QUANTITATIVE PERFORMANCE ASSESSMENT OF CNN UNITS VIA TOPOLOGICAL ENTROPY CALCULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Identifying the status of individual network units is critical for understanding the mechanism of convolutional neural networks (CNNs). However, it is still challenging to reliably give a general indication of unit status, especially for units in different network models. To this end, we propose a novel method for quantitatively clarifying the status of single unit in CNN using algebraic topological tools. Unit status is indicated via the calculation of a defined topological-based entropy, called feature entropy, which measures the degree of chaos of the global spatial pattern hidden in the unit for a category. In this way, feature entropy could provide an accurate indication of status for units in different networks with diverse situations like weight-rescaling operation. Further, we show that feature entropy decreases as the layer goes deeper and shares almost simultaneous trend with loss during training. We show that by investigating the feature entropy of units on only training data, it could give discrimination between networks with different generalization ability from the view of the effectiveness of feature representations.

1 INTRODUCTION

Convolutional neural networks (CNNs) have achieved great success in various vision tasks Szegedy et al. (2016); Redmon et al. (2016); He et al. (2017a). The key to such success is the powerful ability of feature representations to input images, where network units[1] play a critical role. But impacted by the diverse training deployments and huge hypothesis space, networks even with the same architecture may converge to different minima on a given task. Although units between these networks could present similar function for the same task, yet they may have completely different activation magnitudes. Consequently, this makes it fairly hard to give a general indication of the status for a given network unit with respect to how well features are represented by it from images in the same class.

Being rough indicators in practice, magnitude responses of units are usually chosen simply Zhang et al. (2018) or processed statistically (such as average mean) Li et al. (2016); Luo et al. (2017) based on the idea of matched filtering. However, firstly these indicators are apparently sensitive to rescaling operations in magnitude. If performing a simply rescaling operation to the weights such as the strategy introduced in Neyshabur et al. (2015), the results of the network and the function of each unit would all remain unchanged, but these indicators would vary along with the rescaling coefficient. Secondly, as the spatial information in the unit is completely discarded, they could not give discrimination between units with and without random patterns, for example units separately outputted by a well-trained and random-initialized CNN filter. Without a valid indication regarding the mentioned situations, these indicators fail to ensure the universal applicability for units in different network models.

In this paper, we attempt to investigate the status of units from a new perspective. Roughly speaking, natural images in the same class have common features, and meanwhile the locations of these features are spatially correlated in global. For effective units, features are picked out and represented by high activation values in the units. And due to the locality nature in feature extraction by convolution, this global spatial pattern between the common features would be preserved synchronously in

[1]Regarding the term unit, a unit is the perceptive node in networks, which generally refers to the activated feature map outputted by a convolutional filter in CNNs.

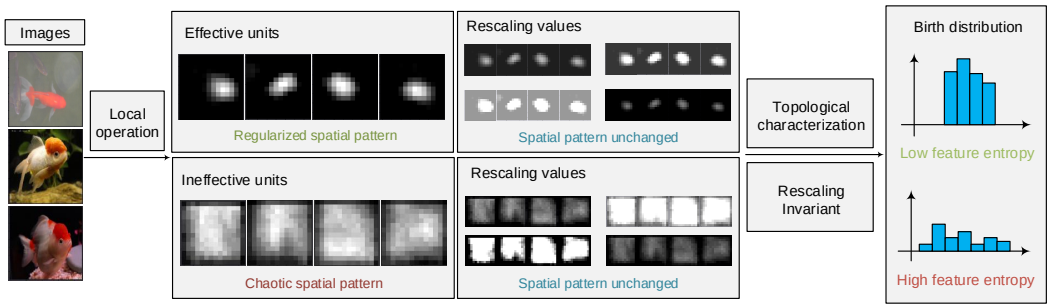

Figure 1: Comparisons between the effective units and ineffective units. For effective units, since the spatial pattern of the features in the images would be preserved, units should stably present this regularized spatial pattern. We propose a topological-based quantity called feature entropy to indicate the unit status, giving reliable indication in various situations like rescaling the values.

the counterpart representations in the effective units. In contrast, for ineffective units, since being incapability of effectively representing these common features, representations would be in chaos and marks of this pattern is vague. This provides a valid road for performance assessment of individual units, and critically it is rescaling-invariant and universally applicable to any CNN architecture.

The investigation of such pattern could naturally lead to topological approaches because knowledge of topological data analysis such as barcodes Ghrist (2008) provides valuable tools to resolve the intrinsic patterns in raw data. Along this line, firstly we introduce a method for characterizing the spatial pattern of feature representations in units for a single sample by incorporating with the topological tools, and then use information entropy to evaluate the stability of this spatial characterizations for various images sampled from the same class, where we call it feature entropy. In this way, a unit is judged to be effective if its feature entropy is high, otherwise ineffective.

In our experiments, we find that feature entropy would gradually decrease as the layer goes deeper and the evolution trends of feature entropy and losses are almost the same during network training. We show that the feature entropy could provide reliable indication of unit status in situations like weight-rescaling and the emergence of random pattern. Finally, we show the value of feature entropy in giving discrimination between networks with different generalization ability by investigating only the training set.

