DL4SciVis: A State-of-the-Art Survey on Deep Learning for Scientific Visualization

Chaoli Wang, *Senior Member, IEEE* and Jun Han

Abstract—Since 2016, we have witnessed the tremendous growth of artificial intelligence+visualization (AI+VIS) research. However, existing survey papers on AI+VIS focus on visual analytics and information visualization, not scientific visualization (SciVis). In this paper, we survey related deep learning (DL) works in SciVis, specifically in the direction of DL4SciVis: designing DL solutions for solving SciVis problems. To stay focused, we primarily consider works that handle scalar and vector field data but exclude mesh data. We classify and discuss these works along six dimensions: domain setting, research task, learning type, network architecture, loss function, and evaluation metric. The paper concludes with a discussion of the remaining gaps to fill along the discussed dimensions and the grand challenges we need to tackle as a community. This state-of-the-art survey guides SciVis researchers in gaining an overview of this emerging topic and points out future directions to grow this research.

Index Terms—Scientific visualization, deep learning, survey

1 Introduction

Back in 2007, Ma [110] pointed out the great potential of machine learning (ML) techniques to boost the next generation of visualization (VIS) research. However, before 2010, only sporadic research efforts applied artificial neural networks (ANN) to solve visualization problems, such as volume classification [145], [146] and flow field modeling [86]. As part of ML techniques, deep learning (DL) uses multiple layers in the ANN design to extract higher-level features from data. With the impressive advances in graphics hardware [28], [118], [138] and DL architectures (e.g., AlexNet [90] for image classification, GAN [42] for image generation, and U-Net [122] for image segmentation), the renaissance of artificial intelligence (AI) as a viable solution for solving challenging problems has quickly swept across a wide variety of fields.

Emerged in 2016, AI+VIS has quickly become the fastest-growing area in VIS, with a foreseeable impact for decades to come. Generally speaking, there are two AI+VIS directions: AI4VIS (i.e., designing AI solutions for solving VIS problems) and VIS4AI (i.e., applying VIS techniques for explainable AI). We refer interested readers to recent surveys on AI+VIS [1], [22], [34], [69], [104], [150], [165], [172] to gain a comprehensive overview of this research area. These prior surveys focus on visual analytics (VA) and information visualization (InfoVis). A recent survey of visualization in astrophysics [92] briefly discusses the use of ML techniques for scientific visualization (SciVis) but is restricted to the particular application of focus. In contrast, this survey studies recent advances in DL for SciVis.

DL techniques can bring crucial benefits to SciVis. For example, inference can be performed more efficiently than conventional methods once a neural network is trained [45]. Furthermore, DL solutions offer a performance boost, such as data interpolation quality [55], reduction rate [109], or segmentation accuracy [154], compared with non-DL solutions. SciVis data and tasks share significant similarities with those in computer vision (CV) and computer graphics (CG). Typical SciVis data are 3D and time-dependent scalar and vector volumes, resembling their 2D image and video counterparts in CV and 3D models and animations in CG. CV and CG tasks (e.g., feature learning, extraction, and tracking; data classification, segmentation, generation, and prediction) can easily find their place in SciVis. Visualizing 3D volumetric data brings the same lighting and viewpoint optimization issues as rendering 3D models in CG. Therefore, it is not surprising that the development of DL solutions in CV and CG has nourished DL4SciVis research.

Nevertheless, unlike images and videos in CV, SciVis data often requires the creation of visual representations and an explicit rendering process of those representations for display. For scalar fields, we either extract isosurfaces for visualization or map voxel values to colors and opacities via the transfer function for direct volume rendering. For vector fields, we place seeding points or curves in the domain to trace integral lines or surfaces for visualization. These resulting curves and surfaces share similarities with geometric models in CG. However, CG models are usually closed, while isosurfaces and flow surfaces are often non-closed. Besides geometric properties (e.g., curvature and torsion), flow surfaces also carry physical properties (e.g., density, viscosity, and tension) that need to be considered. Such differences among SciVis, CV, and CG often require customized DL solutions for best solving SciVis problems.

In this paper, we present DL4SciVis, a state-of-the-art survey on DL works for SciVis. Our aims are three-fold: (1) introducing researchers to the recent advances in DL4SciVis; (2) categorizing DL4SciVis works in terms of domain setting, research task, learning type, network architecture, loss function, and evaluation metric; and (3) outlining research opportunities and open challenges. To our best knowledge, this paper is the first survey on DL4SciVis. We hope this comprehensive survey will help SciVis researchers under-

• *C. Wang and J. Han are with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556. E-mail: {chaoli.wang, jhan5}@nd.edu.*

stand this emerging direction and inspire them to join this vibrant research.

