_{KL}(\Psi||\Omega)$; the higher the marginal entropy terms $h(\Psi)$ and $h(\Omega)$ are, the more information the underlying log-normal distributions carry, and the lower the $D_{KL}(\Psi||\Omega)$ term is, the better some $\omega \in \Omega$ describes all $\psi \in \Psi$ on average. Since capsules are bound to encode shared parameters about pose, texture, color, etc., as they are global properties, by maximizing $h(\Psi)$ and $h(\Omega)$, the capsules are encouraged to capture object-specific properties better, because maximizing differential entropy is basically the same as maximizing variance, which makes the underlying distribution spread out.

The Kullback-Leibler divergence (also known as relative entropy, Equation 6)) $D_{KL}(p||q)$ - as defined for two univariate continuous distributions (Belov & Armstrong, 2011) - is used to model agreement between the actual distribution p and a reference distribution q:

$$D_{KL}(p||q) = log\frac{\sigma_q}{\sigma_p} + \frac{\sigma_p^2 + (\mu_p - \mu_q)^2}{2\sigma_q^2} - \frac{1}{2} \tag{6}$$

Thus, to obtain the relative entropy w.r.t. ψ_i and ω_j:

$$D_{\mathrm{KL}}(\boldsymbol{\psi}_i \parallel \boldsymbol{\omega}_j) = \log\frac{\sigma_{\omega_j}}{\sigma_{\psi_i}} + \frac{\sigma_{\psi_i}^2 + (\mu_{\psi_i} - \mu_{\omega_j})^2}{2\sigma_{\omega_j}^2} - \frac{1}{2} \tag{7}$$

Therefore, relative entropy can be interpreted to measure how much information is lost when approximating p with q; a relative entropy of 0 indicates that the two distributions, p and q, have identical quantities of information and thereby no information is lost when approximating p with q. Therefore, KL divergence

sufficiently models agreement between child capsules Ψ (the actual distribution) and parent capsules Ω (the reference distribution) in our formulation.

The voting scores for each parent capsule, ω_j, are computed as:

$$z_j = \frac{1}{N} \sum_{i=1}^{N} [h(\boldsymbol{\psi}_i) + h(\boldsymbol{\omega}_j) - D_{\mathrm{KL}}(\boldsymbol{\psi}_i \parallel \boldsymbol{\omega}_j)] \tag{8}$$

Finally, to obtain the prediction probabilities, the raw voting scores, $\vec{z}$, are pushed through the softmax function:

$$\hat{y}_j = softmax(\vec{z})_j = \frac{\exp(z_j)}{\sum_{k=1}^{M} \exp(z_k)} \tag{9}$$

Algorithm 1 Entropy voting.

procedure ENTROPY VOTING(Ψ, Ω)
  $\forall \psi \in \Psi : \alpha_i \leftarrow h(\psi)_{i=1}^{|\psi|}$ ▷ Computes Eq. 4
  $\forall \omega \in \Omega : \beta_j \leftarrow h(\omega)_{j=1}^{|\omega|}$ ▷ Computes Eq. 5
  $\forall \psi \in \Psi; \forall \omega \in \Omega : \delta_{ij} \leftarrow D_{KL}(\psi_i || \omega_j)$ ▷ Computes Eq. 7
  $z_j \leftarrow \frac{1}{|\Psi|} \sum_i \alpha_i + \beta_j - \delta_{ij}$ ▷ Computes Eq. 8
  $\hat{y}_j \leftarrow \mathrm{softmax}(\vec{z})_j$ ▷ Computes Eq. 9
  return $\hat{y}_j$
end procedure

The moments are estimated directly from data using the method of moments for the log-normal distribution:

$$\hat{\mu}_x = \log \left(\frac{\mathbb{E}[x]}{\sqrt{\frac{\mathbb{E}[x]^2}{\mathrm{Var}[x]}}} \right) \tag{10}$$

$$\hat{\sigma}_x^2 = \log \left(\frac{\mathrm{Var}[x]}{\mathbb{E}[x]^2} + 1 \right) \tag{11}$$

To estimate the moments for child and parent capsules, we substitute ψ_i and ω_j for x in Equation 10 and Equation 11, respectively.