2 Related Works

One line of research that attracts many researchers is seeking solutions in a way of visualizing what features have learned by the units Zeiler & Fergus (2014); Zhou et al. (2014); Mahendran & Vedaldi (2015); Simonyan et al. (2013). Status is generally identified depending on the degree of alignment between the visualized features and the human-visual concepts Bau et al. (2017); Zhou et al. (2018a); Bau et al. (2020). On the one hand, they meanwhile give excellent visual interpretation of each unit; on the other hand, it hinders its universal application to arbitrary tasks and models in which units' functionalities may be unrecognized to human Wang et al. (2020).

Another related research trace lies in the field of network pruning, where they concentrate on using simple methods to roughly select less important units within a network. Typical approaches include the L1-Norm of units Luo et al. (2017), Average Percentage of Zeros (APoZ) in units He et al. (2017b), some sparse-based methods Li et al. (2019); Yoon & Hwang (2017), and so on. Despite commonly used in practice, since without a specific processing on units in diverse situations, they are unable to provide a general indication for units in different networks.

Besides, Morcos et al. (2018) introduce the class selectivity from neuroscience to investigate the selectivity over classes for a specific unit, on the basis of calculating the mean units. Alain & Bengio (2016) propose linear classifier probe, where they report the degree of linear classification of units in intermediate layers could somehow characterize the status of units.

Lastly, we would like to discuss some recent works related to topological approaches in deep learning. Naitzat et al. (2020) demonstrates the superiority of using ReLu activation by studying the

changes in Betti numbers of a two-class neural network. Montúfar et al. (2020) use neural networks to predict the persistent homology features. In Gabrielsson & Carlsson (2019), by using barcode, they show the topological structure changes during training which correlates to the generalization of networks. Rieck et al. (2018) proposes neural persistence, a topological complexity measure of network structure that could give a criterion on early stopping. Guss & Salakhutdinov (2018) empirically investigates the connection between neural network expressivity and the complexity of dataset in topology. In Hofer et al. (2017), topological signatures of data are evaluated and used to improved the classification of shapes.

3 METHOD

Without loss of generality, we assume the input image for network model to be square. For input image sample $\boldsymbol{I}$ of a given class with size $n \times n$ to be represented by a unit $\boldsymbol{U}$ with size $m \times m$ via feature extraction processing f in CNN, we have,

$$f : \boldsymbol{I} \rightarrow \boldsymbol{U} \tag{1}$$

For image $\boldsymbol{I}$, features are specifically arranged, where each feature has an associated spatial location in the image. After perceived by $\boldsymbol{U}$, features are represented by high activation values at the corresponding locations in the unit. Basically, there are two steps in our assessment of unit performance: firstly, characterizing the spatial pattern hidden in these high activation values in a unit for a single image; secondly, evaluate the stability of this characterization when giving multiple image samples.

3.1 CHARACTERIZING THE SPATIAL PATTERN OF FEATURE REPRESENTATIONS IN A UNIT

For $\boldsymbol{U}_{i,j}$ with a grid structure, the location of an element generally refers to its coordinate index (i, j). And intuitively, the spatial pattern hidden in the elements denotes certain regular relationship among their coordinate indices. So, it is natural to model such relationship with graph structure and tackle it with topological tools in the following.

Unit and graph We use the edge-weighted graphs Mehmet et al. (2019) as our basic model and construct the weighted graph $\mathcal{G} = (V, E)$ from unit $\boldsymbol{U}_{i,j}$, where V is the vertex set and E is the edge set. Define the adjacency matrix $\boldsymbol{A}$ of $\mathcal{G}$ as follows,

$$\boldsymbol{A} \in \mathbb{R}^{m \times m} : \boldsymbol{A}_{i,j} = \boldsymbol{U}_{i,j} \tag{2}$$

It should be noted that the individual element of $\boldsymbol{A}$ is the weight of edge in $\mathcal{G}$, which conveys the intensity of corresponding point in the $\boldsymbol{U}$.

A family of undirected graphs $\mathcal{G}^{(v)}$ with adjacency matrices $\boldsymbol{A}^{(v)}$ could be constructed by following the typical implementation of the sublevel set,

$$\boldsymbol{A}_{i,j}^{(v)} = \mathbf{1}_{\mathrm{A_{i,j} \geq a^{(v)}}} \tag{3}$$

where w_v is the vth value in the descend ordering of elements of $\boldsymbol{A}$ and $\mathbf{1}_{(\cdot)}$ is indicator function. Here, we take the adjustment of $\boldsymbol{A}^{(v)} = max(\boldsymbol{A}^{(v)}, (\boldsymbol{A}^{(v)})^T)$ to ensure the adjacency matrices $\boldsymbol{A}^{(v)}$ of undirected graphs to be symmetric.

So $\mathcal{G}^{(v)} = (V^{(v)}, E^{(v)})$ is the subgraph of $\mathcal{G}$ where $V^{(v)} = V$ and $E^{(v)} \subset E$ only includes the edges whose weights are greater than or equal to $a^{(v)}$. We have the following graph filtration,

$$\mathcal{G}_1 \subset \mathcal{G}_2 \subset \mathcal{G}_3 \subset \cdots \subset \mathcal{G} \tag{4}$$

To be more specifically, in this sublevel set filtration, it starts with the vertex set, then rank the edge weights from the maximum a_{max} to minimum a_{min}, and let the threshold parameters decrease from a_{max} to a_{min}. At each step, we add the corresponding edges to obtain the threshold subgraph $\mathcal{G}(v)$.

