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

Recently, deep learning visualization gained a lot of attentions for understanding deep neural networks. However, there is a missing focus on the visualization of deep model training process. To bridge the gap, in this paper, we firstly define a *discriminability* metric to evaluate neuron evolution and a *density* metric to investigate output feature maps. Based on these metrics, a level-of-detail visual analytics framework is proposed to locally and globally inspect the evolution of deep neural networks. Finally, we demonstrate the effectiveness of our system with two real world case studies.

1. Introduction

With the explosive development of deep learning techniques (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014), deep learning visualization, an effective way to understand deep neural networks, is conceived as a significant research area. There have been efforts promoting the comprehension of deep neural networks, not only from deep learning community but also from visual analytics community. The former focuses on designing optimization based methods to visualize neuron learned features such as NLP, speech features (Karpathy et al., 2015; Bahdanau et al., 2014) and vision features (Zeiler & Fergus, 2014; Simonyan et al., 2013; Krizhevsky et al., 2012) while the latter seeks to incorporate visual analytics techniques to reveal neuron weight connections (Liu et al., 2016; Smilkov et al.; Harley, 2015) and hidden state patterns (Strobelt et al., 2016; Kahng et al., 2017).

However, the visualization of network training process, as an indispensable way to monitor the network evolution, has been largely overlooked by previous works. By examining the drawbacks of current works, we discover three important yet unfathomed aspects:

1) **Monitoring 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 quantitative metrics that can accurately evaluate the neural network training phase.
3) **Expanding 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.

In this work, we investigate and fill these gaps by proposing *Deep View (DV)*, a scalable level-of-detail visualization system for deep learning visualization. *DV* is leveraged by two quantitative metrics, *discriminability* and *density* for layer and neuron evaluation. Based on that, we uncover the evolution of deep neural networks w.r.t. three granularities, network (macroscopic), layer (mesoscopic) and neuron (microscopic).

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

2. Methodology

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

2.1. Discriminability Metric

In deep learning, loss function is a frequently-used metric for the class-wise discriminability evaluation of final layer neurons. As for inner layer neurons, due to lack of ground truth for specific classes or visual concepts, it is quite challenging to quantitatively evaluate them. However, we observe that in most cases, neurons start from a random even distribution and converge to a specific distribution steadily. This process inspires us to devise a distribution distance based metric to describe inner layer neuron discriminability. Specifically, for a neuron n_p, we take average of all the class pair distance as its discriminability:

[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 34^{th} ICML Workshop on Visualization for Deep Learning

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

where (c_i, c_j) is a class pair, m is the number of class, 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 Formula (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 from Arjovsky et al. (2017), 2^{nd} *Wasserstein distance*, a.k.a earth mover's distance fits well here:

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

where α_p and β_p are p^{th} distribution samples.

2.2. Density Metric

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

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

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

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

3. Deep View Visualizations

In this section, we introduce *DV* system in three granularities: macroscopic, mesoscopic and microscopic scales. Specifically, the user first obtains general network overviews in macroscopic scale. Then for interested layers, layer visualization displays neuron evolutions with average features in mesoscopic scale. To detailedly analyze a specific neuron, multifaceted features are shown in microscopic scale.

3.1. Macroscopic Scale: Network Overview

Network overview summarizes the evolution of layers along with training epochs using heatmaps. In Fig. 1(a), each row represents one layer over different training epochs and each column represents one epoch over different layers. The color scheme encodes discriminability or density levels. Since one layer consists of many neurons, layer discriminability and density could be obtained by averaging those of neurons'. For the final layer, we provide loss function, top 1 or 5 training/validation errors.

3.2. Mesoscopic Scale: Layer Visualization

Besides high level network overview exploration, a specific layer is indispensable for detailed analysis. Layer visualization provides an opportunity to understand layer-wise neurons in terms of neuron overview (discriminability or density) and neuron learned features.