Note that the entropy voting function $f(\Psi, \Omega)$ is strongly motivated by mutual information $I(\Psi; \Omega)$: Since $D_{KL}(\Psi||\Omega) = h(\Psi, \Omega) - h(\Psi)$ and $I(\Psi; \Omega) = h(\Psi) + h(\Omega) - h(\Psi, \Omega)$, one can rewrite Equation 2 as $f(\Psi, \Omega) = h(\Psi) + I(\Psi; \Omega)$. Thus, maximizing $f(\Psi, \Omega)$ means both maximizing the marginal differential entropy of Ψ and the mutual information of Ψ and Ω. While the former guarantees that the underlying distribution carries as much information as possible, the latter ensures agreement between Ψ and Ω.

One big advantage of using entropy voting besides its theoretical justification is that it can be computed very efficiently due to its lack of mixed entropy terms; computing the relative entropy of two univariate distributions is computationally tractable as it can be done using their respective moments (Belov & Armstrong, 2011).

5 Experiments

5.1 Network architecture

The overall architecture used in our experiments is illustrated in Figure 2. The first four layers of the network are simple 2D convolutional layers with the ReLU activation function followed by batch normalization. For

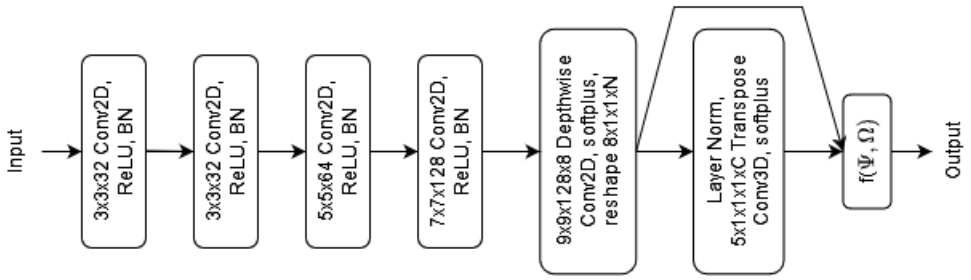

Figure 2: An abstract illustration of the network architecture w.r.t. MNIST. Note that $f(\Psi, \Omega)$ denotes the entropy voting function.

MNIST, the last convolution layer has stride of 1 and stride of 2 for SmallNorb. The preceding depthwise convolutional layer has stride of two for MNIST and stride of four for SmallNORB. Furthermore, the number of filters in the transpose 3D convolutional layer is 10 for MNIST (as can be seen in Figure 2) and 5 for SmallNorb. Otherwise there is no difference in the network configuration between the datasets.

5.2 Experimental setup

We conduct our experiments on the MNIST (LeCun et al., 2010) and smallNORB (LeCun et al., 2004) datasets as these are the standard datasets in research on capsule networks. We use the same training setup for all datasets; a batch size of 32, Adam optimizer with a learning rate of 0.0005. We train both with augmentations and completely without augmentations. We augment the images with random affine transformations; scale, translation, rotation, and shear, all within a $\pm 20\%$ range. Affine transformations are basic data augmentation techniques that apply geometric operations to the image without changing its content, e.g. as opposed to cropping. As for the hardware, we use an NVIDIA V100 GPU for training and testing.

The MNIST dataset contains a total of 70,000 grayscale images of handwritten digits. The images are of size 28x28 pixels. Each image represents a single digit ranging from 0 to 9, with the digit being in the center of the image. The training set consists of 60,000 images, while the test set contains 10,000 images.

The smallNORB dataset consists of grayscale images of 3D objects. The objects belong to five different categories, and each object was imaged by two cameras under 6 lighting conditions, 9 elevations, and 18 azimuths. The dataset is split evenly between training and testing sets, with 24,300 images each. The training set contains 5 instances (4, 6, 7, 8 and 9) of each category, and the test set the remaining 5 instances (0, 1, 2, 3, and 5).

