# Evolutionary Visual Analysis of Deep Neural Networks

**Wen Zhong** [1] **Cong Xie** [1] **Yuan Zhong** [2] **Yang Wang** [1] **Wei Xu** [3] **Shenghui Cheng** [1] **Klaus Mueller** [1]

## Abstract

Visualization has shown great promise to further the understanding how deep neural networks function and operate. However, less focus has been placed on visualizing the training process of these models. This paper attempts to bridge this gap. We define two metrics by which the progress of the learning can be a quantitatively assessed. The first is a *discriminability* metric which evaluates the neuron evolution, while the second is a *density* metric which evaluates the output feature maps. Based on these metrics, a level-of-detail visual analytics framework is established that measures the evolution of the deep neural network both on a local and on a global scale. We demonstrate the efficacy of our system by ways of two real world case studies.

## 1. Introduction

With the rather explosive development of deep learning techniques (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014), visualization has emerged as an effective way to understand how deep neural networks work and operate. These efforts have come from the deep learning community as well as from the visual analytics community. The former has focused largely on visualizing the features learned by the network, such as speech features in NLP (Karpathy et al., 2015; Bahdanau et al., 2014) and image features in computer vision (Zeiler & Fergus, 2014; Simonyan et al., 2013; Krizhevsky et al., 2012). On the other hand, the visual analytics community has primarily sought to visualize the network and its connectivity, such as connection weights(Liu et al., 2016; Smilkov et al.; Harley, 2015) and hidden state patterns (Strobelt et al., 2016; Kahng et al., 2017).

Conversely, much fewer efforts, if any, have gone into devising effective visualization techniques to monitor the network training process and the network's evolution. Since proper training is a crucial prerequisite for a network's effective operation, this is clearly a critical gap. This paper seeks to make progress in this direction. By examining the drawbacks of current works, we elucidate the following three important capabilities a visual analytics framework for network evolution monitoring needs to possess:

1) **Monitor neural network evolution**: Understanding and visualizing how a deep neural network evolves can provide tremendous insights on how it works.
2) **Rigorous quantitative evaluation**: An effective assessment and monitoring requires effective quantitative metrics that can accurately evaluate the neural network training phase.
3) **Expand theoretical insights**: Theoretical insights into deep learning techniques (e.g. batch normalization) and phenomena (e.g. overfitting) are required due to their significance to deep model design and refinement.

We make use of these observations in our new framework. *Deep View (DV)*, a scalable level-of-detail visualization system for deep learning visualization (Fig. 1). DV builds upon two newly defined quantitative metrics, *discriminability* and *density* defined to enable an insightful layer and neuron evaluation. Based on these metrics, we uncover the evolution of deep neural networks on three granularities or levels of scale – network (macroscopic), layer (mesoscopic) and neuron (microscopic).

The remainder of this paper is structured as follows: Section 2 presents our proposed methodology. Section 3 describes the associated visualization framework we propose. Two real world case studies are examined in Section 4. Finally, Section 5 concludes the paper with a brief discussion.

## 2. Methodology

In this section, we propose the two new metrics, discriminability and density, that we use to quantitatively evaluate neurons and layers.

### 2.1. Discriminability Metric

In deep learning, the loss function is a frequently-used metric for the class-wise discriminability evaluation of the neurons in the final layer. As for the neurons in the inner layers,

[1]Stony Brook University, NY, USA [2]Northeastern University, MA, USA [3]Brookhaven National Lab, NY, USA. Correspondence to: Wen Zhong <wezzhong@cs.stonybrook.edu>.