Fig.2 illustrates the construction of certain subgraph through a toy example. Consider the unit $\boldsymbol{U}_{i,j}$. We circle the locations of the top 4 largest elements in $\boldsymbol{U}_{i,j}$ (Fig.2A). Then the nonzero elements in adjacency matrix $\boldsymbol{A}^{(4)}$, $\{(1, 2), (4, 3), (2, 4), (3, 1)\}$, is located (Fig.2B) and corresponding subgraph $\mathcal{G}^{(4)}$ is constructed (Fig.2C).

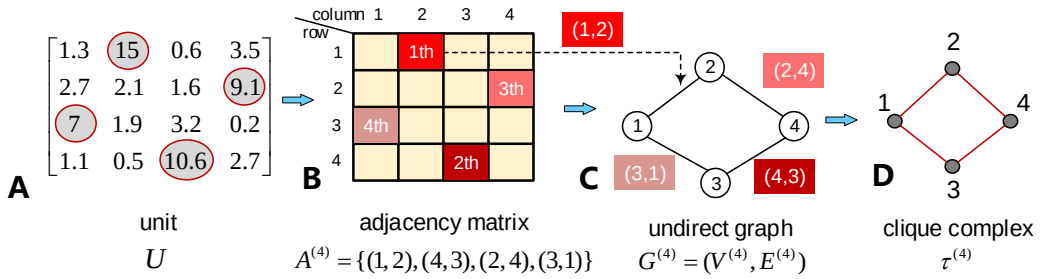

Figure 2: Example of the conversion from a unit to its clique complex.

Complex filtration To further reveal the correlation structure in the graphs, they are typically converted into certain kinds of topological objects, where topological invariants are calculated for capturing the high-level abstraction of correlation structure. Here, by following the common method in Horak et al. (2009); Giovanni et al. (2013), each graph $\mathcal{G}^{(v)}$ is converted to simplicial complex (also called clique complex) $\tau^{(v)}$, as shown in Fig.2D. In this way, we have complex filtration corresponding to graph filtration (Eq.4).

$$\tau^{(1)} \subset \tau^{(2)} \subset \tau^{(3)} \subset \cdots \subset \tau \tag{5}$$

This filtration describes the evolution of correlation structure in graph $\mathcal{G}$ along with the decreasing of threshold parameter. Fig.3A shows the complex filtration of the previous example (Fig.2).

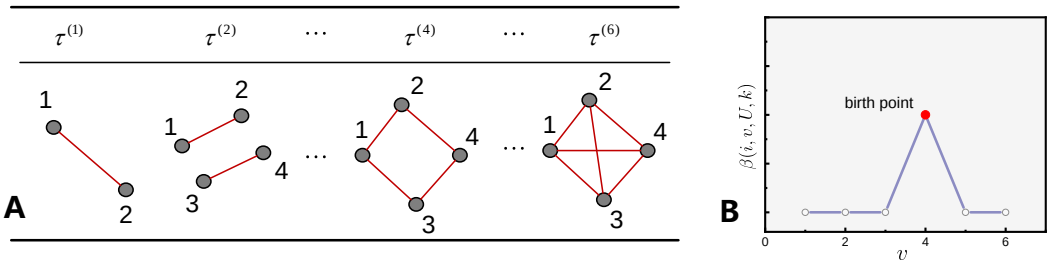

Figure 3: Instance of complex filtration (A) and Betti curve (B).

So far, we have completed the characterization from unit to the topological objects. Other than our strategy, we also discuss other alternative method, which maps the unit to the cubical complex Kaczynski et al. (2004). See Appendix for more details.

Betti curve and its charaterization Next, kth Betti number Hatcher (2002) of each element in the complex filtration could be calculated using the typical computational approach of persistent homology Ninna et al. (2017).

$$\tau^{(v)} \mapsto \beta(\tau^{(v)}) \tag{6}$$

Intuitively, kth Betti number $\beta(\tau^{(v)})$ could be regarded as the number of k-dimensional 'circle's or 'hole's or some higher order structures in complex $\tau^{(v)}$. On the other hand, many meaningful patterns in the unit would lead to the 'circle's or 'hole's of complexes in the filtration (Eq.5), see Fig.2 for illustration. In particular, the number of 'hole's is typically used as an important quantitative index for featuring such patterns. Hence, the kth Betti numbers $\beta(\tau^{(v)}), v \in \{1, \cdots, n\}$ could be arranged into so called kth Betti curves $\beta(\boldsymbol{U}, v, k)$ for the unit $\boldsymbol{U}$. Fig.3B shows the 1th Betti curve of filtration in Fig.3A.

Once having obtained the Betti curve, one needs to interpret the Betti curve and extract its core characterization. Although there exists many choices of distance between two topological diagrams such as persistence images Adams et al. (2017), persistence landscape Bubenik et al. (2015) and persistence entropyNinna et al. (2017), we find that the simple birth time of the Betti curves $\beta(\boldsymbol{U}, v, k)$ is sufficient in this characterization,

$$b(\boldsymbol{U}, k) = \inf\{v | \beta(\boldsymbol{U}, v, k) \neq 0\} \tag{7}$$

We call $b(\boldsymbol{U}, k)$ the birth point. Birth point is the indication of the critical element in complex filtration that begins to carry "hole" structure (Betti number is nonzero). It is an important sign that some essential change has occurred in complex filtration, which implies the appearance of regularized spatial pattern of notable components in the unit. Meanwhile, in some cases, no spatial pattern appear in the components in the unit, so $\beta(\boldsymbol{U}, v, k)$ constantly equals to zero, meaning that birth point doesn't exist. In general, this would happen when the unit is unable to give representations for the image, where its values are almost all zeros.

