Better Loss Landscape Visualization for Deep Neural Networks with Trajectory Information

Ruiqi Ding DRQ12138@SJTU.EDU.CN
Tao Li LI.TAO@SJTU.EDU.CN
Xiaolin Huang XIAOLINHUANG@SJTU.EDU.CN
Department of Automation, Shanghai Jiao Tong University, Shanghai, China

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

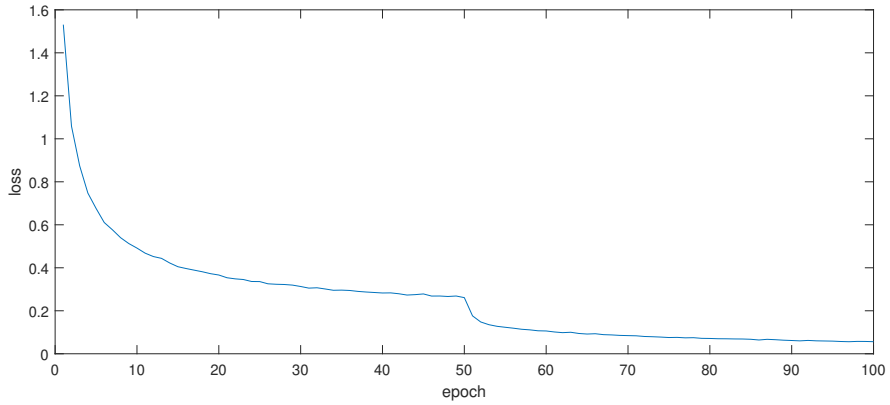
The loss landscape of neural networks is a valuable perspective for studying the trainability, generalization, and robustness of networks, and hence its visualization has been extensively studied. Essentially, visualization methods project the parameter space into a low-dimensional subspace, resulting in a substantial loss of network parameter information. The key is to identify the direction of loss reduction in the visualized loss landscape. However, the existing methods generally focus on one simple point, make it challenging to properly capture the main properties of the landscape. An obvious and important problem is that regardless of whether the center point is the convergence or not, the current methods may depict it a local optimal point in the visualization. To address this issue, we propose a visualization method that relies on the whole training process not a single solution, better reflecting the actual training loss.

Keywords: Loss landscape visualization, Trajectory information

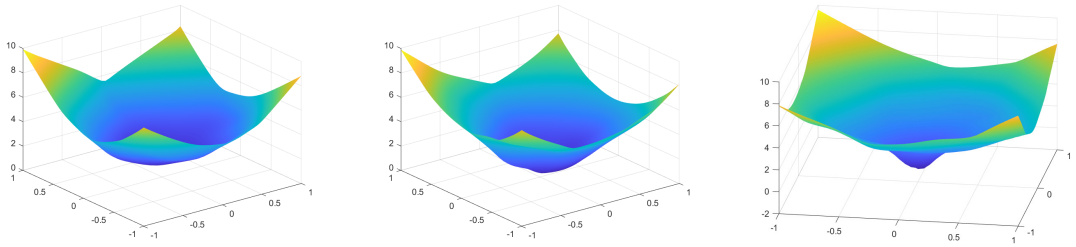
1. Introduction

The loss landscape of deep neural networks (DNNs), which captures the changes of the loss function in the parameter space, can provide a better understanding of the learning and generalization behaviors, and offer insights into the optimization process during training as well as the factors that are related to the model generalization performance, such as architecture and training configurations.

It is widely acknowledged that DNNs often contain a large number of parameters, which leads to difficulties in visualizing the parameter space due to its high dimensionality. To address this issue, early methods for visualizing the loss landscape in one dimension involve selecting two points in the parameter space and calculating the changes in the loss function along the line segment connecting these points. This visualization method is commonly used to explain the training process of neural networks and has found numerous applications in the fields of deep ensembles, and Bayesian networks (Sun et al., 2020; Fort et al., 2019; Garipov et al., 2018). Later researchers suggested projecting the parameter space into a lower dimension and visualizing the loss landscape in the low-dimensional space (Goodfellow et al., 2014; Li et al., 2018c; Chatzimichailidis et al., 2019). A common approach is to select network parameter values as the visualization center point. By calculating the corresponding loss function values as the parameters change in the low-dimensional space near the center



(a) Training loss on CIFAR-10 with ResNet20



(b) Epoch=20 (c) Epoch=50 (d) Epoch=100

Figure 1: **Landscape visualization during the training process by FN method.** The experiment is conducted on CIFAR-10 with ResNet20. The trend of the loss function during the training process with the number of training iterations is shown in (a), while (b), (c), and (d) respectively shows the loss landscape near the network parameter values of epoch 20, 50, and 100 using the FN method for visualization. Notably, the loss landscapes always consider the current center point to be a local minimum, regardless of which training stage is chosen for network parameter values as the center point for visualization.

point, a surface map of the loss landscape in three-dimensional space can be visualized. This approach provides a clear demonstration of how the loss function changes as the network parameters vary. Moreover, by examining the local geometric features (sharpness or flatness) of the landscape, we can analyze the optimization process of parameters. This visualization method is commonly employed to examine the generalization and robustness of networks, as well as the optimization process (Wu et al., 2020; Shafahi et al., 2019; Maddox et al., 2019; Zhou et al., 2020).

However, this visualization method is also subject to the issue of being unable to identify the direction of optimization. In Figure 1, we use "Filter Normalization" (FN) method (Li et al., 2018c) to visualize the loss landscape of the network. We use the parameter values of the network at different stages of training as the center point for visualization. The results

consistently show a pattern of low values at the center and high values around it, without indicating any direction of loss reduction.