For comparison we reimplemented CapsNet by Sabour et al. (2017) according to the description given in the paper, used the official implementation of Efficient-CapsNet by Mazzia et al. (2021), and reimplemented HVC-CapsNet (for the lack of better nomenclature) by Byerly et al. (2021), also as per the description given in the paper. A direct comparison with the results given in their papers is problematic as different augmentation methods are used, which is one of the issues capsule networks were designed to address. Furthermore, since two of the three baselines - CapsNet and Efficient-CapsNet - benefit from a reconstruction network as a regularizer, we tested our model with and without it. That being said, the parameter counts in Table 1 exclude the reconstruction regularizer for simplicity of comparison.

The reconstruction regularizer we use is a simple three-layer feedforward network with 512 and 1024 units in the first two layers, respectively, and both layers use the ReLU activation function. The output layer uses the sigmoid activation function, and the number of units is equal to the number of pixels in the input image. In the loss function, the sum of squared penalties between the normalized input pixel values and the reconstruction output is minimized. The penalty term is scaled down by a factor of 0.0005 and added to the overall loss (Sabour et al., 2017).

Table 1: Results from our implementations. Test error % averages and standard deviations of top-5 trials. No ensembles used. Note that * refers to input size 96×96, ‡ to input size 48×48, and † to input size 32×32.

DATASET	METHOD	PARAMS	FPS	W/O AUG	W/ AUG
MNIST	Our method w/o recon.	604	**161**	$\mathbf{0.28}_{\pm 0.008}$	$\mathbf{0.23}_{\pm 0.007}$
	Our method w/ recon.	604	120	$0.34_{\pm 0.03}$	$0.31_{\pm 0.03}$
	Sabour et al. (2017)	6806	86	$0.57_{\pm 0.02}$	$0.35_{\pm 0.03}$
	Mazzia et al. (2021)	161	112	$0.41_{\pm 0.02}$	$0.37_{\pm 0.06}$
	Byerly et al. (2021)	1237	66	$0.48_{\pm 0.02}$	$0.38_{\pm 0.02}$
smallNORB	Our method w/o recon.*	806	**152**	$\mathbf{2.61}_{\pm 0.09}$	$\mathbf{3.84}_{\pm 0.15}$
	Our method w/ recon.*	806	113	$2.97_{\pm 0.11}$	4.65 ± 0.26
	Mazzia et al. (2021)‡	151	108	$10.58_{\pm 1.15}$	$9.25_{\pm 0.27}$
	Sabour et al. (2017)†	6640	84	$12.88_{\pm 0.31}$	$11.6_{\pm 0.28}$

5.3 Results

As can be seen in Table 1 below, our approach outperforms all of the three baselines on every dataset, not only in the error rates but also in inference speed, given in frames per second (FPS). Compared to CapsNet the number of parameters is one order of magnitude lower although it does not yet reach the low parameter count of Efficient-CapsNet. Regardless, our method still achieves a higher FPS. This is mainly due to differences in the voting procedure, as attention-based methods tend to be computationally more expensive.

For the smallNORB experiments, we use original input size for our model, and different input sizes for the baselines as defined in the respective papers. In the case of CapsNet, smallNORB images are first resized to 48×48 and then patches of 32×32 are cropped out (Sabour et al., 2017), whereas in the case of Efficient-CapsNet, the images are resized to 64×64, after which patches of 48×48 are cropped out (Mazzia et al., 2021). We noticed that CapsNet performs suboptimally on inputs sizes larger than 32×32, and that Efficient-CapsNet is limited to only two different input sizes, i.e. 28×28 and 48×48 (hence experiments only on MNIST and smallNORB). We tested our own method on the original input sizes for smallNORB, and outperformed both baselines without any resizing or cropping.

We also tested our model on the original smallNORB dataset with input image size of 96×96, and achieved a test error % of $3.84_{\pm 0.15}$ with augmentations and $2.61_{\pm 0.09}$ without augmentations. These results will be analyzed further in section 5.4.