3.2 Assessing the unit performance using feature entropy

For a specific target class $\mathcal{C}$, consider image $\boldsymbol{I}_{\mathcal{C}}^{(i)}$ sampled from the dataset $\{\boldsymbol{I}_{\mathcal{C}}^{(i)}\}$ of $\mathcal{C}$. By perceived with an *ideal* unit $\boldsymbol{U}$, it should present similar pattern with other image samples. In other words, the birth point obtained from each realization of units should be relatively close. That is to say, the performance of *good* unit for certain target class should be *stable* over all the samples of this class. It is the key idea for performance assessment of network unit.

Birth distribution Essentially, birth point $\mathrm{b}_{\mathcal{C}}(i, \boldsymbol{U}, k)$ is a random variable since sampling images from the specific class $\mathcal{C}$ could be regarded as statistical experiments. In fact, the probability space (Ω, Σ, P) could be constructed. The elements in sample space Ω are the unit $\boldsymbol{U}$ resulted from the image samples in dataset of class C. Σ could be set as common discrete σ-field and probability measure P is uniformly distributed on Ω. In other words, every image sample has an equal chance to be chosen as the input of network model. Afterwards, $\mathrm{b}_{\mathcal{C}}(i, \boldsymbol{U}, k)$ is defined as a random variable on Ω (where the argument is i, and $\boldsymbol{U}$ and k are parameters),

$$\mathrm{b}_{\mathcal{C}}(i, \boldsymbol{U}, k)(\cdot) : \Omega \to \mathbb{Z} \tag{8}$$

with the probability distribution

$$P_{\mathcal{C},\boldsymbol{U},k}(x) = P(\mathrm{b}_{\mathcal{C}}(i, \boldsymbol{U}, k) = x) = \frac{b_x}{\#(\Omega)}, \tag{9}$$

where

$$b_x = \sum_{j=1}^{\#(\Omega)} \mathbf{1}_{\mathrm{b}_{\mathcal{C}}(\mathrm{i,U,k})=\mathrm{x}} \tag{10}$$

Here the composite mapping $\mathrm{b}_{\mathcal{C}}(i, \boldsymbol{U}, k)(\cdot)$ from Ω to $\mathbb{Z}$ is composed of all the operation mentioned above, including construct weighted graphs, building complex filtration, calculating Betti curve and extracting birth point.

The degree of concentration of $P_{\mathcal{C},\boldsymbol{U},k}(x)$ gives a direct view about the performance of unit $\boldsymbol{U}$ on class $\mathcal{C}$, as illustrated in Fig.1. More specifically, if the distribution presents close to a degenerate-like style, it means that the underlying common features of the class $\mathcal{C}$ could be *stably* perceived by the unit $\boldsymbol{U}$. On the contrary, the distribution presents close to a uniform-like style when features are perceived almost blindly, indicating that unit $\boldsymbol{U}$ is invalid for $\mathcal{C}$. In summary, the degree of concentration of $P_{C,\boldsymbol{U},k}(x)$ is supposed to be an effective indicator of the performance of unit $\boldsymbol{U}$.

Feature entropy To further quantize the degree of concentration of birth distribution $P_{\mathcal{C},\boldsymbol{U},k}(x)$, we introduce its entropy $H_{\mathcal{C},\boldsymbol{U},k}$ and call it feature entropy,

$$H_{\mathcal{C},\boldsymbol{U},k} = -\sum_{x} P_{\mathcal{C},\boldsymbol{U},k}(x) \log P_{\mathcal{C},\boldsymbol{U},k}(x) \tag{11}$$

It should be noted that the birth point in Eq.7 may not exist for some input images in class $\mathcal{C}$ and unit $\boldsymbol{U}$. For unit $\boldsymbol{U}$, the percentage of images in class $\mathcal{C}$ having birth points, termed as selective rate $\epsilon_{\mathcal{C},\boldsymbol{U}}$, is also a crucial factor to the effectiveness of $\boldsymbol{U}$ on $\mathcal{C}$. If the $\epsilon_{\mathcal{C},\boldsymbol{U}}$ is too low, it indicates that the unit could not perceive most of the image samples in this class. In this situation, extremely low $\epsilon_{\mathcal{C},\boldsymbol{U}}$ would cause the feature entropy approach to zero, but the unit should be judged as completely invalid. Therefore, we rule out this extreme case by setting a threshold p and for completeness, and assign the feature entropy associated with the maximum of feature entropy for the set of samples,

$$H_{\mathcal{C},U,k} = \begin{cases} H_{\mathcal{C},U,k} & \epsilon_{\mathcal{C},\boldsymbol{U}} \geq p \\ \epsilon_{\mathcal{C},\boldsymbol{U}} \cdot \log|\Omega| & \epsilon_{\mathcal{C},\boldsymbol{U}} < p \end{cases} \tag{12}$$

Here, p is prescribed as 0.1 in our computation.