*Submitted to 34th ICML Workshop on Visualization for Deep Learning*

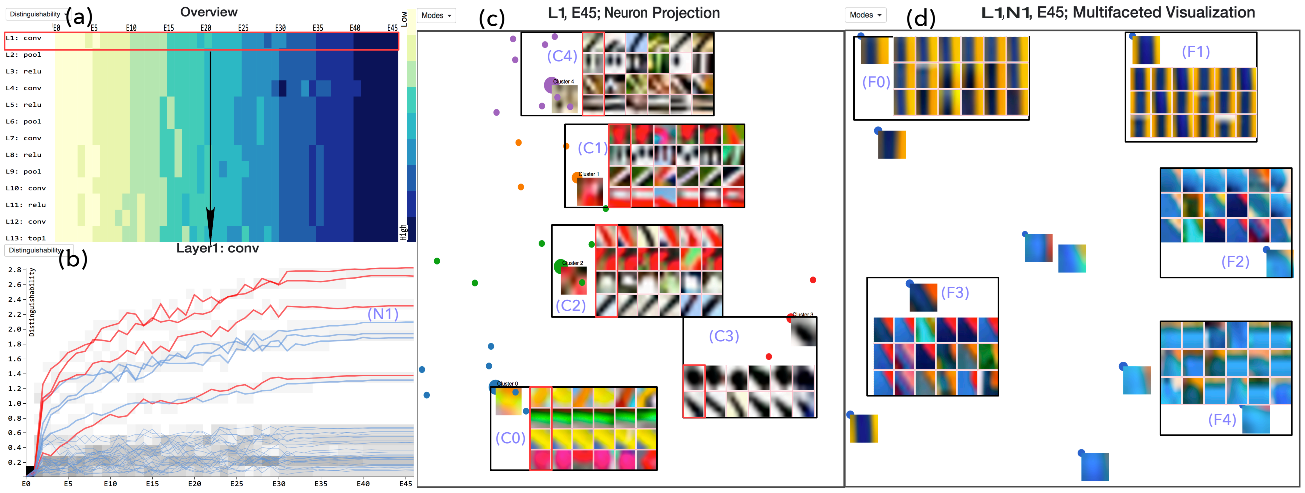


*Fig. 1.* The interface of our *DV* system for *lenet* visualization. Network discriminability overview (a), is encoded using heatmap across layers (y axis) and epochs (x axis). After selecting a specific layer (layer1), the discriminability overview (b) is shown as a heatmap embedded with line chart. Each line represents one neuron discriminability evolution and exceptional (good or bad) neurons are indicated using red lines. Besides, layer1 neuron features (c) are clustered and projected into 2D space to preserve similarity. Here, a weighted average feature method is used to enable hierarchical neuron cluster feature exploration. Multifaceted features of the $1^{st}$ neuron (d) in layer1 These features are obtained through the cluster of feature collections and presented using weighted average image to summarize main patterns. They are projected into 2D space for locality preservation.

due to the lack of ground truth for specific classes or visual concepts, it is challenging to quantitatively evaluate them. However, we observe that in most cases, neurons start from a random even distribution but then converge to a specific distribution steadily. This observation inspired us to devise a distribution distance-based metric geared to describe the discriminability of the inner layer neurons. Specifically, for a neuron $n_p$, we take the average of all class pair distances to assess their discriminability:

$$D(n_p) = avg\left[\sum_{c_i,c_j=0}^{m-1} dist\left(\theta_{c_i}^{n_p}, \theta_{c_j}^{n_p}\right)\right] \tag{1}$$

where $(c_i, c_j)$ is a class pair, $m$ is the number of classes, and $\theta_{c_i}^{n_p}$ is the activation distribution of neuron $n_p$ over class $c_i$. In terms of computation, $\theta_{c_i}^{n_p}$ is approximated from average activation samples $O_{c_i}^{n_p} \in \mathbb{R}^{w\times h\times \#c_i}$, where $w$ and $h$ are the feature map width and height, and $\#c_i$ is the number of input images for class $c_i$.

As seen from equation (1), discriminability heavily relies on the distance metric. Among multifarious candidates, most divergence based metrics (e.g. KL-divergence, Jensen-Shannon divergence and Hellinger distance) are ineligible since they fail when two distributions have no overlap. Inspired by Arjovsky et al. (2017), the $2^{nd}$ *Wasserstein distance*, aka earth mover's distance, fits well here:

$$W_2(\alpha,\beta) = \sqrt{\sum_p(\alpha_p-\beta_p)^2}$$

where $\alpha_p$ and $\beta_p$ are $p^{th}$ distribution samples.

## 2.2. Density Metric

In CNN structures, rectified linear unit (ReLU) activation, which usually succeeds each convolutional (CONV) layer, cuts negative values and preserves positive values to achieve representation sparseness. Nonetheless, output that is too dense or sparse indicates pathological deep model training. Therefore we define the denseness of positive activations for a neuron as *density*. Given neuron $n_p$, the density is formulated as:

$$DN\left(n_p\right) = avg\left[g\left(\underset{axis=w,h}{avg}\left(O^{n_p}\right)\right)\right] \tag{2}$$

where $O^{n_p} \in \mathbb{R}^{w\times h\times\sum_{i=0}^{m-1}c_i}$ is neuron $n_p$'s activation while $g\left(\cdot\right)$ is an activation mapping function that sets negative values to zero, positive values larger than one to one, and preserves the remaining values.

Density directly reveals the condition of the neuron learning process and assists in detecting potential training problems. For instance, overfitting, a common symptom in deep learning, can only be diagnosed based on loss curves. With the density metric, instead of waiting for overfitting to happen in the final layer, we can "infer" it by observing neuron density in the early training phase.

## 3. Deep View Visualizations

As mentioned, our *DV* system operates at three granularities of focus: macroscopic, mesoscopic, and microscopic. In the onset, the user first obtains a general network overview, which forms the macroscopic scale. Here, layers may pop out that peek the user's interest. They can be further inspected in a layer visualization that displays the evolution of neurons with averaged features – the mesoscopic scale. Finally, the visual analysis of specific neuron multifaceted features occurs at the microscopic scale.

### 3.1. Macroscopic Scale: Network Overview

The network overview summarizes the evolution of layers along training epochs using a heatmap, In Fig. 2(a), each row represents one layer over different training epochs and each column represents one epoch over different layers. The color scheme can encode either discriminability or density levels. Since one layer consists of many neurons, layer discriminability and density is obtained by averaging over neurons. The final layer shows the loss function and the top 1 or 5 training/validation errors.

### 3.2. Mesoscopic Scale: Layer Visualization

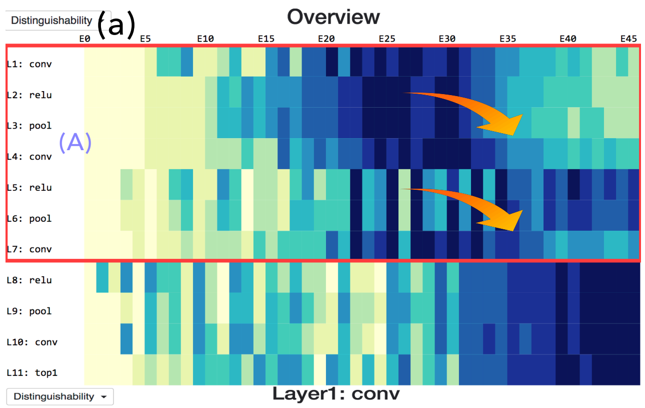
Having gained a high level network overview, the layer visualization allows users to dig in and inspect specific layers of interest for a detailed analysis. The layer visualization provides users with the opportunity to understand the layer-wise neurons in relation to all neurons (in terms of discriminability or density) and the neuron's learned features.