During training process, it is easy to identify the direction of loss reduction in the early stages (such as the direction of gradient descent). As the training progresses, the gradients of the parameters decrease, which makes it difficult to identify an obvious direction for reducing the loss. When using the FN method, the visualized loss landscape does not reveal the expected direction of loss reduction.

The FN visualization method is limited by its inability to preserve sufficient information when projecting the parameter space into a low-dimensional space, which makes it challenging to identify the direction of loss reduction in this space. Consequently, regardless of the parameter point selected as the visualization center point, the current position is displayed as a local minimum. To preserve as much network parameter information as possible during the projection process, a suitable visualization subspace can be selected by utilizing the network training trajectory. Researchers have proved that the training trajectory of neural networks encapsulates specific network parameter information that can be leveraged to guide network training (Li et al., 2022b). During the training process, not all parameters are optimized independently, leading to a training trajectory that manifests low-dimensional characteristics rather than variation throughout the entire parameter space.

Based on this inspiration, we propose that visualizing the loss landscape of neural networks only requires attention on a low-dimensional space that contains the training trajectory, rather than the entire parameter space. This low-dimensional space offers sufficient information to reflect the changes in the neural network throughout the training process, and its low dimensionality is suitable for visualization. Our visualization method can clearly display the direction of loss reduction in the loss landscape during training. Moreover, our method can visualize the training trajectory, which can assist researchers in further studying the process of network training.

2. Related Works

2.1. Loss Landscape Visualization

Several studies have analyzed the neural network loss function's changes along a parameter space line segment, visualizing the loss landscape in a one-dimensional scenario.Goodfellow et al. (2014) consider two points $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_1$ in the parameter space, and points on the line segment connecting them $\boldsymbol{\theta} = (1-\alpha)\boldsymbol{\theta}_0 + \alpha\boldsymbol{\theta}_1$ for a range of α values. By calculating the loss function, $Loss(\boldsymbol{\theta})$, at a series of points along this line segment, the change in loss function can be visualized. This study reveals that various state-of-the-art neural networks follow a straight path from initialization to solution, encountering no significant obstacles. This method is widely used to study the sharpness and flatness of different minima (Keskar et al., 2016; Dinh et al., 2017; Im et al., 2016; Smith and Topin, 2017) and to explain the training process of neural networks (Sun et al., 2020; Fort et al., 2019; Garipov et al., 2018). However, the visualization results are not sufficiently intuitive and only cover a limited range of parameter space.

In addition to one-dimensional visualization methods, people have also proposed the use of projection to visualize the surface map of the loss landscape Li et al. (2018c); Chatzimichailidis et al. (2019). In a network with parameters $\boldsymbol{\theta}$, they generate random

Gaussian direction vectors and normalize them. With two normalized directional vectors $\{\boldsymbol{d_1}, \boldsymbol{d_2}\}$, loss landscape surf map can be plotted by calculating the loss value of $f(\alpha, \beta) = Loss\left(\boldsymbol{\theta}^* + \alpha\boldsymbol{d_1} + \beta\boldsymbol{d_2}\right)$, where $\boldsymbol{\theta}^*$ represents the center point of loss landscape. This method can visually demonstrate changes in the loss function within the parameter space and is frequently utilized to elucidate how network structure affects generalization, robustness, and trainability (Foret et al., 2020; Shafahi et al., 2019; Maddox et al., 2019; Zhou et al., 2020; Loshchilov and Hutter, 2017; Zhai et al., 2019). However, this visualization method loses excessive network parameter information during projection, which makes it impossible to find the loss reduction direction on the landscape and represents any parameter values as a local optimal. In contrast, our method selects a suitable visualization subspace according to the training trajectory, retains more network information in the landscape, and reflects the network training process.

Besides visualizing the neural network landscape, researchers are also interested in visualizing the trajectory of network training. Multidimensional scaling (MDS) is used to reduce the dimensionality of network parameter values updated in each training iteration, (Poggio and Liao, 2017) can visualize the reduced training trajectory parameter values in a two-dimensional plane. This method can clearly demonstrate the changes in the training trajectory and is used to study the training process characteristics of different networks (Sun, 2019; Nguyen and Hein, 2017; Poggio et al., 2017; Kriegeskorte and Golan, 2019; Gotmare et al., 2018). However, the visualization results are insufficient in displaying the training trajectory on the loss landscape.

2.2. Low-dimensional Training

Although the number of parameters in a neural network is large, their redundancy enables low-dimensional subspace training, leading to remarkably effective results. Gur-Ari et al. (2018) proposed the hypothesis that the landscape of DNNs' objective functions may exist within a lower-dimensional subspace. Numerous studies (Izmailov et al., 2018, 2020; Hu et al., 2020; Dogra and Redman, 2020; Li et al., 2022b, 2018b; Ghorbani et al., 2019; Athiwaratkun et al., 2018) have been conducted based on this hypothesis, with the goal of investigating the optimal approach to training networks in a parameter space of lower dimensionality, while simultaneously achieving high network performance.

Researchers randomly project the neural network parameters into a low-dimensional space and trained the network therein (Li et al., 2018a). This method achieves 90% accuracy of regular SGD training. Some subsequent improvement methods (Gressmann et al., 2020) take into account the impact of network structure on dimensionality reduction and adjust the method of random projection based on different segments of the network parameters. In Li et al. (2022b), the authors reduce the dimensionality of the network parameters and obtain a low-dimensional subspace. By training in this subspace, they achieve almost the same accuracy as regular SGD training, while improving training efficiency and network robustness. This technology can be applied to many fields, such as federated learning (Li et al., 2023; Xie et al., 2022), image reconstruction (Barbano et al., 2023), and network weight averaging (Li et al., 2022a).