Note that the test results given in the baseline papers (see Table 2) differ from our own experiments, mainly due to their use of different data augmentations and experimental setup. Moreover, since one of the main advantages of capsule networks from a theoretical point of view is the ability to learn robust abstract representations without the need for data augmentation, we believe that it is best to compare different capsule network approaches without any data augmentation at all, but provide results with data augmentation as well. However, while Sabour et al. (2017), Mazzia et al. (2021), and Byerly et al. (2021) use different augmentations compared to the augmentations we use (they also differ from each other), we use the same augmentation setup for all methods in our experiments.

It is also worth to note that the biggest discrepancy in terms of the results from our experiments and what was reported by the authors in their respective papers is between our implementation of HVC-CapsNet (Byerly et al., 2021) as tested on MNIST, and the results reported by the authors on MNIST. This might be due to the specific data augmentation techniques used by the authors, which include cropping a 4×4 patch of pixels from the input image, for example. However, it can also be that there is, for example, a disconnect between the description of the model as provided in the paper and the implementation details of the model

Table 2: Test error % reported in other CapsNet works. Best only, if no standard deviation is given. All reported results use some form of data augmentation or reconstruction regularizer. Results obtained with ensembles in italics.

METHOD	MNIST	SmallNORB
Sabour et al. (2017)	$0.25_{\pm 0.005}$	2.7
Hinton et al. (2018)	-	1.4
Hahn et al. (2019)	-	$15.91_{\pm 1.09}$
Rajasegaran et al. (2019)	0.28	-
Ribeiro et al. (2020)	-	$1.6_{\pm 0.06}$
Zhang et al. (2020)	0.38	2.2
De Sousa Ribeiro et al. (2020)	$0.28_{\pm 0.01}$	$1.4_{\pm 0.09}$
Mazzia et al. (2021)	$0.26_{\pm 0.0002}$	$2.54_{\pm 0.003}$
Byerly et al. (2021)	*0.13* (0.17)	-
Renzulli et al. (2022)	$0.44_{\pm 0.0002}$	-

used in their experiments. Regardless, as can be seen in Table 1, of the chosen baselines, HVC-CapsNet performs the poorest on MNIST using generic data augmentations. That being said, none of the baseline implementations achieved performance similar to what was reported in their respective papers.

For completeness, Table 2 also gives the results of the three baseline methods as well as additional ones as given in the respective papers. Note, that the exact results cannot directly be compared to each other since they use different augmentation techniques, some use a reconstruction regularizer, and some results are based on ensembles. Moreover, most of the results are best only and not averages with some given standard deviation. Nevertheless, our proposed method, coupled with simple affine transformations as data augmentation, achieves better performance on MNIST than all approaches but Byerly et al. (2021), which we were not able to reproduce using our implementation of their approach.

However, when considering other datasets, our method does not exhibit comparable performance to the leading approaches reported in the table, which may be because we used the same model for all datasets, whereas others tweaked their model to the dataset in question - or used ensembles - coupled with extensive data augmentation techniques. We chose to not tweak our model to each dataset or use an ensemble because we were interested in exploring the generalization ability of our approach in order to investigate the potential of capsules as-is. That is also why we deem it important to measure the performance of capsule networks without data augmentation.

This brings forth an important question regarding the effectiveness of data augmentation in the context of capsule networks. Considering the fundamental objective of capsules, which is to acquire robust and equivariant object representations and instantiation parameters, the use of data augmentation becomes a matter of inquiry. Capsule networks aim to address the limitations associated with the need for extensive data augmentation, which can be computationally strenuous. One of the primary challenges that capsules seek to overcome is the reliance on data augmentation techniques, and an exhaustive amount of data in general, which are computationally expensive.

5.4 Ablation Study

The motivation for this ablation study arises from the hypothesis that an inherent relationship exists between between child and parent capsules, which can be effectively captured by the proposed voting term $h(\Psi) + h(\Omega) - D_{KL}(\Psi||\Omega)$. This formulation suggests that a balance of uncertainty and alignment is crucial for robust decision-making. The ablation study is designed to empirically validate whether this theoretical relationship holds in practice and to evaluate the contribution of each component — $h(\Psi)$, $h(\Omega)$, and $D_{KL}(\Psi||\Omega)$ — towards achieving optimal performance across the chosen datasets.