#### 3.2.1. Layer Overview

To visualize the evolution of neurons in a specific layer, we chose a contextual visualization where a heatmap shows the neuron overview and the lines show the neuron evolution over epochs. In Fig. 2(b), the $x$ axis indicates epochs and the $y$ axis indicates discriminability levels. For better visibility of the main patterns, we discretize the layer overview panel epoch-wise into buckets (e.g. 45 epochs $\times$ 20 levels) where the percentage of neurons in each bucket is visualized by grid opacity. The darker a grid cell is, the higher the percentage it represents. In the line chart, each line represents the evolution of one neuron. When the number of neurons grows large, in order to avoid clutter,. only exceptional neurons are shown using red lines.

#### 3.2.2. Layer Feature

A shortcoming of the present layer-based neuron visualization is that the corresponding learned features cannot be discerned. To remedy this, we employ the receptive field (RF) based neuron feature extraction method (Girshick et al., 2014) to extract the top $k$ activated patches each



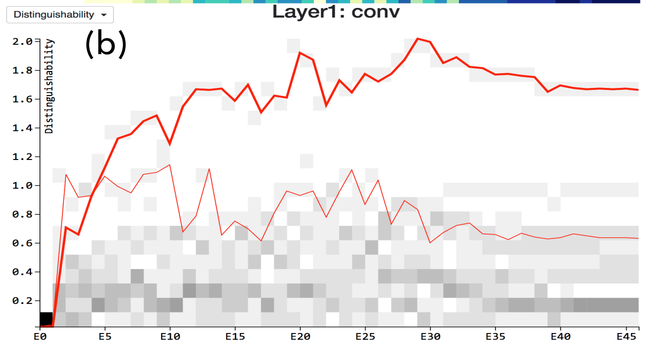


*Fig. 2.* (a): Discriminability overview of *shallow*. (b): Layer1 discriminability overview of *Shallow*. Only exceptional neurons are shown using red lines.

neuron is sensitive to. However, two challenges persist: 1) the impossibility of visualizing all neurons, 2) the difficulty of exploring all facets, even for a single neuron.

We solve the first challenge through neuron clustering. We exploit the distribution distance of class-wise neuron pairs, where the distance of neuron pair $(n_p, n_q)$ is defined as:

$$N_{dis}(n_p, n_q) = \sum_{i=0}^{m-1} W_2\left(\theta_{c_i}^{n_p}, \theta_{c_i}^{n_q}\right) \tag{3}$$

Via this equation we obtain a neuron distance matrix and adopt agglomerative clustering and multidimensional scaling (MDS) to hierarchically cluster and project neurons into 2D space. By default, we set the number of clusters to five, but the the user can change this interactively.

To address the second challenge, we are inspired by the average image explorer of Zhu et al. (Zhu et al., 2014), which effectively extracts informative patterns and filters out irrelevant noise. A clear average image will be indicative of well-regulated features in a pure neuron with good quality. On the other hand,. a messy average image will exhibit a chaotic pattern, indicating an impure neuron with bad quality. We apply this weighted average methodology to our hierarchical neuron feature exploration. Here, we treat the activation level for each neuron as a weight and average the levels over the top $k_0$ ($k_0 = 20$) features to represent the highest activated pattern. For a neuron cluster, we treat the inverse of the distance from each neuron to the cluster center as a weight and average the corresponding average features of the $k_1$ ($k_1 = 4$) nearest neurons.

The integration of MDS, agglomerative clustering, and weighted average image techniques provides a clear layer-wise neuron feature visualization (Fig. 3(a)). The color of the nodes encodes the cluster groups and the radius of the center node encodes the size of a cluster. For each neuron cluster, we visualize its average as a single image attached to the cluster center inside the black rectangle. For each neuron inside a cluster, the neuron average image is shown in the leftmost red column followed by the top 5 highest activated patterns.

### 3.3. Microscopic Scale: Neuron Multifaceted Feature Visualization

When interrogating the features learned by the neuron, typically the most attention only goes to the highest activated feature. But a neuron learns multifaceted feature patterns (Nguyen et al., 2016), of which the highest activated feature is a part of. As deep neural networks are highly non-linear with complex weight connections, these multiple aspects all play key roles in the understanding of neurons and should be considered. Unfortunately, previous works do not realize these complex interactions and so lead to a biased and incomplete view of the neurons.

Similar to (Nguyen et al., 2016), we apply K-means, t-SNE (Maaten & Hinton, 2008) and image weighted average methods to uncover multifaceted features. The procedure we use to extend the layer feature visualization was presented in Section 3.2.2. First, a range of features (e.g. the top 1,000 highest activated features) are extracted and clustered into groups. Second, the clusters and centroids are projected into 2D space using t-SNE. Third, for each feature cluster, the 20 nearest images are weight-averaged for representation. As shown in Fig. 3(b), each node in the average feature represents one feature facet. A current limitation is that it is hard to detect the number of multifaceted features. Unsupervised clustering is a good direction, however, tuning their hyper-parameters (e.g. the leaf size in DBSCAN) is bothersome. We decided to use K-means with 10 clusters by default. Users can change $k$ based on their preferences.