We utilize the low-dimensional training characteristics of the network to visualize the loss landscape. By projecting the network parameters into the subspace containing training

trajectory, the visualization results preserve as much network parameter information as possible.

3. Methods

Our approach utilizes the low-dimensional characteristics of the network trajectory to visualize the loss landscape. The low-dimensional space containing the training trajectories can be obtained by recording the parameter values during network training and applying PCA to extract principal components. Furthermore, the standardization process ensures the consistency of the visualization subspace, maximizing the representation of network parameter information within the visualized space.

3.1. Landscape Visualization

Consider a DNN $f(\boldsymbol{x}, \boldsymbol{w})$ with input $\boldsymbol{x}$ and parameters $\boldsymbol{w} \in \mathbb{R}^n$. To identify the subspace of the visualized loss landscape, we first pretrain the network and denote the training trajectory as $\{\boldsymbol{w}_i\}_{i=0,1,\ldots,t}$, where $\boldsymbol{w}_i$ represents the model's parameters at training step i, and t is the total number of training iterations.

Without losing generality, we choose the training steps from m to $m+n$, denoted as $W = [\boldsymbol{w}_m, \boldsymbol{w}_{m+1}, \ldots, \boldsymbol{w}_{m+n}]$. It is necessary to centralize these samples as $\hat{W} = [\boldsymbol{w}_m - \bar{\boldsymbol{w}}, \boldsymbol{w}_{m+1} - \bar{\boldsymbol{w}}, ..., \boldsymbol{w}_{m+n} - \bar{\boldsymbol{w}}]$, where $\bar{\boldsymbol{w}} = \frac{1}{n}\sum_{i=m}^{m+n} \boldsymbol{w}_i$.

We need to find a d-dimensional subspace to cover $\hat{W}$, which is denoted as a space spanned by $P = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_d]$, and this space can be determined by minimizing the sum of distances from the column vectors of $\hat{W}$ to the subspace. By using the l_2 norm, the problem can be formulated as maximizing the variance of the projection of $\hat{W}$, which is a PCA problem:

$$\max_{P} \quad \mathrm{tr}\left(P^\top \hat{W}\hat{W}^\top P\right), \quad \text{s.t.} \quad P^\top P = I. \tag{1}$$

To create a surf plot of the loss landscape, it is only necessary to identify a two-dimensional subspace that encompasses the training trajectory. So we extract the eigenvectors corresponding to the two largest eigenvalues of $\hat{W}$, which is denoted as $P = [\boldsymbol{p}_1, \boldsymbol{p}_2]$.

As the eigenvectors are all unit vectors, normalization is necessary to match the model parameters and ensure that the loss landscape is plotted within the correct range. Previous research has utilized the "Filter Normalization" method for normalization. The FN method normalizes each filter in $\boldsymbol{p}_i$ separately, altering the subspace that initially contained the training trajectory and leading to a higher information loss during projection. To preserve the subspace's integrity, we normalize $\boldsymbol{p}_i$ as

$$\boldsymbol{d}_i = \|\boldsymbol{w}_{\text{center}}\| \, \boldsymbol{p}_i, \tag{2}$$

here, $\mathbf{w}_{\text{center}} \in R^n$ denotes the model parameter value at the center of the loss landscape. $\boldsymbol{d}_i$ represents the normalized eigen vector, which is also the direction vector of loss landscape. The loss landscape can be visualized by computing $f(\alpha, \beta) = Loss(\boldsymbol{w}_{\text{center}} + \alpha\boldsymbol{d}_0 + \beta\boldsymbol{d}_1)$, where α and β takes several values along the range of -1 to 1.

3.2. Trajectory Visualization

In addition to visualizing the loss landscape of networks, our method can also be applied to visualize the training trajectory by projecting the network parameters into a low-dimensional subspace. Given the projection matrix $P = [\boldsymbol{p}_1, \boldsymbol{p}_2]$ and the training trajectory $W = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n\}$, the projected path trajectory in low-dimensional space can be calculated as $\tilde{W} = P^T W$. Each element in $\tilde{W}$ indicates the corresponding coordinate of the parameter values along the trajectory in the low-dimensional space. By using the low-dimensional projection coordinates of the training trajectory and the corresponding loss at each training step, we can create a line chart to visually represent the trajectory.

Since the projection from a high-dimensional space to a low-dimensional space is many-to-one, it implies that multiple high-dimensional parameter points can project to the same low-dimensional point. As a result, there exists a discrepancy between the loss value on the training trajectory and corresponding coordinate on the loss landscape when visualizing the trajectory on the loss landscape. This is due to the fact that, despite having the same coordinates in the low-dimensional space, they correspond to distinct values in the high-dimensional parameter space. We propose "Fitting Error" to measure the effectiveness of the trajectory visualization result on the loss landscape.

$$\begin{aligned} Err &= \frac{1}{n}\sum_{i=1}^{n}\left[\ln\frac{Loss(\boldsymbol{w}_i)}{Loss(\tilde{\boldsymbol{w}}_i)}\right]^2 \\ \tilde{\boldsymbol{w}}_i &= \boldsymbol{w}_{\text{center}} + \alpha_i \boldsymbol{d}_0 + \beta_i \boldsymbol{d}_1 \\ [\alpha_i, \beta_i] &= (\boldsymbol{w}_i - \boldsymbol{w}_{\text{center}})^T P. \end{aligned} \tag{3}$$