Table 3: Results from ablation study, comparing different voting terms.

DATASET	VOTING TERM	W/O AUG	W/ AUG
MNIST	$h(\Psi) + h(\Omega) - D_{KL}(\Psi\|\|\Omega)$	$\mathbf{0.28}_{\pm 0.008}$	$\mathbf{0.23}_{\pm 0.007}$
	$h(\Psi) + h(\Omega)$	$0.37_{\pm 0.009}$	$0.31_{\pm 0.019}$
	$h(\Omega) - D_{KL}(\Psi\|\|\Omega)$	$0.33_{\pm 0.018}$	$0.29_{\pm 0.014}$
	$-D_{KL}(\Psi\|\|\Omega)$	$0.50_{\pm 0.011}$	$0.43_{\pm 0.026}$
	$h(\Omega)$	$0.36_{\pm 0.009}$	$0.30_{\pm 0.022}$
smallNORB	$h(\Psi) + h(\Omega) - D_{KL}(\Psi\|\|\Omega)$	$\mathbf{2.61}_{\pm 0.09}$	$\mathbf{3.84}_{\pm 0.15}$
	$h(\Psi) + h(\Omega)$	$3.86_{\pm 0.35}$	$5.13_{\pm 0.27}$
	$h(\Omega) - D_{KL}(\Psi\|\|\Omega)$	$3.03_{\pm 0.14}$	$4.00_{\pm 0.21}$
	$-D_{KL}(\Psi\|\|\Omega)$	$4.31_{\pm 0.37}$	$6.04_{\pm 0.29}$
	$h(\Omega)$	$3.63_{\pm 0.19}$	$4.15_{\pm 0.52}$

The ablation study results for the MNIST dataset demonstrate that incorporating the voting term $h(\Psi) + h(\Omega) - D_{KL}(\Psi||\Omega)$ leads to the best performance, achieving the lowest error rates both with and without data augmentation. This suggests that the combination of entropy terms $h(\Psi)$ and $h(\Omega)$ along with the Kullback-Leibler divergence voting term $D_{KL}(\Psi||\Omega)$, synergistically improves decision-making. Other terms, such as $h(\Psi) + h(\Omega)$ and $h(\Omega) - D_{KL}(\Psi||\Omega)$, are outperformed by the full combination, emphasizing the necessity of integrating all components. Interestingly, relying solely on $-D_{KL}(\Psi||\Omega)$ or $h(\Omega)$ results in significantly higher error rates, highlighting their insufficiency in isolation.

For the smallNORB dataset, a similar trend emerges, with $h(\Psi) + h(\Omega) - D_{KL}(\Psi||\Omega)$ again yielding the best performance. It achieves the lowest error rates in both settings, with and without augmentation. The improvements seen with this term are even more pronounced compared to other terms, particularly when data augmentation is applied. For example, $-D_{KL}(\Psi||\Omega)$ performs poorly, yielding a very high mean error rate with augmentation, which is notably higher than the best-performing term. This underscores the importance of balancing entropy terms with divergence for robust performance. The results indicate that while smallNORB is more challenging than MNIST, the proposed combination of terms generalizes well to more challenging datasets. SmallNORB performs worse with augmentation likely due to its inherent complexity, as it consists of diverse object images captured under varying viewpoints, where the viewpoints are different in training and test sets. Augmentation introduces additional variability, which may amplify the dataset's existing challenges, such as increased ambiguity in object recognition under altered transformations. This suggests that while augmentation enhances robustness in simpler datasets like MNIST, it can overwhelm models on inherently complex datasets like smallNORB by exacerbating the need for fine-grained feature discrimination.