2 Scope of Survey

To collect the related papers for this survey, we started with familiar papers published in VIS-relevant venues and searched the DBLP website. We scanned six visualization journals (*IEEE Transactions on Visualization and Computer Graphics*, *Computer Graphics Forum*, *IEEE Computer Graphics and Applications*, *Computers & Graphics*, *Journal of Visualization*, and *Visual Informatics*). In addition, we searched three visualization conferences (VIS, EuroVis, and PacificVis) for relevant full (PacificVis) and short papers (all three). The following keywords were used to identify related papers in their titles: "deep", "learning", "neural", "convolution", "generative", "adversarial", "recurrent", "CNN", "RNN", and "GAN". We then scanned each paper's abstract, introduction, and conclusion to make a final decision.

The criterion for selecting a paper is that the work utilizes deep neural networks (DNNs) to solve SciVis problems. In many cases, DL techniques are applied to address the end goal. However, we also included works that leverage DL techniques to solve intermediate steps that set up the solution to accomplish the final objective. To stay focused, we excluded ML but not DL works (e.g., using the random forest for data classification, support vector machine for data segmentation, or dictionary learning for data synthesis) and papers primarily related to VA or InfoVis instead of SciVis. We omitted works more closely related to CV or CG (e.g., image, video, point cloud, mesh geometry, illumination, shape, and motion) than SciVis topics. We augmented this collection by adding the more recent works of active DL4SciVis researchers from their websites and open-access repositories such as arXiv. Finally, we included a particular set of fluid flow simulation works. They are related to flow visualization (a key SciVis topic) and have inspired DL4SciVis researchers.

It is often difficult to decide whether a paper is closer to SciVis than CV or CG. Ultimately, we check if the paper meets one of the two conditions for possible inclusion: (1) does the publication appear in a VIS venue? (2) has the publication significantly influenced SciVis work? Examples that meet each of the conditions are [154] and [168], respectively.

In the end, we gathered 59 papers for this survey, as listed in Table 1. In the table, we order papers according to task (primary) and setting (secondary), followed by publication year within the same task-setting group. We note that this collection is by no means exhaustive. Instead, it serves as a representative one that allows us to perform a comprehensive review.

3 DL4SciVis Works

This section discusses DL4SciVis works along six dimensions: domain setting, research task, learning type, network architecture, loss function, and evaluation metric. The discussion on the research task dimension provides the primary thread, allowing readers to dive into the surveyed works for in-depth comprehension. We also refer to their research tasks when categorizing the surveyed papers in the respective tables according to learning type, network architecture, loss function, and evaluation metric. The description

TABLE 1: All surveyed papers and their publication venues, tasks, and settings. The five tasks are data **gen**eration [○ ○ ● ●], **vis**ualization **gen**eration [●], prediction [● ●], **obj**ect **det**ection and **seg**mentation [●], and **feat**ure lea**rn**ing and **ext**raction [●]. The four data generation subcategories are super-resolution [○], compression and reconstruction [●], translation [●], and extrapolation [●]. The two prediction subcategories are data-relevant prediction [●] and visualization-relevant prediction [●]. The five settings are scalar, vector, fluid **sim**ulation, particle, and image.