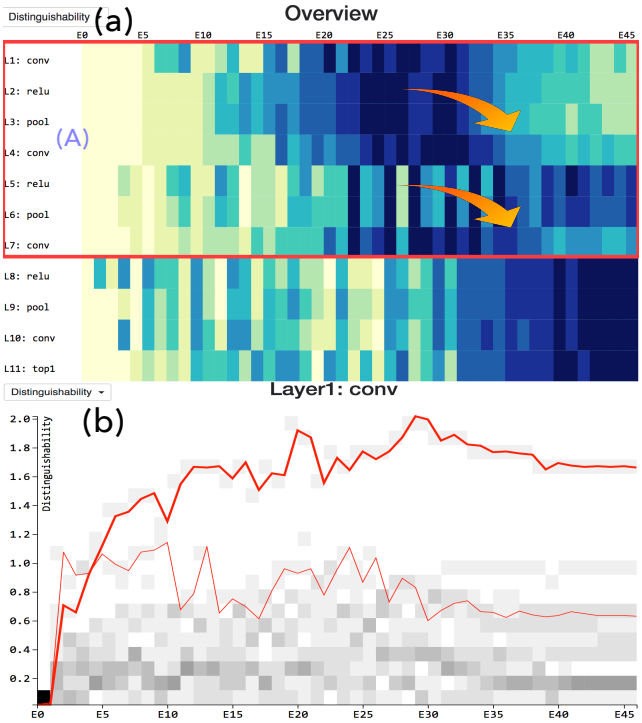

Fig. 1. (a): Discriminability overview of *shallow*. (b): Layer1 discriminability overview of *Shallow*. Only exceptional neurons are shown using red lines.

3.2.1. Layer Overview

To visualize neuron evolution of a specific layer, an elegant design: heatmap embedded with line chart is proposed, where heatmap shows neuron overview and lines show neuron evolution over epochs. In Fig. 1(b), x axis indicates epochs and y axis indicates discriminability levels. The layer overview panel is discretized into buckets (e.g. 45 epochs $\times$ 20 levels). In the heatmap, with epoch (column-wise) normalization, we denote the percentage of neurons in each bucket by grid opacity. The darker a grid is, the higher percentage it represents. In the line chart, each line presents one neuron evolution. When the number of neurons becomes large, only exceptional ones are shown using red lines to avoid visual clutter.

3.2.2. Layer Feature

The general overview access of the neurons in one layer is helpful, however, the corresponding learned features of neurons are completely abandoned. To remedy this, we employ receptive field (RF) based neuron feature extraction method (Girshick et al., 2014), where top k activated patches are extracted for each neuron. However, there are two challenges entangling feature visualization: 1) the impossibility of visualizing all neurons, 2) the difficulty of exploring all facets, even for a single neuron.

For the first challenge, we solve it through neuron clustering. The distribution distance of class-wise neuron pair is exploited, where the distance of neuron pair (n_p, n_q) is defined as:

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

With neuron distance matrix obtained from Formula (3), we adopt agglomerative clustering and multidimensional scaling (MDS) to hierarchically cluster and project neurons into 2D space. By default, we set the number of clusters five, and the user could change it dynamically.

For the second challenge, an elegant solution is average image explorer (Zhu et al., 2014), which effectively extracts informative patterns and filters irrelevant noise. Clear average image shows well-regulated features and indicates a pure neuron with good quality, while messy average image shows chaotic patterns and indicates an impure neuron with bad quality. Hence, we apply this weighted average methodology to hierarchical neuron feature exploration. Specifically, for a neuron, we treat the activation level as weight and average over top k_0 ($k_0 = 20$) features to represent the highest activated pattern. For a neuron cluster, we treat the inverse of distance from neuron to the cluster center as weight and average the corresponding average features of k_1 ($k_1 = 4$) nearest neurons.

The integration of MDS, agglomerative clustering and weighted average image techniques offers us a clear view of layer-wise neuron feature visualization (Fig. 2(a)). The color of nodes encodes clustering groups and center node radius encodes the size of a cluster. For each neuron cluster, the average image neuron cluster is the 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 top 5 highest activated patterns.

3.3. Microscopic Scale: Neuron Multifaceted Feature Visualization

When considering neuron learned feature, most attentions are paid to the highest activated feature. As a matter of fact, a neuron learns multifaceted feature patterns (Nguyen et al., 2016), among which is the highest activated feature. As deep neural networks are highly nonlinear with complex weight connections, these multiple aspects all play key roles for neuron understanding and deserve equal attentions. Unfortunately, previous works do not realize it, thus leading to a biased and incomplete view of 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 analogizes and extends layer feature visualization in Section 3.2.2. Firstly, a range of features (e.g. top 1000 highest activated features) are extracted and clustered into groups. Secondly, the clusters and centroids are projected into 2D space using t-SNE. Lastly, for each feature cluster, 20 nearest images are weighted averaged for representation. As shown in Fig. 2(b), each node with average feature represents one feature facet. Currently, the limitation lies in that it is hard to detect the number of multifaceted features. Unsupervised clustering is a good direction, however, tuning hyper-parameter (e.g. leaf size in DBSCAN) is miserable. So we simply use K-means with 10 clusters by default. The user could change it based on preferences.