Overall, the results show that $h(\Psi)+h(\Omega)-D_{KL}(\Psi||\Omega)$ is the most effective voting term across both datasets, regardless of augmentation. The term effectively balances marginal entropy, captured by the terms $h(\Psi)$ and $h(\Omega)$, with alignment via $-D_{KL}(\Psi||\Omega)$, leading to consistent performance improvements. The superior performance on both MNIST and smallNORB also highlights its versatility, as it excels in both simpler (MNIST) and more complex (smallNORB) scenarios. These findings underline the importance of jointly optimizing entropy and divergence in tasks requiring robust generalization, particularly when augmentation techniques are used to simulate real-world variability.

6 Conclusion

In this work, we proposed a non-iterative voting procedure between child and parent capsules that is motivated by information theory. We proved empirically that through discriminative learning, our method is able to learn representations that generalize sufficiently well, even without data augmentation, by maximizing

the marginal differential entropies of all capsules while minimizing the relative entropy between child and parent capsules.

Thus, our work provides two contributions to the field of capsule networks; (1) an interpretation of capsules and the voting procedure that is firmly rooted in information theory, which also allows for the use of a differentiable activation function for capsules, and (2) the use of 3D transpose convolution to derive parent capsules from child capsules so that changes in dimensions and sizes between the capsules are taken into account, instead of using workaround matrix operations to achieve the same effect.

Although our proposed method exhibits degradative performance on more complex datasets, the experiments also indicate that it benefits from a larger input size and, to some extent, larger capsule size. Hence, there is room for research on capsule networks with respect to more complex, higher resolution images, especially without data augmentation. In general, methods that aim to make data augmentation unnecessary should receive more attention in the research community as data augmentation is a roundabout way of addressing one of the most fundamental challenges in computer vision.

References

Dmitry I Belov and Ronald D Armstrong. Distributions of the kullback–leibler divergence with applications. *British Journal of Mathematical and Statistical Psychology*, 64(2):291–309, 2011.

Adam Byerly, Tatiana Kalganova, and Ian Dear. No routing needed between capsules. *Neurocomputing*, 463: 545–553, 2021. ISSN 0925-2312.

Jaewoong Choi, Hyun Seo, Suii Im, and Myungjoo Kang. Attention routing between capsules. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 1981–1989. IEEE Computer Society, 2019.

Fabio De Sousa Ribeiro, Georgios Leontidis, and Stefanos Kollias. Introducing routing uncertainty in capsule networks. *Advances in Neural Information Processing Systems*, 33:6490–6502, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvian Gelly, Jakob Uszoreit, and Neil Houldby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*, 2021.

Taeyoung Hahn, Myeongjang Pyeon, and Gunhee Kim. Self-routing capsule networks. *Advances in neural information processing systems*, 32, 2019.

Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I 21*, pp. 44–51. Springer, 2011.

Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *International conference on learning representations*, 2018.

Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:II–104, 2004.

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Vittorio Mazzia, Francesco Salvetti, and Marcello Chiaberge. Efficient-capsnet: Capsule network with self-attention routing. *Scientific reports*, 11(1):14634, 2021.

Jathushan Rajasegaran, Vinoj Jayasundara, Sandaru Jayasekara, Hirunima Jayasekara, Suranga Seneviratne, and Ranga Rodrigo. Deepcaps: Going deeper with capsule networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10725–10733, 2019.

Riccardo Renzulli, Enzo Tartaglione, and Marco Grangetto. Rem: Routing entropy minimization for capsule networks. *arXiv preprint arXiv:2204.01298*, 2022.

Fabio De Sousa Ribeiro, Georgios Leontidis, and Stefanos Kollias. Capsule routing via variational bayes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3749–3756, 2020.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *Advances in neural information processing systems*, 30, 2017.

Dilin Wang and Qiang Liu. An optimization view on dynamic routing between capsules. In *International Conference on Learning Representations*, 2018.

Liheng Zhang, Marzieh Edraki, and Guo-Jun Qi. Cappronet: Deep feature learning via orthogonal projections onto capsule subspaces. *Advances in neural information processing systems*, 31, 2018.

Suofei Zhang, Quan Zhou, and Xiaofu Wu. Fast dynamic routing based on weighted kernel density estimation. *Cognitive Internet of Things: Frameworks, Tools and Applications*, pp. 301–309, 2020.