paper	name	venue	task					setting				
			data gen	vis gen	prediction	obj det & seg	feat lrn & ext	scalar	vector	fluid sim	particle	image
Zhou et al. [182]		CGI	○					✓				
Han and Wang [51]	TSR-TVD	TVCG	○					✓				
Han and Wang [50]	SSR-TVD	TVCG	○					✓				
Han et al. [55]	STNet	TVCG	○					✓				
Wurster et al. [167]		arXiv	○					✓				
Guo et al. [47]	SSR-VFD	PVIS	○						✓			
Jakob et al. [76]		TVCG	○						✓			
Sahoo and Berger [126]	IA-VFS	EVIS	○						✓			
An et al. [2]	STSRNet	CG&A	○						✓			
Han and Wang [53]	TSR-VFD	C&G	○						✓			
Xie et al. [168]	tempoGAN	TOG	○							✓		
Werhahn et al. [162]		CGIT	○							✓		
Wang et al. [156]	DeepOrganNet	TVCG	●					✓				
Lu et al. [109]	neurcomp	CGF	●					✓				
Weiss et al. [160]	fV-SRN	arXiv	●					✓				
Shi et al. [131]	GNN-Surrogate	TVCG	●					✓				
Han and Wang [54]	VCNet	VI	●					✓				
Liu et al. [106]		JOV	●						✓			
Han et al. [49]		CG&A	●						✓			
Gu et al. [45]	VFR-UFD	CG&A	●						✓			
Han et al. [56]	V2V	TVCG	●					✓				
Gu et al. [46]	Scalar2Vec	PVIS	●					✓	✓			
Kim et al. [84]	Deep Fluids	CGF	●							✓		
Chu et al. [27]		TOG	●							✓		
Wiewel et al. [163]	LSP	CGF	●							✓		
Wiewel et al. [164]	LSS	CGF	●							✓		
Berger et al. [12]		TVCG		●				✓				
Hong et al. [70]	DNN-VolVis	PVIS		●				✓				
He et al. [63]	InSituNet	TVCG		●				✓				
Weiss et al. [159]		TVCG		●				✓				
Weiss et al. [161]		TVCG		●				✓				
Weiss and Navab [158]	DeepDVR	arXiv		●				✓				
He et al. [62]	CECAV-DNN	VI			●			✓				
Tkachev et al. [143]		TVCG			●			✓				
Hong et al. [71]		PVIS			●				✓			
Kim and Günther [85]		CGF			●				✓			
Han et al. [57]		arXiv			●				✓			
Yang et al. [169]		JOV			●			✓				
Shi and Tao [130]		TIST			●			✓				
Engel and Ropinski [35]	DVAO	TVCG			●			✓				
Wang et al. [154]	VC-Net	TVCG				●		✓				
Nguyen et al. [116]		arXiv				●		✓				
Ghahremani et al. [40]	NeuroConstruct	TVCG				●		✓				
He et al. [65]		JOV				●		✓				
Deng et al. [31]	Vortex-Net	JOV				●			✓			
Berenjkoub et al. [11]		VIS				●			✓			
Kashir et al. [81]		JOV				●			✓			
Borkiewicz et al. [15]	CloudFindr	VIS				●						✓
Raji et al. [120]		EGPGV					●	✓				
Cheng et al. [24]		TVCG					●	✓				
Porter et al. [117]		VIS					●	✓				
Tkachev et al. [144]	S4	TVCG					●	✓				
He et al. [64]	ScalarGCN	JOV					●	✓				
Han et al. [48]	FlowNet	TVCG					●		✓			
Han and Wang [52]	SurfNet	CGF					●	✓	✓			
Chu and Thuerey [26]		TOG					●			✓		
Liu et al. [103]		arXiv					●				✓	
Li and Shen [97]		TVCG					●				✓	
Zhu et al. [183]		CG&A					●					✓

TABLE 2: Acronyms for network architectures. Refer to Section 3.4 for a detailed explanation of basic network architectures and structures (i.e., AE, CNN, GAN, GNN, MLP, RNN).

acronym	full name
AE [4]	autoencoder
CNN [94]	convolutional neural network
DenseNet [73]	densely connected convolutional network
EnhanceNet [127]	enhance neural network
ESPCN [132]	efficient sub-pixel convolutional network
FCN [108]	fully convolutional network
FRVSR-Net [128]	frame-recurrent video super-resolution network
Geo-CNN [93]	geometric-induced CNN
ResNet [61]	residual neural network
Siamese [16]	Siamese neural network (max-pooling CNN)
GAN [42]	generative adversarial network
BEGAN [13]	boundary equilibrium GAN
cBEGAN [111]	conditional BEGAN
cGAN [115]	conditional GAN
ESRGAN [153]	enhanced super-resolution GAN
WGAN [3]	Wasserstein GAN
GNN [43]	graph neural network
GAE [87]	graph autoencoder
GCN [88]	graph convolutional network
GRN [129]	graph recurrent network
STGNN [171]	spatial-temporal GNN
MLP [123]	multilayer perceptron
FCCNN [83]	fully connected cascade neural network
RNN [125]	recurrent neural network
LSTM [68]	long short-term memory
ConvLSTM [133]	convolutional LSTM

TABLE 3: Acronyms for loss functions and evaluation metrics.