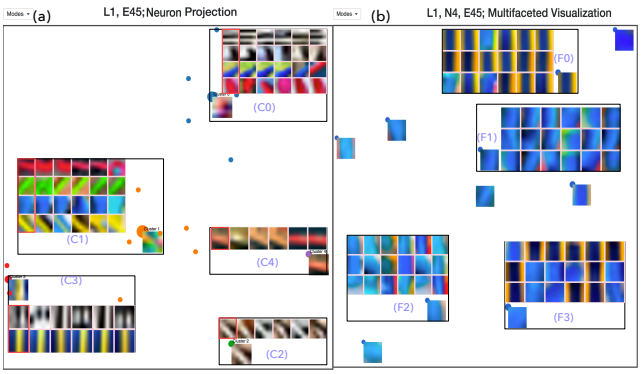

Fig. 2. (a): Layer1 neuron features of *shallow*. (b): Multifaceted feature visualization of 4^{th} neuron in *shallow* layer1.

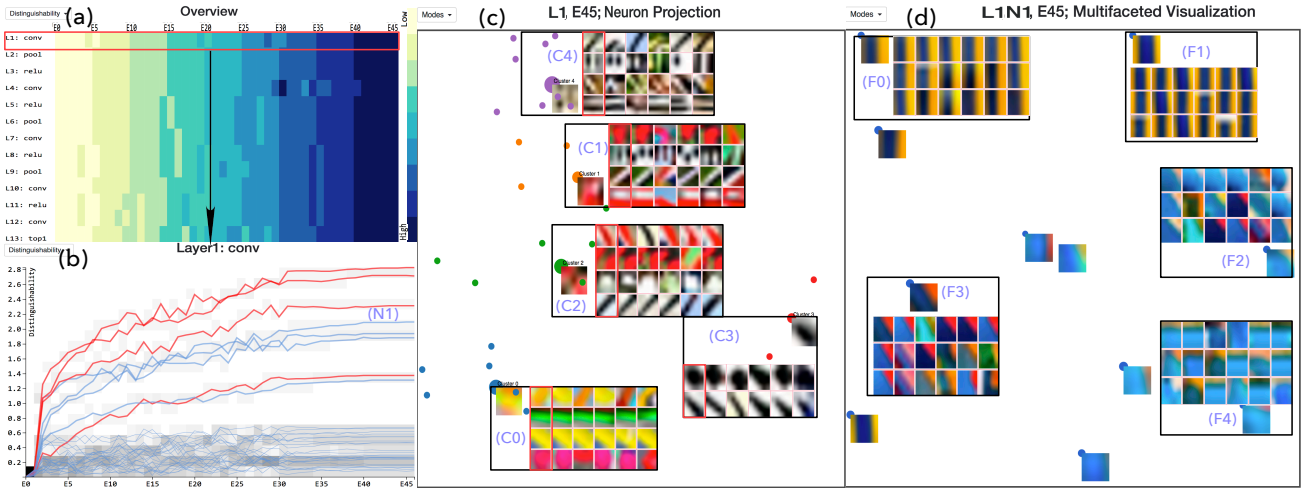

Fig. 3. The interface of our *DV* system for *lenet* visualization. Network discriminability overview (a), is encoded using heatmap across layers (y axis) and epochs (x axis). After selecting a specific layer (layer1), the discriminability overview (b) is shown as a heatmap embedded with line chart. 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.

4. Case Study

To demonstrate the effectiveness of *DV* system, we performed case studies with two domain experts (co-authors) from university research labs. The first case study (Section 4.1) shows our system can promote deep neural network comprehension, reveal potential underfitting/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 the ease of evaluation, we employ CIFAR10 (Krizhevsky & Hinton, 2009), a small but difficult dataset. During case study, they experienced *DV* with various deep neural networks. For reference, Fig. 4 shows related network structures during exploration.