Here, Err denotes the "Fitting Error" of the trajectory and $\boldsymbol{w}_i$ denotes the parameters of the neural network at the i-th iteration. The coordinates of these values in the low-dimensional space are represented by α_i and β_i. Additionally, $\tilde{\boldsymbol{w}}_i$ corresponds to the parameter values at the coordinates on the loss landscape. By calculating the difference of loss function when the network parameters are $\boldsymbol{w}_i$ and $\tilde{\boldsymbol{w}}_i$, we measure the fitting effect of the visualized trajectory and the loss landscape. As the loss exhibits a large variation in magnitude when network parameters changes in the parameter space, we take the logarithm of the loss. According to 2, we can simplify the computation of the corresponding coordinates on the loss landscape as

$$\tilde{\boldsymbol{w}}_i = \boldsymbol{w}_{\text{center}} + \|\boldsymbol{w}_{center}\| \, PP^T \left(\boldsymbol{w}_i - \boldsymbol{w}_{\text{center}}\right). \tag{4}$$

4. Experiments

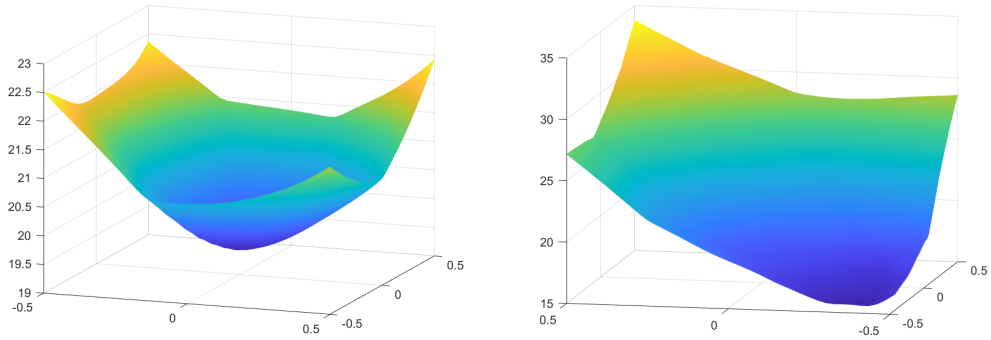
4.1. Experiments Setup

We experiment over two datasets, CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), and apply basic data augmentations, such as random horizontal flip with a 0.5 probability and random crop with a size of 32. Additionally, the input data for each channel are normalized using their respective mean and standard deviation. The model structures we used include ResNet20 (He et al., 2016), VGG11 (Simonyan and Zisserman, 2014), GoogLeNet (Szegedy et al., 2015), and other architectures. We train the DNNs using SGD optimizer (Ruder,

2016), for which the learning rate is set as 0.1 and weight decay is set as 1e-4. We train the DNNs for 100 epochs with batch size of 128.

4.2. Visualizing Loss Landscape

We employ both FN method and our method to visualize the loss landscape of the ResNet20 network. Figure 2 depicts the visualization results.



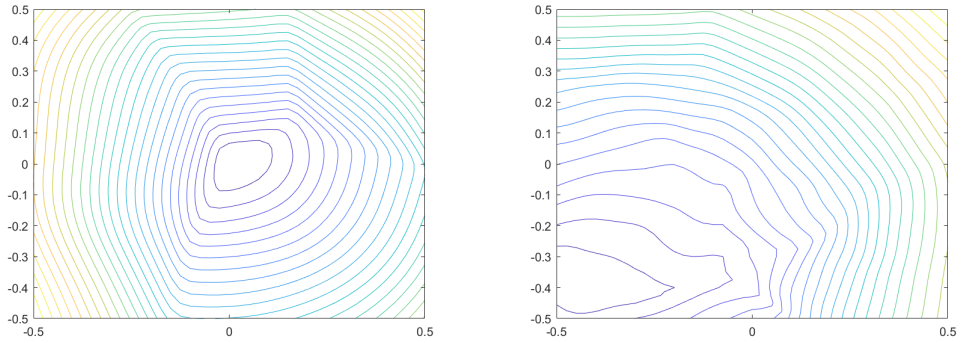
(a) FN method (b) Our method

Figure 2: **Landscape visualization by FN method and our method.** The experiment is conducted on CIFAR-10 with ResNet20, and training initial point is the visualization center point. (a) shows the loss landscape visualized using the FN method, where the center point is considered as a local minimum point; (b) shows the loss landscape visualized using our method, where directions of loss reduction can be found near the center point.

In the visualization result using the FN method, all directions around the center point of the landscape lead to an increase of loss, which means the current position is considered as a local minimum. However, in training process, the direction of loss reduction is found based on the gradient of network parameters, which is not reflected in the loss landscape. By utilizing our method, the resulting visualization shows that the center point is not a local optimal point, but rather there are directions of both loss increase and decrease in its surroundings. This corresponds to the fact that during the training process, there are still directions of loss reduction near the current position, such as the gradient descent direction.