## 4. Case Study

To demonstrate the effectiveness of our *DV* system, we performed case studies with two domain experts (co-author 3 and 4) from two university research labs. The first case study (Section 4.1) shows that our system can promote deep neural network comprehension, reveal potential un-

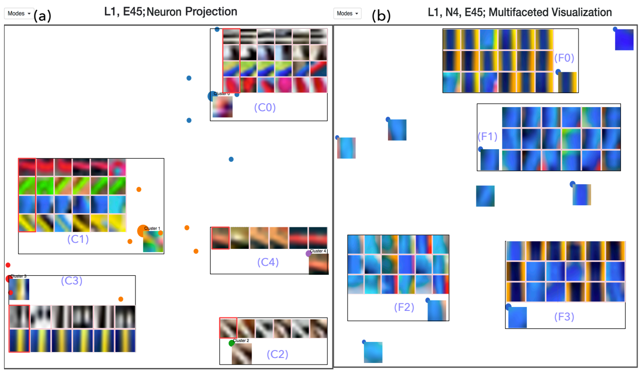
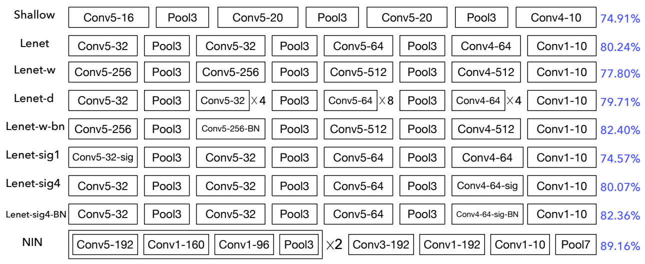

(a)
L1, E45;Neuron Projection
(C0)
(C1)
(C2)
(C3)
(C4)
(b)
L1, N4, E45; Multifaceted Visualization
(F0)
(F1)
(F2)
(F3)

*Fig. 3.* (a): Layer1 neuron features of *shallow*. (b): Multifaceted feature visualization of $4^{th}$ neuron in *shallow* layer1.

derfitting/overfitting and refine network structure. The second case study (Section 4.2) manifests that *DV* advances the understanding of state-of-the-art deep learning techniques.

For this evaluation we employ CIFAR10 (Krizhevsky & Hinton, 2009) which is a relatively small but difficult dataset. During the case study, the experts experienced *DV* with various deep neural networks. Fig. 4 shows related network structures during exploration.



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Shallow | Conv5-16 | Pool3 | Conv5-20 | Pool3 | Conv5-20 | Pool3 | Conv4-10 | | 74.91% |
| Lenet | Conv5-32 | Pool3 | Conv5-32 | Pool3 | Conv5-64 | Pool3 | Conv4-64 | Conv1-10 | 80.24% |
| Lenet-w | Conv5-256 | Pool3 | Conv5-256 | Pool3 | Conv5-512 | Pool3 | Conv4-512 | Conv1-10 | 77.80% |
| Lenet-d | Conv5-32 | Pool3 | Conv5-32 ×4 | Pool3 | Conv5-64 ×8 | Pool3 | Conv4-64 ×4 | Conv1-10 | 79.71% |
| Lenet-w-bn | Conv5-256 | Pool3 | Conv5-256-BN | Pool3 | Conv5-512 | Pool3 | Conv4-512 | Conv1-10 | 82.40% |
| Lenet-sig1 | Conv5-32-sig | Pool3 | Conv5-32 | Pool3 | Conv5-64 | Pool3 | Conv4-64 | Conv1-10 | 74.57% |
| Lenet-sig4 | Conv5-32 | Pool3 | Conv5-32 | Pool3 | Conv5-64 | Pool3 | Conv4-64-sig | Conv1-10 | 80.07% |
| Lenet-sig4-BN | Conv5-32 | Pool3 | Conv5-32 | Pool3 | Conv5-64 | Pool3 | Conv4-64-sig-BN | Conv1-10 | 82.36% |
| NIN | Conv5-192 | Conv1-160 | Conv1-96 | Pool3 ×2 | Conv3-192 | Conv1-192 | Conv1-10 | Pool7 | 89.16% |

*Fig. 4.* Network structures and their corresponding accuracies.

### 4.1. Case Study 1: Network Diagnosis and Refinement

Expert A ($E_A$) is a third year Ph.D. student. His research interest is data mining and general machine learning, and uses our system as a deep model practitioner focusing on task-specific networks. Recently, he designed attention based models for location prediction. He wanted to utilize our system to compare baseline deep models to better understand their inner strategies and improve their performance.

#### 4.1.1. Understanding Baseline Network

Adopting *lenet* (LeCun et al., 1998) as baseline, $E_A$ designed the *shallow* network (Fig. 4) with fewer layers and neurons to see the influence of deeper and wider structures.

The results obtained with *shallow* are shown in Fig. 2 and Fig. 3, and those of *lenet* are shown in Fig. 1. After exam-

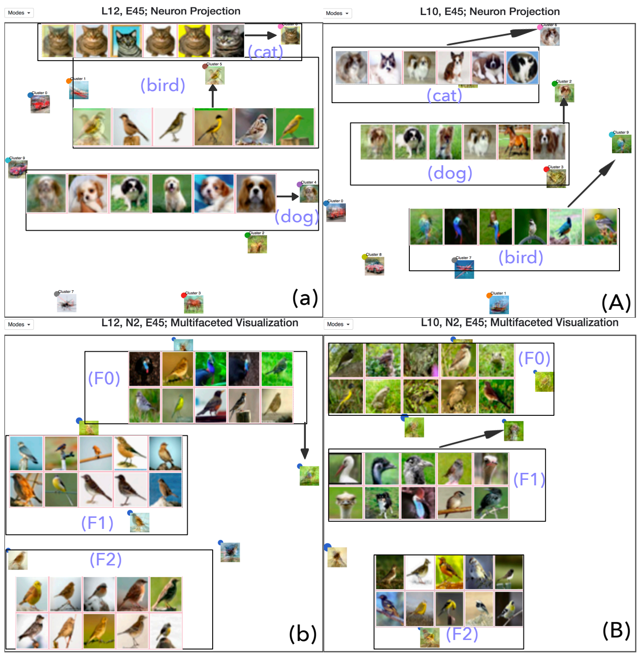

L12, E45; Neuron Projection
(cat)
(bird)
(dog)
(a)
L10, E45; Neuron Projection
(cat)
(dog)
(bird)
(A)
L12, N2, E45; Multifaceted Visualization
(F0)
(F1)
(F2)
(b)
L10, N2, E45; Multifaceted Visualization
(F0)
(F1)
(F2)
(B)