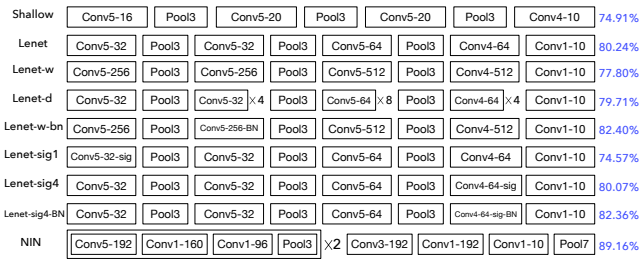

Fig. 4. Network structures and their corresponding accuracies.

4.1. Case Study 1: Network Diagnosis and Refinement

Expert A (E_A), as a third year Ph.D. student, his research interest is data mining and general machine learning. He acts more like a deep model practitioner focusing on task specific networks. Recently, he designed attention based models for location prediction. He would like to utilize our system to compare baseline deep models for understanding inner strategies and pursuing better performances.

4.1.1. Understanding Baseline Network

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

Results of *shallow*'s are shown in Fig. 1 and Fig. 2, and those of *lenet* are shown in Fig. 3. After examining the discriminability overviews in Fig. 1(a) and Fig. 3(a), he found that both networks have a gradually increasing trend, however discriminability in *shallow* was not as stable as *lenet*. What is more, he noticed that layers in Fig. 1(a) (A) block had a decreasing discriminability trend after the 30^{th} epoch. This was consistent with the training setting that he lowered down the learning rate after 30^{th} epoch for fine tuning. From Fig. 3(b) and Fig. 1(b), E_A also observed that better quality neurons in *lenet*. All 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. 3(c) and Fig. 2(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.

Additionally, E_A focused on neuron multifaceted feature visualization for detailed analysis. He found two neurons, 1^{st} neuron in layer1 of *lenet* and 4^{th} neuron in layer1 of *shallow*, having similar feature facets. The corresponding multifaceted features are visualized in Fig. 3(d) and Fig. 2(b) respectively. Both neurons mainly detected same features: light blue edges and dark blue edges. This thorough analysis indicated that in those networks, layer1 neurons learned very similar things. On account of fewer 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 motley average feature background. In view of this, E_A concluded that network depth (capturing higher level features) is one major advantage of *lenet*.

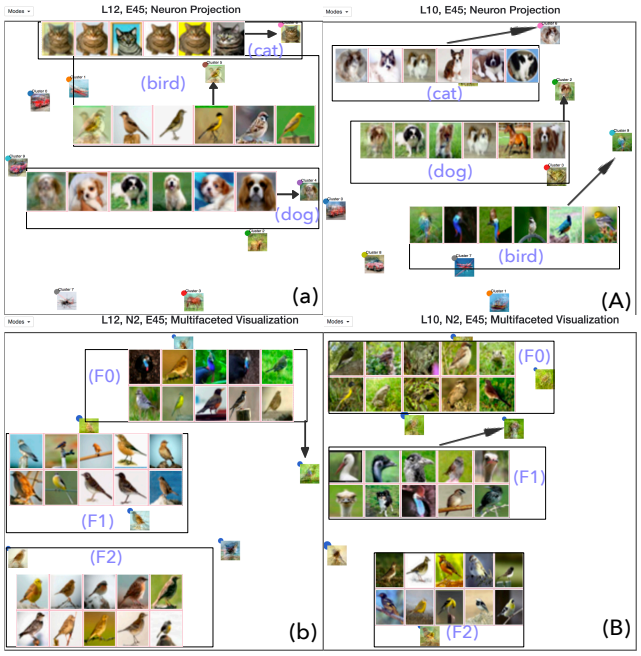

Fig. 5. (a) and (A): Final layer neuron features of *lenet* and *shallow*. Cat, dog and bird neurons are shown with their average feature and top 5 highest activated features. (b) and (B): Bird neuron multifaceted features of *lenet* and *shallow*. Three facet average features are shown in rectangles.

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 1^{st} epoch (a), and learned too sparse representations (b). It is known, sparse representation is beneficial since it reduces noise and outliers, but too sparse is not a good signal.