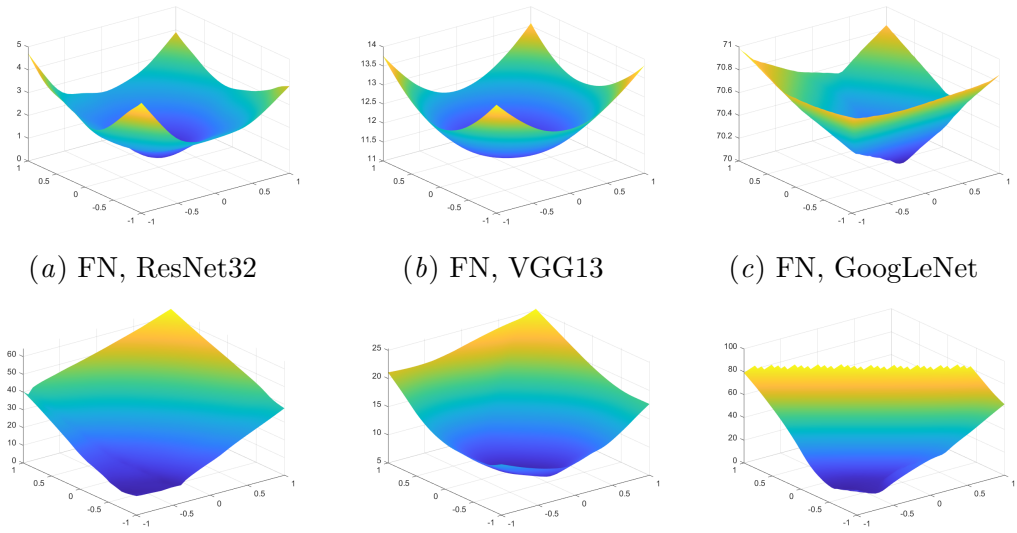
Furthermore, we generate contour maps of the corresponding loss landscapes, which is illustrated in Figure 3. The loss landscape visualized by FN method only displays a local minimum at the center point, while the surrounding direction uniformly increases. In contrast, the visualization result obtained using our method indicates a clear direction of loss decrease. The center point is situated along the path of descent, but not at a local minimum.

To validate the effectiveness of our visualization method across different models, we train ResNet32, VGG13, and GoogLeNet networks on CIFAR-100 and visualize the loss landscape using both FN method and our method. The visualization results are presented



(*a*) FN method (*b*) Our method

Figure 3: **Contour maps of landscape visualized by FN method and our method.** The experiment is conducted on CIFAR-10 with ResNet20, and training initial point is the visualization center point. (a) shows the contour map of landscape visualized by FN method. (b) shows the contour map of landscape visualized by our method.



(*a*) FN, ResNet32 (*b*) FN, VGG13 (*c*) FN, GoogLeNet

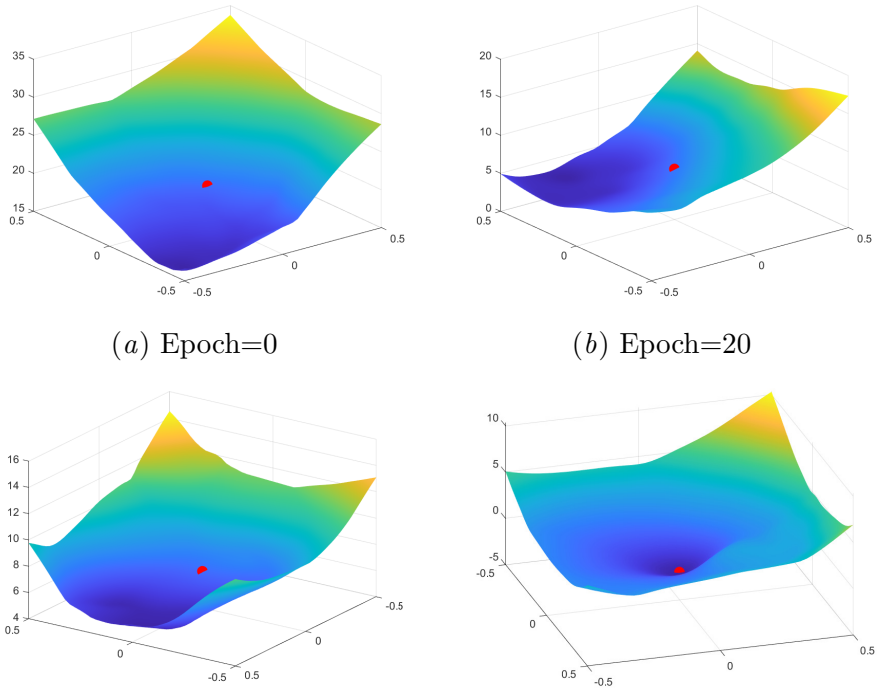
(*d*) Ours, ResNet32 (*e*) Ours, VGG13 (*f*) Ours, GoogLeNet

Figure 4: **Landscape visualization by FN method and our method of different networks.** The experiment is conducted on CIFAR-100, and training initial point is considered as center point. (a), (b), and (c) depict the visualization results of the ResNet32, VGG13, and GoogLeNet networks using the FN method, while (d), (e), and (f) illustrate the results of the corresponding networks when visualized using our method.

in Figure 4. The center point of the landscape was chosen as the training initialization point in all above experiments. When using FN method, the center point in the loss landscape of VGG11 and VGG13 networks is represented as a local minimum, while the center point in the loss landscape of GoogLeNet is located near a local minimum, but there is no clear direction of loss decrease around it. However, the visualization results obtained using our method reveal clear directions of loss decrease near the center point. It clearly shows that our method is capable of displaying the loss descent direction on the landscape when visualizing networks with varying structures.

4.3. Visualizing Training Process

We conduct experiments on CIFAR-10 with ResNet20 network, and select parameter values at different training steps as the center point to visualize the loss landscape around them. Figure 5 presents the visualization results.