*Fig. 5.* (a) and (A): Final layer neuron features of *lenet* and *shallow*. Cat, dog and bird neurons are shown with their average feature and top 5 highest activated features. (b) and (B): Bird neuron multifaceted features of *lenet* and *shallow*. Three facet average features are shown in rectangles.

ining the discriminability overviews in Fig. 2(a) and Fig. 1(a), he found that both networks had a gradually increasing trend, however discriminability in *shallow* was not as stable as *lenet*. He also noticed that the layers in Fig. 2(a) (A) block had a decreasing discriminability trend after the 30th epoch. This was consistent with the training setting where he lowered the learning rate after the 30th epoch for fine tuning. From Fig. 1(b) and Fig. 2(b), $E_A$ also observed that the quality of the neurons was better in *lenet*. All of these visualizations prove *lenet*'s superiority over *shallow*.

**Width factor**: For a specific layer, $E_A$ inspected neuron and neuron cluster learned features from *lenet* and *shallow* in Fig. 1(c) and Fig. 3(a). From observations, he knew that for layer1, *lenet* and *shallow* learned the same type of features. However, careful scrutiny reveals that *lenet* employing more diverse features, indicating *shallow* network was under fitting.

In addition, $E_A$ focused on neuron multifaceted feature visualization for detailed analysis. He found that two neurons, the 1st neuron in layer1 of *lenet* and the 4th neuron in layer1 of *shallow*, had similar feature facets. The corresponding multifaceted features are visualized in Fig. 1(d) and Fig. 3(b), respectively. Both neurons mainly detected the same features: light blue edges and dark blue edges. This thorough analysis indicate that in those networks, layer1 neurons learned very similar things. However, due to the fewer number of neurons, *shallow* underfits.

**Depth factor**: In Fig. 5(a) and 5(A), through comparing final layer features, $E_A$ noticed *lenet*'s cat, dog and bird neuron average features are clearer than those of *shallow*. This is obvious, since in *shallow*, the $1^{st}$, $2^{nd}$ and $3^{rd}$ features of cat neuron were dogs and the $4^{th}$ feature of dog neuron was horse. Although bird neuron seems pure (all the top 5 are birds), further scrutiny on multifaceted features (Fig. 5(b)) reveals that it mostly concentrates on the discriminative part of bird without fully considering the background. This leads to a mostly average feature background. In view of this, $E_A$ concluded that network depth (capturing higher level features) is one major advantage of *lenet*.

### 4.1.2. Diagnosing Deep and Wide Networks

After visualizing simple networks, $lenet$ and $shallow$, $E_A$ attempted much deeper and wider networks, namely *lenet-d* and *lenet-w* (shown in Fig. 4).

**Wider network *lenet-w*:** From Fig. 6, $E_A$ observed a mild discriminability degeneration after the $1^{st}$ epoch (a), and the network learned representations that were too sparse (b). It is well known that a sparse representation is beneficial since it reduces noise and outliers, but a too sparse representation is not a good signal.

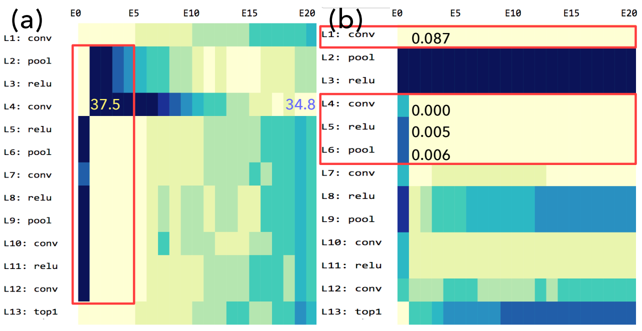


*Fig. 6.* The overviews of *lenet-w* discriminability (a) and density (b). The digits are density values.

$E_A$ concluded by conjecture that: *lenet-w* is prone to overfitting. His reasons came from two observations: 1) Neuron sparseness: the layer evolution overview shows that layer4 (CONV, Fig. 7(a)) and layer5 (ReLU, 7(b)) neurons have very minor discriminability. 2) Scant feature diversity: the features learned by the neurons of layer5 (Fig. 7(c)) show that the dominant cluster (C1) was composed of dead neurons and that the remaining neuron clusters learned the same feature: colorful background with bars inside. In fact, *lenet-w* overfitted after the $40^{th}$ epoch and achieved only 77.8% accuracy, verifying $E_A$'s conclusion.