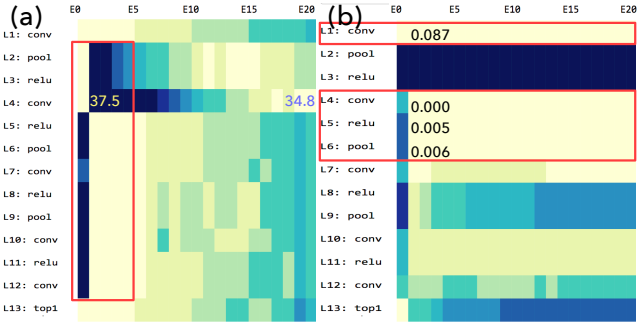

Fig. 6. The overviews of *lenet-w* discriminability (a) and density (b). The digits are density values.

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

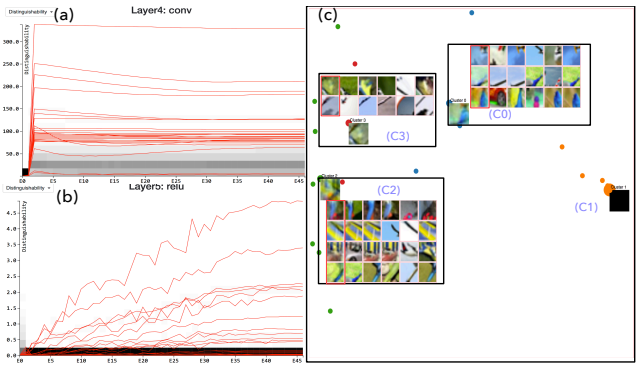

Fig. 7. (a) and (b): Layer4 (CONV) and layer5 (ReLU) discriminability evolution overviews of *lenet-w*. Most neurons keep steady (dead). (c): Layer5 (CONV) neuron learned features. C1 is the dead neuron cluster.

Deep network *lenet-d*: From Fig. 8(a), E_A found that layers in red block suffered from unstable optimization.

For *lenet-d*, in network discriminability overview (Fig. 8(a)), E_A found that layers in red block suffered from unstable optimization. The reason might be vanishing/explosive gradient problem accompanying deep structure. From Fig. 8(b), he noticed that most CONV (with arrows) and ReLU layers learned dense representations (large

density). These dense representation is undesirable owing to the sensitivity to noise and tendency towards overfitting.

After training terminated, E_A checked *lenet-d* layer overview and learned features to see what happened inside the network. He discovered the inner layer (from 27^{th} epoch to 33^{th} epoch) discriminability stayed unchanged, showing a terrible evolution direction (Fig. 8(c)). Besides, from Fig. 10, he observed the neurons had already learned to detect class-wise concepts (truck/car and ship/dog). It seems that the network was premature (learning complex features early), i.e. former layers being high level enough and latter layers being unnecessary, adding model complexity and risking of overfitting. After that, to further check neuron quality, he looked into neuron multifaceted features in Fig. 11, where he found there are many pure facets other than highest activated patterns. From this, E_A made for sure that *lenet-d* was premature. He thought because of 3×3 filter size, RF increased too fast and captured high level features too early. Actually, *lenet-d* became overfitting after 28^{th} epoch and the prediction accuracy was only 79.7%.

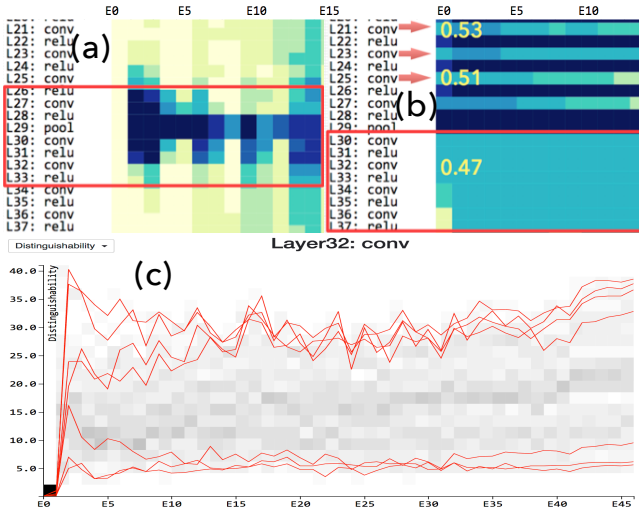

Fig. 8. The overviews of *Lenet-d* discriminability (a) and density (b). The digits are density values.