(*a*) Epoch=0 (*b*) Epoch=20

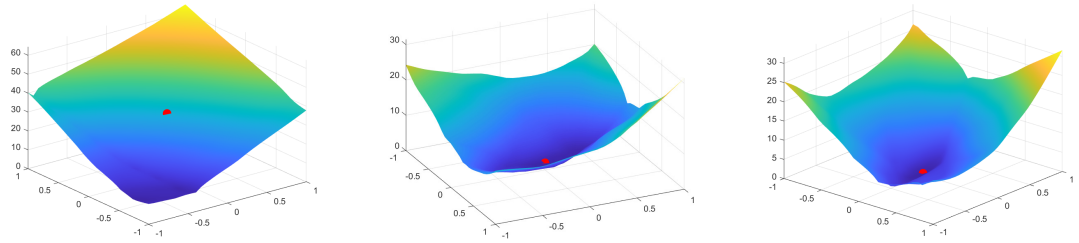
(*c*) Epoch=50 (*d*) Epoch=100

Figure 5: **Landscape visualization by our method with different center points.** The experiment is conducted on CIFAR-100 with ResNet20. (a), (b), (c), and (d) sequentially show the visualization results of loss landscape with network parameters of epoch 0, 20, 50, and 100 in the training trajectory as the center point. The center point are marked with red bubbles in the landscapes.

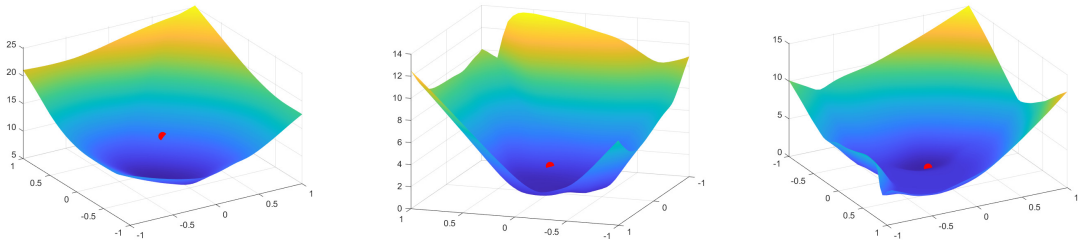
During the early stages of network training, the network loss function is large, and the direction of loss decrease (e.g., gradient descent direction) can be easily determined. The

corresponding loss landscape also reveals that the current center point is on the descent trajectory, providing a clear direction for loss reduction. As training progresses, the network parameters gradually approach the convergence point, leading to a gradual decrease in the parameters' gradients. The corresponding loss landscape exhibits a smoother terrain, with less pronounced trends in the downward direction. As the network approaches convergence, the network parameters approach local optimal, and it becomes difficult to find the direction of loss reduction in the surrounding area. The corresponding loss landscape takes the center point as the local minimum point.

In figure 6, experiments are conducted on CIFAR-100 with ResNet32 and VGG13 networks. We select the network parameters at epoch 0, 50, and 100 during the training process as the center point and visualize loss landscapes around them using our method. Regardless of the type of network used, the loss landscape can identify the direction of loss reduction around center point in the early training stage and take the form of a local minimum point near the center point when the training approaches convergence. It clearly shows that at different stages of training, loss landscapes visualized using our method show the corresponding direction of loss decrease, and our method performs well for models with diverse structures.

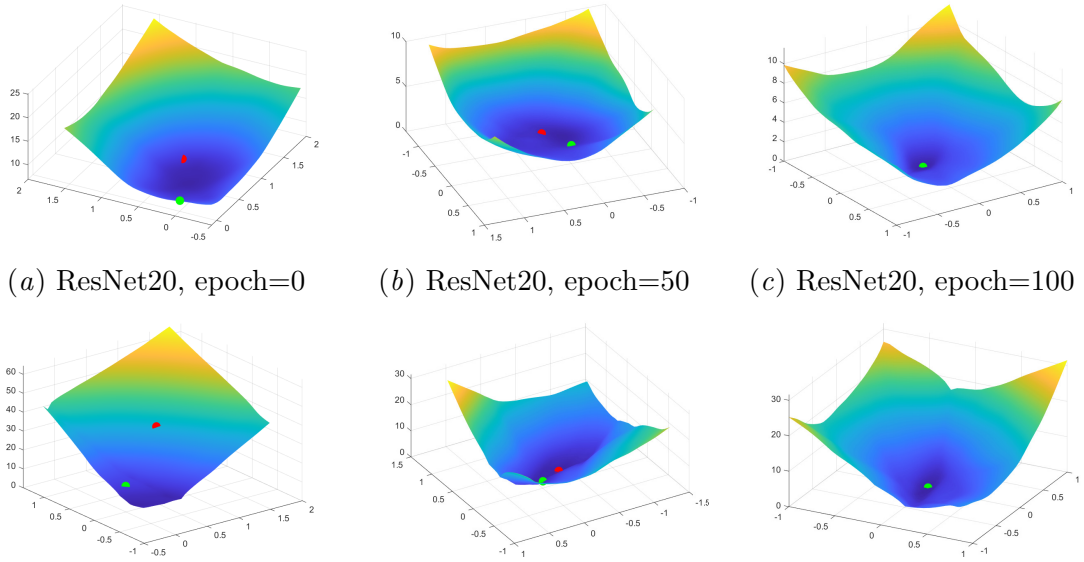


(a) ResNet32, epoch=0 (b) ResNet32, epoch=50 (c) ResNet32, epoch=100



(d) VGG13, epoch=0 (e) VGG13, epoch=50 (f) VGG13, epoch=100

Figure 6: **Landscape visualization by our method with different networks and center points.** The experiment is conducted on CIFAR-100 with ResNet32 and VGG13 networks. (a), (b), and (c) illustrate the visualization results of ResNet32 network with network parameters of epoch 0, 50, and 100 in the training trajectory as the center point, respectively. (c), (d), and (e) show the visualization results of VGG13 network under the same conditions. The center points are marked with red bubbles in the landscapes.