4 EXPERIMENTS

For experiments, we use the VGG16 network architecture to perform the image classification task on the ImageNet dataset. Unless otherwise stated, the exampled VGG16 model is trained from scratch with the hyperparameters deployed in Simonyan & Zisserman (2014). For clarity, we only calculate birth points $\mathrm{b}_{\mathcal{C}}(i, U, 1)$ based on 1th betti curve for all the units. Also, it should be noted that our method focuses on the behaviors of feature extraction operations and has not utilized any kind of particular nature of VGG network architecture, and all our investigation could be applicable to other network architectures effortlessly.

4.1 CALCULATION FLOW

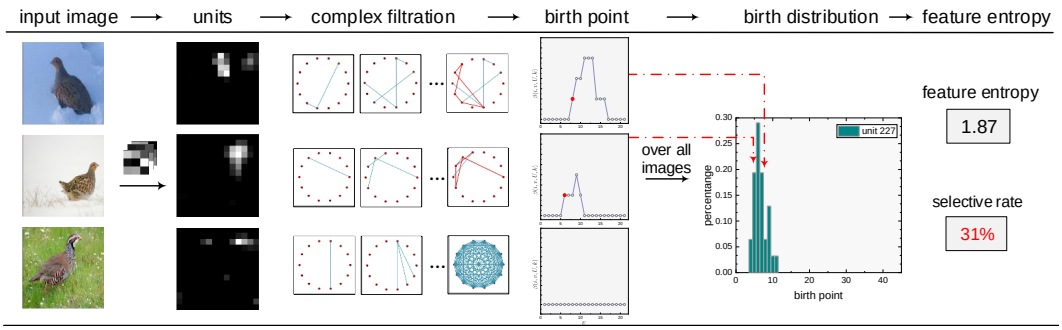

Figure 4: Calculation flow of feature entropy.

As an example, the class partridge (wnid n01807496) in ImageNet is chosen for illustration. Here, we sample 100 images from its training set as the image set for building the birth distribution. Fig.4 shows the calculation flow. It starts from extracting all the units for each image sample. By characterizing the unit with graph model, each unit corresponds to a specific filtration. Then, using formular 7, we can obtain the birth point of each unit. In this way, the distribution of birth point could be set up via Eq.9 over the sampled images. Fig.4 shows the histogram of the birth point distribution for a specific unit in the the last convolution layer "block5_conv3". Likewise, the feature entropy can be calculated via Eq.11 for all other units.

4.2 LAYER AND TRAINING ANALYSIS

Layer analysis Here, we check the status of units in each convolutional layer, where we average the feature entropy across all the units within the layer to indicate the overall status of units in this layer. Using the same image set in the previous section, Fig.5A(1-2) give the comparison between the results of the convergence model and the random-initialized model.

For the converge model, we could clearly see in Fig.5A(1) that the feature entropy continually decrease as the layers go deeper. This is as expected because as the layer goes deeper, units are considered to perceive more advanced features based on the representations in the previous layer, so the spatial pattern in these features would be more significant. As for the random-initialized model, since units are incapable to perceive the common features, no clear decrease of feature entropy could be found and feature entropy in every layer is higher than that in the convergence model. Meanwhile in Fig.5A(2), the layers in the convergence model present a higher selective rate than those in the random-initialized model, except for the last convolutional layer "block5_conv3". Also, the selective rate of the last convolutional layer is much more lower than other layers. The low feature entropy and fairly high selective rate indicate that for the convergence model, units in the last convolutional layer present strong specialization and exhibit the most effective representations of features to this class comparing to units in other layers.

Further, for the convergence model, we randomly choose 100 classes in ImageNet and average the corresponding results across all these classes to give the ensemble view, as shown in Fig.5A(3). The results are very similar to that in (1-2), which confirms the fact.

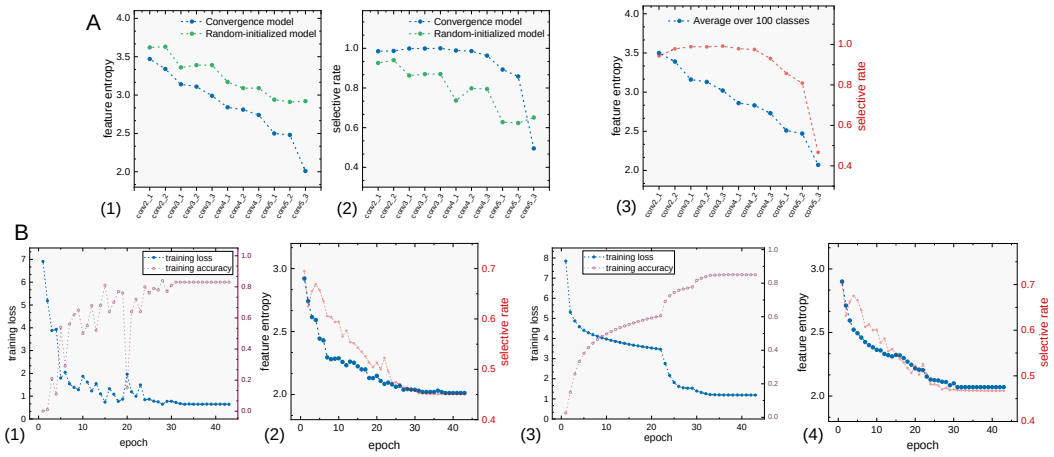

Figure 5: (A) Comparisons of feature entropy (1) and selective rate (2) of different layers between different convergence model and random-initialized model, and (3) shows the results over 100 classes. (B) Simultaneous evolution of training loss and feature entropy during training for the chosen class (1-2) and for the 100 classes (3-4).