Based on the above analysis, E_A came to know that moderate depth, width and RF size are vital to network performance. To validate, he applied network in network (*NIN*) (Lin et al., 2013) structure with self-designed 1×1 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 CONV layer output to increase CNN performance. But he is curious about why BN is so robust to large learning rate and careless initialization. E_B adopted our system to resolve these questions in practice.

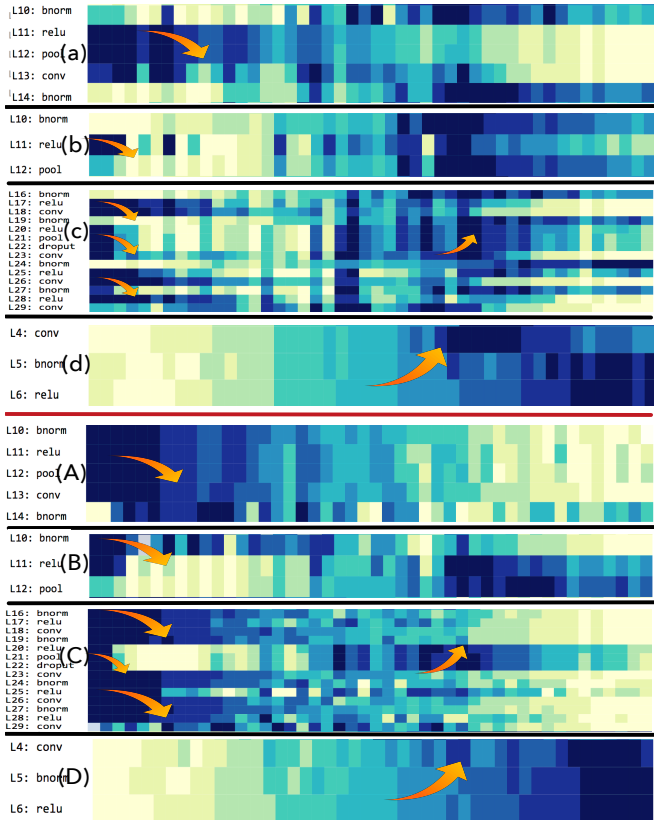

Fig. 9. Layers with exceptional discriminability patterns of *lenet-bn* (a), *shallow-bn* (b), *NIN-bn* (c) and *lenet-w-bn* (d). (A), (B), (C) and (D): The corresponding layer density patterns. The arrows indicate increase or decreaspatterns and heatmaps colors are encoded locally to visualize layer evolution.

Starting from simple neural networks, E_B added BN layers after every CONV layer of *lenet*, *shallow* and *NIN* networks and noted them as *lenet-bn*, *shallow-bn* and *NIN-bn* networks. He first checked discriminability overviews and found some layers presenting exceptional patterns (Fig. 9(a), (b) and (c)). Normally, most layers of a well trained network will exhibit a general increasing trend of discriminability, but these selected layers experienced initial decreasing trends. It is hard for E_B to explain the decline reasons of a well-trained network.

Afterwards, he switched back to network density overview (Fig. 9(A), (B) and (C)) and captured that these layers' densities had the same evolution trends as discriminabilities. This means all these layers originally tried to learn sparse representation. It is interesting since it signifies 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 too sparse representation in layer4, he added a BN layer after layer4 to see the response. In Fig. 9(d) and (D), he noticed the consistency as well. While the only difference was that this time layer4 gradually learned well populated representation 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. 9(a), (b) and (c)), the intuition is that the normalization and rescaling of BN may shrink input range and fail Wasserstein distance . After all, optimizing the network discriminability is different from optimizing a single neuron (Bau et al., 2017).

From above analysis, E_B concluded that BN tends to learned dense or sparse representation 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 too dense representation, BN would normalize ReLU input to smaller values and learn sparse representations.