**Deep network *lenet-d*:** Using the network discriminability overview (Fig. 8(a)), $E_A$ found that layers in the red block

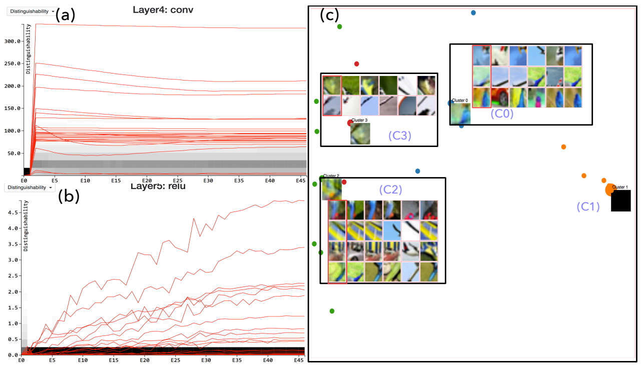


*Fig. 7.* (a) and (b): Layer4 (CONV) and layer5 (ReLU) discriminability evolution overviews of *lenet-w*. Most neurons keep steady (dead). (c): Layer5 (CONV) neuron learned features. C1 is the dead neuron cluster.

suffered from unstable optimization. The reason might be a problem with vanishing/explosive gradients accompanying the deep structure. With the help of the visualization shown in Fig. 8(b) he noticed that most CONV (with arrows) and ReLU layers learned dense representations (large density). These dense representations are undesirable owing to the sensitivity to noise and tendency towards overfitting.

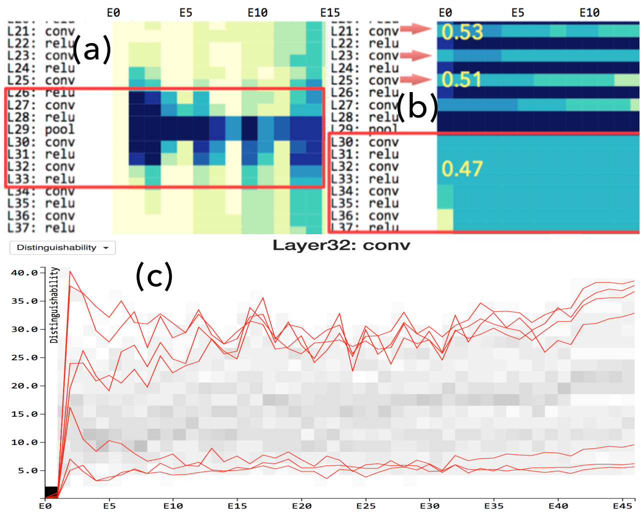


*Fig. 8.* The overviews of *Lenet-d* discriminability (a) and density (b). The digits are density values.

After training terminated, $E_A$ checked the *lenet-d* layer overview and the learned features to see what happened inside the network. He discovered that the inner layer (from the $27^{th}$ epoch to the $33^{th}$ epoch) the discriminability stayed unchanged, indicating a poor evolution trend (Fig. 8(c)). Besides, from Fig. 9, he observed that the neurons had already learned to detect class-wise concepts (truck/car and ship/dog). Hence, it seems that the network was premature (learning complex features early), i.e. former layers are already advanced enough rendering later layers unnecessary. This added model complexity and a risk for overfitting. After making these discoveries, to further check neuron quality, $E_A$ looked into the neuron multi-faceted features shown in Fig. 10, where he found that there were many pure facets in addition to the highest activated patterns. From this, $E_A$ confirmed for sure that *lenet-d* was premature. He initially thought that this was because of the $3 \times 3$ filter size, RF increased too fast and captured high level features too early. But actually, *lenet-d* became overfitting after the $28^{th}$ epoch and the prediction accuracy was only 79.7%.

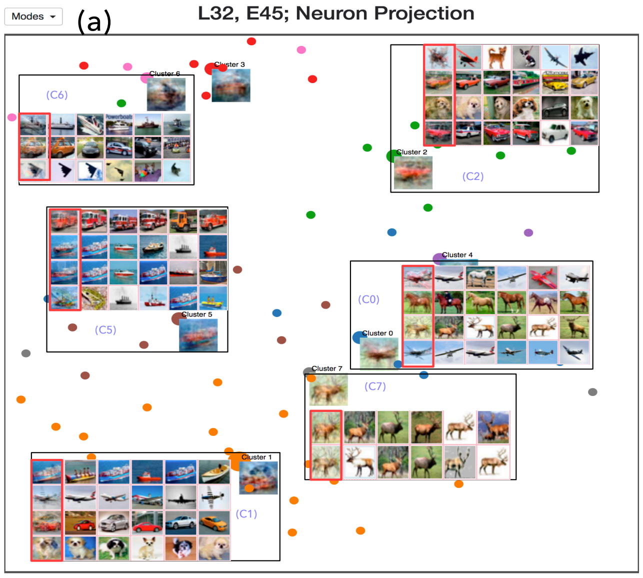


*Fig. 9.* Neuron features of *lenet-d* layer32.

Based on the above analysis, $E_A$ came to know that moderate depth, width, and RF size are all vital to network performance. To validate, he applied the network in network (*NIN*) (Lin et al., 2013) structure with a self-designed $1 \times 1$ filter to restrict the increase of RF. Surely, *NIN* exhibited increased layers' discriminability and moderate density.

## 4.2. Case Study 2: State-of-the-Art Deep Learning Techniques Understanding

Expert B ($E_B$), is a fourth year computer vision Ph.D. student, focusing on video analysis for action recognition. He is also interested in designing new deep learning techniques (e.g. better pooling strategy), so he cares more about understanding the mechanism underneath deep models and state of the art methodologies.

### 4.2.1. Batch Normalization Understanding

$E_B$ is curious about batch normalization (BN), a simple yet effective deep learning technique. He knows BN mainly tries to normalize the CONV layer output to increase CNN performance. But he is curious about why BN is so robust to large learning rates and careless initialization. $E_B$

adopted our system to resolve these questions in practice.