Training analysis Then, we investigate the variation of feature entropy of the last convolutional layer during training. Fig.5B(1-2) show the results on the same example class and Fig.5B(3-4) show the results across the same 100 chosen classes. In both situations, we could find the feature entropy decreases during training, which means that units are gradually learned to be able to perceive the common features in the class. And remarkably, the decreasing pattern of feature entropy and that of training cross-entropy loss coincide approximately. Both of them experience a comparable big drop in the first epoch and gradually down to the convergence level. This shows that feature entropy is a valid indicator of network performance.

4.3 Indicator of status of network unit

To investigate the ability of feature entropy as indicator of unit status, we make comparisons with some commonly-used analogous network indicators including L1-norm Li et al. (2016), APoZ He et al. (2017b), and a more generalized form of class selectivity used in Zhou et al. (2018b). Here, the unit and the image set in the previous subsection are still used in the following demonstration.

Rescaling investigation The comparison is implemented by rescaling the magnitude of values to half for all the input images or all the CNN filters connecting to the layer. Both the two implementations could potentially cause the values in units within the layer vary with the same scale, but in general have no substantial impact on the network performance and the function of each unit. In other words, units should be indicated as almost the same with or without such implementation.

Table 1: Comparisons of unit status by rescaling the values in images or units

Images	CNN filters	Accuracy	L1-norm	APoZ	Class selectivity	**Feature entropy**
×	×	0.83	29.5	17.14%	0.58	1.87
Half scale	×	0.81	14.7	17.15%	0.31	1.90
×	Half scale	0.83	14.6	17.14%	0.30	1.87
Half scale	Half scale	0.79	7.2	17.16%	0.03	1.92

Table 1 shows the results where × denotes no rescaling operation for the item. As half scaling the magnitude in input images or units, the performance of the model fluctuates slightly. We could find that APoZ and feature entropy vary in the similar way with the performance, but L1-norm and class selectivity vary terribly. Apparently, despite little effect for the network, rescaling operations would have a major impact on these magnitude-based indicators, like L1-norm and class selectivity. These indicators fail to give accurate and stable measure of unit status especially when facing images or units with different value scales.

Table 2: Comparisons of unit status with respect to well-trained units and random units

	L1-norm	APoZ	Class selectivity	**Feature entropy**
Well-trained unit	29.5	17.14%	0.58	1.87
Random initialized units	32.2(30.9)	41%(40%)	0.01(0.003)	2.87(0.21), 0.83(0.22)

Detecting randomness in units Next, we compare the status of this unit with random units (units yielded by random-initialized models). Table 2 presents the results. The random units are sampled 100 times and the presented results are averaged over the 100 samples where the value in the brackets denotes the standard deviation. Since random units are clearly incapable to well perceive features like those trained units, they are expected to be indicated as ineffective units. We could see that when using L1-norm and APoZ indicators, they are impossible to give a stable indication as the standard deviation is extremely large. In some samples, the random units are judged as much "better" than the trained units, which is obviously incorrect. Accordingly, it could be also misleading using APoZ as the indicator of unit status. In contrast, the feature entropy would consistently be very high when random pattern exists in the unit, providing a well discrimination between trained units and random ones.

4.4 Using feature entropy to indicate networks with different generalization

In general, due to the large hypothesis space, CNNs could converge to a variety of minima on the dataset. Since feature entropy could reliably indicate the status of network units, it is natural to use it to discriminate which minima could provide more effective feature representations.

Models In this subsection, we prepare two sets of VGG16 models. Model set A consists of four models trained from scratch with different hyperparameters on ImageNet dataset, and Model set B consists of five models trained from scratch to almost zero training error with the same hyperparameters but on ImageNet dataset with different fractions of randomly corrupted labels as introduced in Zhang et al. (2017). Table 3 and 4 separately show the performance of models in the two model sets. In model set B, we use Model AD in Model set A as the first model Model BA with no corruption rate. Besides, it should be noted that all the calculation in this section is based on the image sampled from the training dataset.

Table 3: Model set A

	Train Acc	Test Acc
Model AA	0.732	0.657
Model AB	0.818	0.532
Model AC	0.828	0.444
Model AD	0.996	0.378

Table 4: Model set B

	Train Acc	Test Acc	Corrupted
Model BB	0.992	0.297	0.2
Model BC	0.994	0.166	0.4
Model BD	0.992	0.074	0.6
Model BE	0.993	0.010	0.8

Model set A Using the same image set in previous section, we start by investigating the feature entropy of units at different layers in the four models. Here, we still use the averaged feature entropy across all the units within a layer to indicate the overall level of how well the units in this layer could perceive the features. Fig.6A(1-2) shows the results of this class. We could see in the figure that there would not be significant difference of feature entropy between these models in layers except for the last convolutional layer. And in the last convolutional layer, for models with better generalization, their feature entropy would be lower than those with poor generalization, indicating that they would provide more effective feature representations. Besides, as for the selective rate, the four models are quite close.