4.2.2. Activation Function Understanding

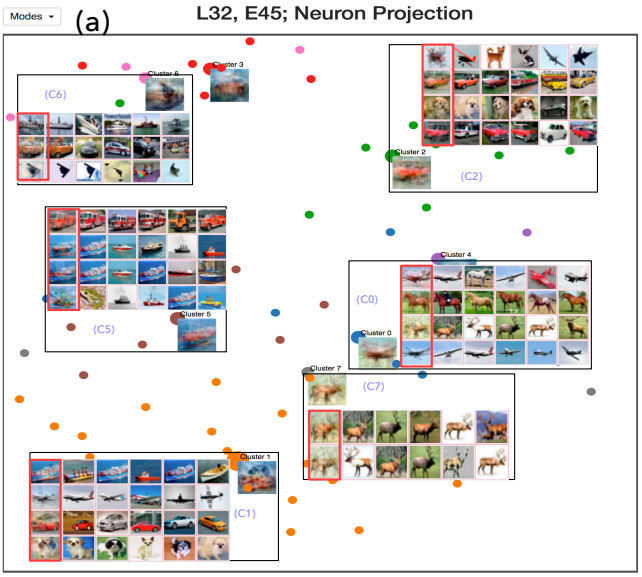

Fig. 10. Neuron features of *lenet-d* layer32.

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

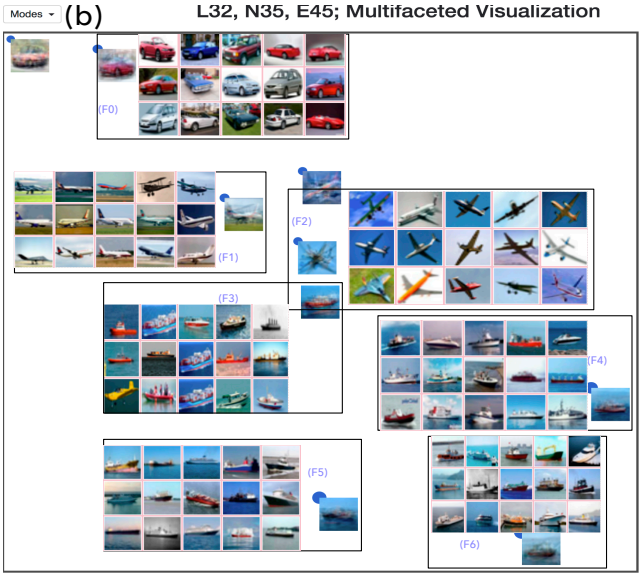

Fig. 11. Multifaceted feature visualization of 35^{th} neuron in *lenet-d* layer32.

E_B trained two new networks, namely *lenet-sig1* and *lenet-sig4* (in Fig. 4), which replaced *lenet*'s 1^{st} ReLU and 4^{st} 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 *lenet-sig4* (layer11). However, contrasting Fig. 12(a) and Fig. 3(a), he realized that this tiny influence really changed network fine tuning optimization pattern. He later examined layers' 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 other than weak increasing trends. In Fig. 12(c), he noticed layer11 evolved unhealthily fast with minor increasing after 3rd epoch. Since layer11 was a sigmoid layer, a neuron's discriminability stopped increasing after becoming saturated.

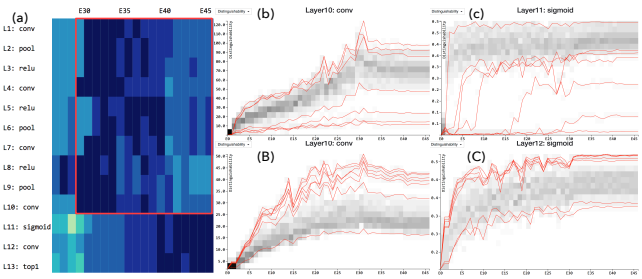

Fig. 12. (a): *Lenet-sig4* discriminability overview. Layers inside the red rectangle show tiny discriminability dropping down after 30^{th} epochs. (b) and (c): Layer10 and layer11 discriminability evolutions of *lenet-sig4*. (B) and (C): Layer10 and layer11 discriminability evolutions of *lenet-sig4-bn*.

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 *lenet-sig4* sigmoid layer with over 2% accuracy increase. After checking layer10 and layer12 discriminability evolutions in Fig. 12(B) and (C), he found layer10 had stronger increasing trends, and layer12 learned steadily and became saturated much later. This provides a good solution to solve sigmoid saturation problem and validates the effectiveness of BN.

5. Conclusion and Discussion

In this paper, we propose, *DV*, a scalable visual analytics approach, enabling deep neural network inspection in real-time for understanding, diagnosis and refinement. Specifically, two powerful quantitative metrics, *discriminability* and *density* are devised for layer and neuron evaluation. Weighted average based hierarchical exploration method is designed for multifaceted neuron features visualizations. Based on two case studies, we prove the correctness, effectiveness and efficiency of our system.

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

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. Multifaceted 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.