Starting from simple neural networks, $E_B$ added BN layers after every CONV layer of *lenet*, *shallow* and *NIN* networks and labeled them *lenet-bn*, *shallow-bn* and *NIN-bn* networks. He first checked the discriminability overviews and found that some layers exhibited exceptional patterns (Fig. 11(a), (b) and (c)). Normally, most layers of a well trained network will exhibit a general increasing trend in discriminability, but these selected layers experienced initially decreasing trends. It was difficult for $E_B$ to explain the reasons for these declines.

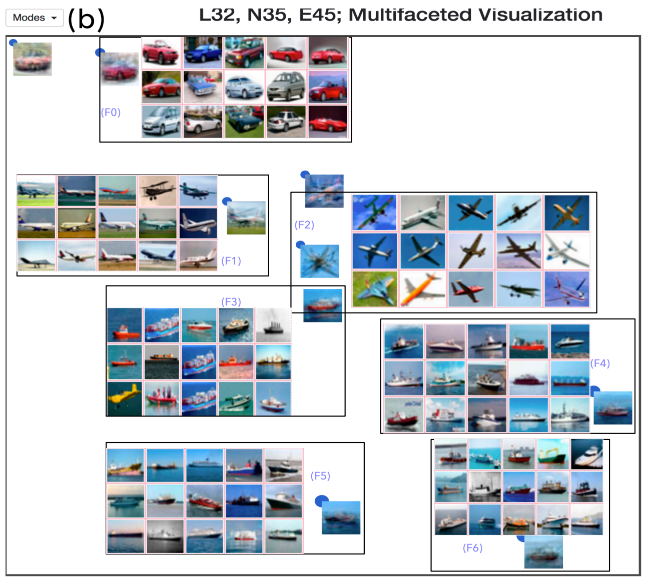


*Fig. 10.* Multifaceted feature visualization of 35th neuron in *lenet-d* layer32.

To gain more insight, he switched back to the network density overview (Fig. 11(A), (B) and (C)) where he learned that these layers' densities had the same evolutionary trends as the discriminabilities. This means that all these layers originally tried to learn sparse representations. This is interesting since it signifies that the behavior of discriminability and density are highly consistent after adding BN layers. He then conducted an experiment: Since *lenet-w* was overfitting due to learning a too sparse representation in layer4, he added a BN layer after layer4 to see the response. In Fig. 11(d) and (D), he noticed the consistency as well – the only difference was that this time layer4 gradually learned well populated representations to avoid overfitting. Considering these cases, BN seems to motivate sparsity/density networks and avoid overfitting through normalizing the ReLU input. As for the confusion of neuron discriminability (Fig. 11(a), (b) and (c)), the intuition is that the normalization and rescaling of BN may shrink the input range and fail the Wasserstein distance. After all, optimizing the network discriminability is different from optimizing a single neuron (Bau et al., 2017).

From the above analysis, $E_B$ concluded that BN tends to learn dense or sparse representations for better performance. Actually, in CNNs, BN interplays closely with ReLU. ReLU enforces the network to learn sparse representations by cutting off negative activations. But when a network tries to learn too sparse representations, BN would normalize input to larger values and guide ReLU to learn dense representations, while for the case of a too dense representation, BN would normalize ReLU input to smaller values and learn sparse representations.

### 4.2.2. Activation Function Understanding

$E_B$'s other interest is the activation function. Recently, he noticed that the activation function evolved from sigmoid to ReLU. He hoped to adopt *DV* for sigmoid interpretation.

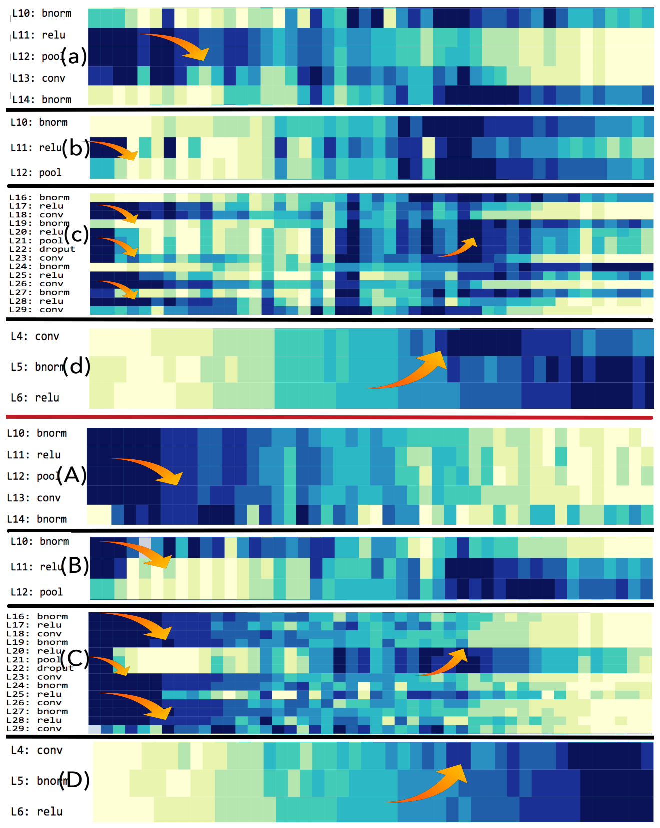


*Fig. 11.* Layers with exceptional discriminability patterns of *lenet-bn* (a), *shallow-bn* (b), *NIN-bn* (c) and *lenet-w-bn* (d). (A), (B), (C) and (D): The corresponding layer density patterns. The arrows indicate increase or decreaspatterns and heatmaps colors are encoded locally to visualize layer evolution.