Then, we randomly choose 100 classes in the ImageNet dataset and calculate the feature entropy of the units in the last convolutional layer. Fig.6A(3) presents the scatter plot for the four models, where each point stands for the feature entropy and selective rate of a specific class. For each model, its points locate at an area separately from other models, giving a discrimination between models. Also similarly, models with better generalization have points with lower feature entropy.

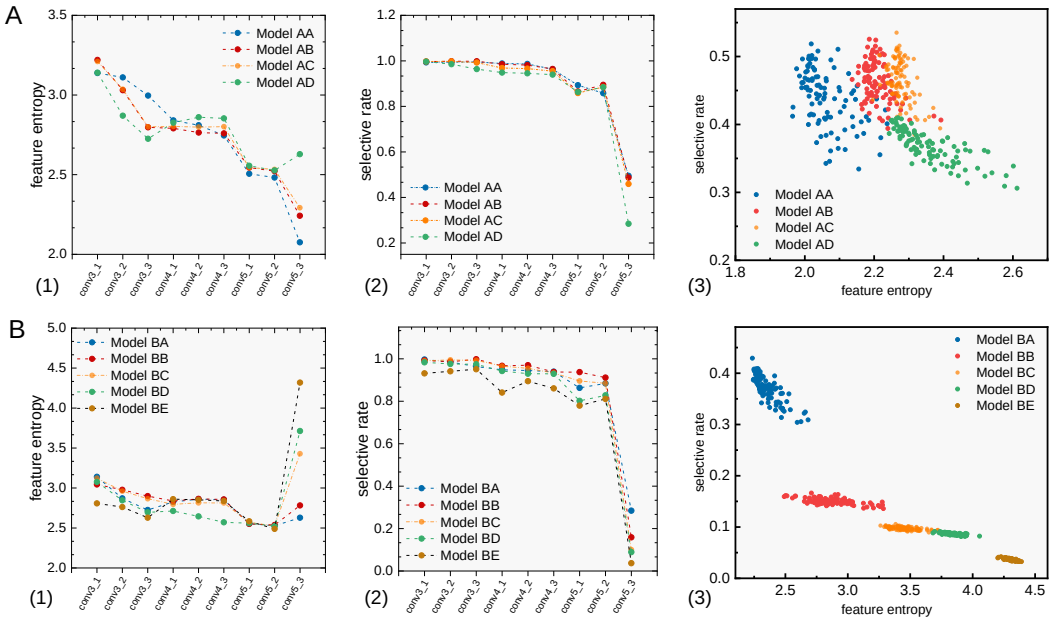

Figure 6: Comparisons between models in separately model set A (A) and model set B (B). Compare the feature entropy (1) and selective rate (2) of units at different layers between models in the corresponding model set on the exampled class. (3) shows the scatter plot between feature entropy and selective rate of units at the last convolutional layer on the 100 sampled classes.

Model set B For model set B, we use the same implementation as applied previously in the model set A, where the results are shown in Fig.6B. Comparing to the Model set A, since using the partially corrupted labels, units in the Model Set B are unable to perceive the common features between samples in the same class, which causes that the selective rate of most units are extremely low as shown in Fig.6B(2). Due to such low selective rate, we could also find in Fig.6B(1) that feature entropy of the units in the last convolutional layer may abruptly reach to a very high point. The more fraction the labels are corrupted, the higher feature entropy the units are and in the meantime the lower the selective rate the units are. This could be observed as well in Fig.6B(3) where the 100 classes are used for calculation.

5 Conclusion

We propose a novel method that could give quantitative identification of individual unit status, called feature entropy, for a specific class using algebraic topological tools. We show that feature entropy is a reliable indicator of unit status that could well cope with various cases such as rescaling values or existence of randomness. Also we show that feature entropy behaves in the similar way as loss during the training stage and presents a descending trend as convolutional layers go deeper. Using feature entropy, we show that CNNs with different generalization could be discriminated by the effectiveness of feature representations of the units in the last convolutional layer. We suppose this would be helpful for further understanding the mechanism of convolutional neural networks.

References

Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, and Lori Ziegelmeier. Persistence images: A stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18, 2017.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, 2020.

Peter Bubenik et al. Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.*, 16(1):77–102, 2015.

Rickard Brüel Gabrielsson and Gunnar Carlsson. Exposition and interpretation of the topology of neural networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 1069–1076. IEEE, 2019.

Robert Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

Petri Giovanni, Scolamiero Martina, Donato Irene, and Vaccarino Francesco. Topological strata of weighted complex networks. *PLOS ONE*, 8(6):1–9, 2013.

William H Guss and Ruslan Salakhutdinov. On characterizing the capacity of neural networks using algebraic topology. *arXiv preprint arXiv:1802.04443*, 2018.

Hatcher. *Algebraic Topology*. Cambridge University Press, London, 2002.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2980–2988. IEEE Computer Society, 2017a. doi: 10.1109/ICCV.2017.322.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397, 2017b.

Christoph Hofer, Roland Kwitt, Marc Niethammer, and Andreas Uhl. Deep learning with topological signatures. In *Advances in neural information processing systems*, pp. 1634–1644, 2017.

Danijela Horak, Slobodan Maletić, and Milan Rajković. Persistent homology of complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03034, 2009.