(*a*) ResNet20, epoch=0 (*b*) ResNet20, epoch=50 (*c*) ResNet20, epoch=100

(*d*) ResNet32, epoch=0 (*e*) ResNet32, epoch=50 (*f*) ResNet32, epoch=100

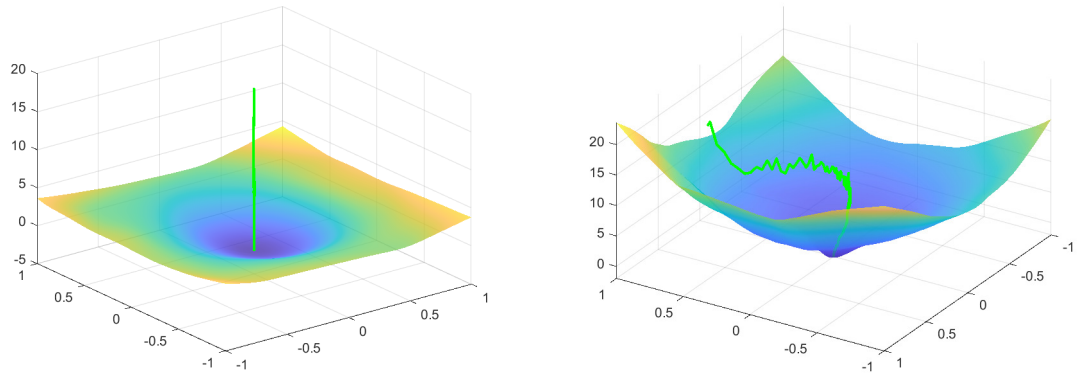
Figure 7: **Landscape visualization by our method with different models and center points.** The experiment is conducted on CIFAR-100 with ResNet20 and ResNet32. (a), (b), and (c) illustrate the visualization results of ResNet20 with network parameters of epoch 0, 50, and 100 in the training trajectory as the center point, respectively. (c), (d), and (e) show the visualization results of ResNet32 network under the same conditions. The center point are marked with red bubbles and convergence point are marked with green bubbles in the landscape.

In order to validate the accuracy of the descent direction in visualization result, we conduct experiments on CIFAR-100 with ResNet20 and ResNet32 networks, and project the network's converged parameter values into visualization subspace and indicate their location on the loss landscape. Figure 7 depicts the projected results.

In the loss landscape centered at the epoch 0 network parameters, an apparent loss descent direction can be observed near the center point, and the network convergence point lies on this direction. For the loss landscape near epoch 50 network parameters, the center point is closer to the convergence point, but the loss descent direction is not as evident. In the loss landscape close to the network convergence point, the center point coincides with the convergence point, and they are regarded as a local minimum on the loss landscape. It clearly shows that by using our method, the loss descent direction on the loss landscape indicates the direction towards the network's converged parameter value.

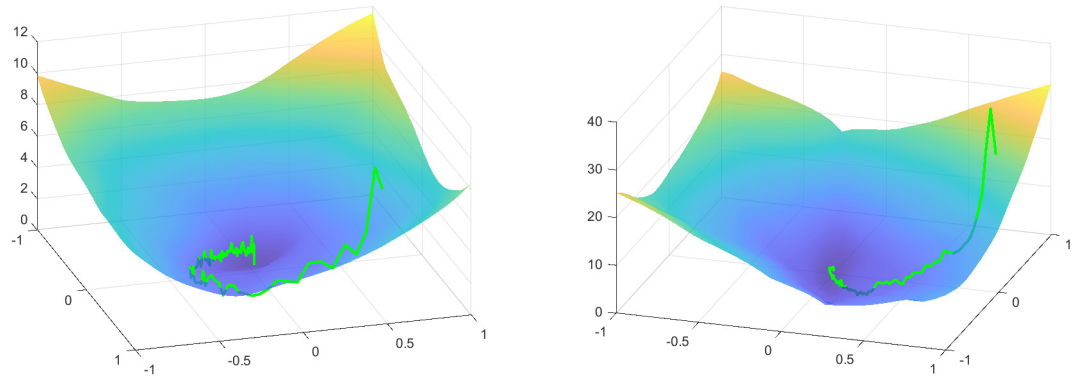
4.4. Visualizing Trajectory

We visualize the training trajectory of ResNet20 network on CIFAR-10 using both FN method and our method, and the result is shown in Figure 8. The FN method's landscape produces a visualization result where the training trajectory is presented as a vertical



(*a*) FN method (*b*) Our method

Figure 8: **Trajectory visualization by FN method and our method.** The experiment is conducted on CIFAR-10 with ResNet20. (a) shows the visualization result of the training trajectory using FN method. (b) shows the visualization result using our method.



(*a*) ResNet20 (*b*) ResNet32

Figure 9: **Trajectory visualization by our method of different networks.** The experiment is conducted on CIFAR-100 with ResNet20 and ResNet32 networks. (a) shows the training trajectory of ResNet20 visualized using our method. (b) shows the visualization result of ResNet32 using our method.

line since different trajectory parameters are projected into the same coordinate in low-dimensional space. This presentation is unable to depict the changes in the loss function and network parameters during the training process. Conversely, the visualized trajectory using our method clearly displays the changes of network parameters during the training process. Moreover, the trajectory in the visualized low-dimensional subspace fits the

Table 1: Fitting Error of ResNet20 and ResNet32 on CIFAR-10 and CIFAR-100

dataset	cifar10		cifar100	
model	resnet20	resnet32	resnet20	resnet32
FN	77.42	424.54	4.51	38.51
Ours	35.88	41.80	0.82	4.72

loss landscape well. We also visualize the training trajectory of ResNet20 and ResNet32 networks on CIFAR-100 using our method. The results are shown in figure 9.