$E_B$ trained two new networks, namely *lenet-sig1* and *lenet-sig4* (in Fig. 4), which replaced *lenet*'s 1st ReLU and 4st ReLU layers with sigmoid layers, respectively. From Fig. 4, *lenet-sig1* seems to have a large decrease, which means the saturation in *lenet-sig1* (layer4) appears much earlier than in the *lenet-sig4* (layer11). However, contrasting Fig. 12(a) and Fig. 1(a), he realized that this tiny influence really changed the network fine-tuning optimization pattern. He later examined the layer discriminability overviews and found layer10 and layer11 were very unusual. In Fig. 12(b), layer10 was very near to the final layer and should have strong rather than weak increasing trends. In Fig. 12(c), he noticed that layer11 evolved much too fast with only minor increased after the $3^{rd}$ epoch. Since layer11 was a sigmoid layer, a neuron's discriminability stopped increasing after becoming saturated.

Recalling that BN could normalize data, $E_B$ employed BN to alleviate sigmoid saturation. He trained *lenet-sig4-bn*, where a BN layer was inserted before the *lenet-sig4* sigmoid layer with over 2% accuracy increase. After checking the layer10 and layer12 discriminability evolutions in Fig. 12(B) and (C), he found that layer10 had stronger increasing trends, and that layer12 learned steadily and became saturated much later. This provides a good solution to solve the sigmoid saturation problem and validates the effectiveness of BN.

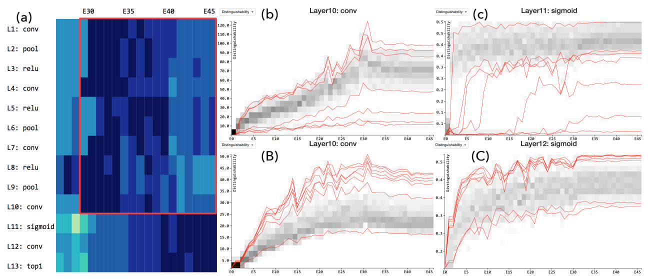


*Fig. 12.* (a): *Lenet-sig4* discriminability overview. Layers inside the red rectangle show tiny discriminability dropping down after $30^{th}$ epochs. (b) and (c): Layer10 and layer11 discriminability evolutions of *lenet-sig4*. (B) and (C): Layer10 and layer11 discriminability evolutions of *lenet-sig4-bn*.

## 5. Conclusion and Discussion

We presented, *DV*, a scalable visual analytics approach for deep neural network inspection in real-time, aiding understanding, diagnosis and refinement. To help in this task, we devised two new and useful quantitative metrics, *discriminability* and *density* for layer and neuron evaluation. We also designed a hierarchical exploration method based on weighted averages for a multifaceted neuron feature visualization. Based on two case studies, we show the correctness, effectiveness and efficiency of our system.

The bottleneck of *DV* is rooted in the limitations of the weighted average image method. There are two essential aspects: 1) This method could only be applied to vision datasets, while for NLP and speech datasets, a more universal method is required for feature exploration; 2) The average image may conceal some features. For example, bright colors will be neutralized by dark colors when combined together. We plan to investigate these aspects in future work, and conduct further case studies on other types of networks.

## Acknowledgements

This research was partially supported by NSF grant IIS 1527200 and LDRD grant 16-041 by Brookhaven National Lab.

## References

Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Bahdanau, Dzmitry, Cho, Kyunghyun, and Bengio, Yoshua. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bau, David, Zhou, Bolei, Khosla, Aditya, Oliva, Aude, and Torralba, Antonio. Network dissection: Quantifying interpretability of deep visual representations. *arXiv preprint arXiv:1704.05796*, 2017.

Girshick, Ross, Donahue, Jeff, Darrell, Trevor, and Malik, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

Harley, Adam W. An interactive node-link visualization of convolutional neural networks. In *International Symposium on Visual Computing*, pp. 867–877. Springer, 2015.

Kahng, Minsuk, Andrews, Pierre, Kalro, Aditya, and Chau, Duen Horng. Activis: Visual exploration of industry-scale deep neural network models. *arXiv preprint arXiv:1704.01942*, 2017.

Karpathy, Andrej, Johnson, Justin, and Fei-Fei, Li. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lin, Min, Chen, Qiang, and Yan, Shuicheng. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Liu, Mengchen, Shi, Jiaxin, Li, Zhen, Li, Chongxuan, Zhu, Jun, and Liu, Shixia. Towards better analysis of deep convolutional neural networks. *arXiv preprint arXiv:1604.07043*, 2016.

Maaten, Laurens van der and Hinton, Geoffrey. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

Nguyen, Anh, Yosinski, Jason, and Clune, Jeff. Multi-faceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616*, 2016.

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

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

Smilkov, Daniel, Carter, Shan, Sculley, D, Viégas, Fernanda B, and Wattenberg, Martin. Direct-manipulation visualization of deep networks.

Strobelt, Hendrik, Gehrmann, Sebastian, Huber, Bernd, Pfister, Hanspeter, and Rush, Alexander M. Visual analysis of hidden state dynamics in recurrent neural networks. *arXiv preprint arXiv:1606.07461*, 2016.

Zeiler, Matthew D and Fergus, Rob. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pp. 818–833. Springer, 2014.

Zhu, Jun-Yan, Lee, Yong Jae, and Efros, Alexei A. Averageexplorer: Interactive exploration and alignment of visual data collections. *ACM Transactions on Graphics (TOG)*, 33(4):160, 2014.