Tomasz Kaczynski, Konstantin Michael Mischaikow, and Marian Mrozek. *Computational homology*, volume 3. Springer, 2004.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *Iternational Conference on Learning Representations*, 2016.

Yuchao Li, Shaohui Lin, Baochang Zhang, Jianzhuang Liu, David Doermann, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Exploiting kernel sparsity and entropy for interpretable cnn compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2800–2809, 2019.

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pp. 5058–5066, 2017.

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5188–5196, 2015.

Aktas Mehmet, Akbas Esra, and El Fatmaoui Ahmed. Persistence homology of networks: methods and applications. *Applied Network Science*, 4(61):1–28, 2019.

Guido Montúfar, Nina Otter, and Yuguang Wang. Can neural networks learn persistent homology features? *arXiv preprint arXiv:2011.14688*, 2020.

Ari S Morcos, David GT Barrett, Neil C Rabinowitz, and Matthew Botvinick. On the importance of single directions for generalization. In *Iternational Conference on Learning Representations*, 2018.

Gregory Naitzat, Andrey Zhitnikov, and Lek-Heng Lim. Topology of deep neural networks. *arXiv preprint arXiv:2004.06093*, 2020.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing*, pp. 2422–2430, 2015.

Otter Ninna, A Porter Mason, Tillmann Ulrick, Grindrod Peter, and A Harrington Heather. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(17):1–38, 2017.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

Bastian Rieck, Matteo Togninalli, Christian Bock, Michael Moor, Max Horn, Thomas Gumbsch, and Karsten Borgwardt. Neural persistence: A complexity measure for deep neural networks using algebraic topology. *arXiv preprint arXiv:1812.09764*, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2818–2826. IEEE Computer Society, 2016.

Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8684–8694, 2020.

Jaehong Yoon and Sung Ju Hwang. Combined group and exclusive sparsity for deep neural networks. In *International Conference on Machine Learning*, pp. 3958–3966, 2017.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017*, 2017.

Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8827–8836, 2018.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

Bolei Zhou, David Bau, Aude Oliva, and Antonio Torralba. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*, 41(9): 2131–2145, 2018a.

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Revisiting the importance of individual units in cnns via ablation. *arXiv preprint arXiv:1806.02891*, 2018b.

A APPENDIX

A.1 IMPLEMENTATION DETAILS OF MODELS

In our experiments, the networks we used are the standard VGG16 architecture. We simply resize all the mentioned sample images to 224×224 without implementing any data augment (like random crop) during all the experiments (including the feature entropy calculation and the reported performance). The model used for demonstration in the experiment section (besides the last subsection) is Model AA in model set A.

For model set A, we use the following implementations,

- Model AA. The hyper-parameters are the same with them in the paper Simonyan & Zisserman (2014).
- Model AB. The hyper-parameters are the same with them in Model AA, except for changing the momentum to 0 and without using the data augmentation strategy.
- Model AC. The hyper-parameters are the same with them in Model AB, except for that only the first fully connected layer use the dropout with the rate of 0.3.
- Model AD. None of the conventional training enhancement technique is applied. Basically, It is Model AC without using dropout and l2 regularization.

For model set B, we use the following implementations,

- Model BA. It is actually the Model AD.
- Model BB. The hyper-parameters are the same with them in Model BA, except for corrupting the labels with 0.2 fraction.
- Model BC. The hyper-parameters are the same with them in Model BA, except for corrupting the labels with 0.4 fraction.
- Model BD. The hyper-parameters are the same with them in Model BA, except for corrupting the labels with 0.6 fraction.
- Model BE. The hyper-parameters are the same with them in Model BA, except for corrupting the labels with 0.8 fraction.

A.2 SPATIAL CHARACTERIZATION USING CUBICAL COMPLEX

In our method, a unit is firstly convert to a set of graphs and then each graph could be converted to the corresponding clique complex. In this way, a unit is characterized by the filtration of clique complexes. Alternatively, the unit could be directly modeled as a cubical complex Kaczynski et al. (2004), and then use the similar sublevel set implementation to acquire the filtration of cubical complexes. So here, we use cubical complex instead of clique complex and meanwhile keep the rest of the methods unchanged.

By investigating the birth distribution on the classes in the Imagenet with Model AA, we found that for every unit especially at relatively deeper layers, almost 95% images would have no birth point. In other words, no persistence homology emerges for almost all the units when using cubical complex. Thus, it is impossible to give a further calculation of the stability for images in the same class.

A.3 DISCUSSIONS OF USING BIRTH POINT

When characterizing the filtration of topological complexes, we use the birth time of the filtration. Comparing to some other characterizations, we suppose that the advantages of using birth point may lie in,

- Birth point is very easy to compute. In practice, when using birth point, it is not necessary to go through the whole filtration process, where we could stop the calculation when the birth point emerges. In this way, we could largely save the computation especially when the size of units is very high. This could be very helpful for calculating the large amount of units in neural networks.

- Birth point is very convenient for the investigation of stability between samples because it is an integer essentially. So, the discrete distribution could be set up effortlessly. Besides, when using entropy to investigate the stability of birth points, the results are bounded strictly in the range from 0 to $\log N$.
- Birth point is suitable for the idea. Since the significant features are generally represented by the high activation values in the units, it is more expected that the regularized pattern could be formed by these values. So it is natural to focus on these high activation values, which leads to the use of birth point.