In Table 1, we conduct experiments on CIFAR-10 and CIFAR-100 with ResNet20 and ResNet32 networks. The Fitting Error is calculated according to 3. When using our method to visualize training trajectory, the Fitting Error is significantly lower than that of the results obtained using the FN method, for different datasets and networks under the same conditions. It clearly shows that our method is capable of effectively visualizing the training trajectory of DNNs. The visualized trajectory provides a clear display of the changes in the loss function and parameters during the network's training process, and exhibits a better fit with the loss landscape.

5. Conclusion

This paper presents a loss landscape visualization method based on low-dimensional characteristics derived from the training trajectory. Compared to traditional methods, our approach is capable of preserving more network parameter information during the visualization process, displaying feasible loss descent directions on a visualized landscape, and visualizing changes to the training trajectory in a low-dimensional space. Our goal is for these visualization methods to assist researchers in explaining the network's trainability, robustness, and generalization.

Acknowledgments

We thank the anonymous reviewers for their thoughtful comments that greatly improved the final texts. This work was partially supported by National Natural Science Foundation of China (61977046), Research Program of Shanghai Municipal Science and Technology Committee (22511105600), and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

References

Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. *arXiv preprint arXiv:1806.05594*, 2018.

Riccardo Barbano, Javier Antorán, Johannes Leuschner, José Miguel Hernández-Lobato, Željko Kereta, and Bangti Jin. Fast and painless image reconstruction in deep image prior subspaces. *arXiv preprint arXiv:2302.10279*, 2023.

Avraam Chatzimichailidis, Janis Keuper, Franz-Josef Pfreundt, and Nicolas R Gauger. Gradvis: Visualization and second order analysis of optimization surfaces during the training of deep neural networks. In *2019 IEEE/ACM Workshop on Machine Learning in High Performance Computing Environments (MLHPC)*, pages 66–74. IEEE, 2019.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017.

Akshunna S Dogra and William Redman. Optimizing neural networks via koopman operator theory. *Advances in Neural Information Processing Systems*, 33:2087–2097, 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31, 2018.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pages 2232–2241. PMLR, 2019.

Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. *arXiv preprint arXiv:1810.13243*, 2018.

Frithjof Gressmann, Zach Eaton-Rosen, and Carlo Luschi. Improving neural network training in low dimensional random bases. *Advances in Neural Information Processing Systems*, 33:12140–12150, 2020.

Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Wei Hu, Lechao Xiao, Ben Adlam, and Jeffrey Pennington. The surprising simplicity of the early-time learning dynamics of neural networks. *Advances in Neural Information Processing Systems*, 33:17116–17128, 2020.

Daniel Jiwoong Im, Michael Tao, and Kristin Branson. An empirical analysis of deep network loss surfaces. 2016.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Pavel Izmailov, Wesley J Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Subspace inference for bayesian deep learning. In *Uncertainty in Artificial Intelligence*, pages 1169–1179. PMLR, 2020.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Nikolaus Kriegeskorte and Tal Golan. Neural network models and deep learning. *Current Biology*, 29(7):R231–R236, 2019.

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

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018a.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018b.

Guanghao Li, Li Shen, Yan Sun, Yue Hu, Han Hu, and Dacheng Tao. Subspace based federated unlearning. *arXiv preprint arXiv:2302.12448*, 2023.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018c.

Tao Li, Zhehao Huang, Qinghua Tao, Yingwen Wu, and Xiaolin Huang. Trainable weight averaging for fast convergence and better generalization. *arXiv preprint arXiv:2205.13104*, 2022a.

Tao Li, Lei Tan, Zhehao Huang, Qinghua Tao, Yipeng Liu, and Xiaolin Huang. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022b. ISSN 0162-8828.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2017.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

Quynh Nguyen and Matthias Hein. The loss surface of deep and wide neural networks. In *International conference on machine learning*, pages 2603–2612. PMLR, 2017.

Tomaso Poggio and Qianli Liao. *Theory II: Landscape of the empirical risk in deep learning.* Thesis, 2017.

Tomaso Poggio, Kenji Kawaguchi, Qianli Liao, Brando Miranda, Lorenzo Rosasco, Xavier Boix, Jack Hidary, and Hrushikesh Mhaskar. Theory of deep learning iii: explaining the non-overfitting puzzle. *arXiv preprint arXiv:1801.00173*, 2017.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

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

Leslie N Smith and Nicholay Topin. Exploring loss function topology with cyclical learning rates. *arXiv preprint arXiv:1702.04283*, 2017.

Ruoyu Sun. Optimization for deep learning: theory and algorithms. *arXiv preprint arXiv:1912.08957*, 2019.

Ruoyu Sun, Dawei Li, Shiyu Liang, Tian Ding, and Rayadurgam Srikant. The global landscape of neural networks: An overview. *IEEE Signal Processing Magazine*, 37(5): 95–108, 2020.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33:2958–2969, 2020.

Yueqi Xie, Weizhong Zhang, Renjie Pi, Fangzhao Wu, Qifeng Chen, Xing Xie, and Sunghun Kim. Optimizing server-side aggregation for robust federated learning via subspace training. *arXiv preprint arXiv:2211.05554*, 2022.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1476–1485, 2019.

Pan Zhou, Jiashi Feng, Chao Ma, Caiming Xiong, Steven Chu Hong Hoi, et al. Towards theoretically understanding why sgd generalizes better than adam in deep learning. *Advances in Neural Information Processing Systems*, 33:21285–21296, 2020.