acronym	full name (a.k.a. or closely similar name)
AAD	average angle difference
ACPPE	average critical point position error
AEDR [121]	adaptive edit distance on real sequence
AER	average error rate
ALP	average last position
AWL [151]	adaptive wing loss
CD^1 [5]	chamfer distance
CD^2	cosine distance
	(cosine dissimilarity, cosine similarity)
CE	cross-entropy, cross-entropy loss
	(logarithmic loss, logistic loss, log loss)
BCE	binary CE
CR	compression rate, compression ratio
	(reduction rate)
EMD [124]	earth mover's distance
	(Wasserstein distance, Wasserstein metric)
F-score	F-1 score
	(Dice similarity coefficient, Sørensen-Dice coefficient)
FID [66]	Fréchet inception distance
FN	false negative
FP	false positive
FPR	FP rate
	(fall-out)
GSAL [130]	geometric structure-aware loss
HD	Hausdorff distance
HR	hit ratio
IOU	intersection over union
IS [18]	isosurface similarity
Jaccard	Jaccard index in binary classification
	(visual object class)
LPIPS [175]	learned perceptual image patch similarity
LSiM [89]	learned simulation metric
MAE	mean absolute error
	(L1 error, L1 loss, L1 norm)
RAE	root absolute error
MCPD [29]	mean of the closest point distances
ME	mean error
MI	mutual information
MOS	mean opinion score
MSE	mean squared error
	(L2 error, L2 loss, L2 norm)
RMSE	root MSE
PPV	precision
	(positive predictive value)
PSNR	peak signal-to-noise ratio
REC [14]	regression error characteristic
ROR	recall over rank
SC	silhouette coefficient
	(silhouette score)
SSIM [157]	structural similarity index
	(structural dissimilarity index)
TN	true negative
TP	true positive
TPD	time partial derivative
TPR	true positive rate
	(sensitivity, recall, hit rate)

of the other five dimensions aims to summarize and categorize these works from crucial aspects of design and evaluation for a comprehensive understanding. In Tables 2 and 3, we list acronyms that are used throughout this paper, where the indents represent hierarchical relationships.

3.1 Domain Settings

We group the surveyed works based on their *original* forms of data in their domain settings. The five settings are *scalar*, *vector*, *fluid simulation*, *particle*, and *image*. Note that the original data in such a setting is not necessarily the input to the DL model. For example, Berger et al. [12] designed a GAN model for volume rendering, where the domain setting is a volumetric scalar field, and the input to the DL model is the viewpoint and transfer function.

Table 1 shows that more than half of the papers handle scalar field data, including a single volume (e.g., [182]), time-varying (e.g., [51]), and multivariate (e.g., [56]) volumetric data. This is not surprising as scalar field data are most commonly produced and widely available in SciVis. Apart from scalar field data, more than a quarter of the papers tackle vector field data, include 2D and 3D steady (e.g., [11], [49]) and unsteady (e.g., [45], [85]) vector fields. Two papers (i.e., [46], [52]) cover scalar and vector domains. We single out seven works (i.e., [26], [27], [84], [162], [163], [164], [168]) and label them in the category of fluid simulation, as the primary focus of these works is simulation. Finally, two works (i.e., [97], [103]) target particle data, and the remaining two (i.e., [15], [183]) deal with image data.

3.2 Research Tasks

Along the research task dimension, we classify the surveyed works into five categories: *data generation* [○ ● ● ●], *visualization generation* [●], *prediction* [○ ●], *objection detection and segmentation* [●], and *feature learning and extraction* [●], as shown in Table 1. These research tasks are often the end goals of their respective works. However, in some cases, they only solve a subproblem, setting up a critical step to their final solution (e.g., [24], [26], [120], [183]).

Data generation tasks produce or reconstruct data or models from low-resolution versions (e.g., [182]), feature representations (e.g., [49]), or lower-dimensional counterparts (e.g., [156]) [○ ●]. They also address data translation (e.g., [56]) [●] (e.g., translating from one variable sequence to another or from an input field to another that satisfies specific properties or constraints) and extrapolation (e.g., [163])

TABLE 4: Neural networks' inputs and outputs for data generation [○ ○ ● ●] papers.

paper	name	input	output
Zhou et al. [182]		low-resolution volume	super-resolution volume
Han and Wang [51]	TSR-TVD	two end volumes	intermediate volumes
Han and Wang [50]	SSR-TVD	low-resolution volume	super-resolution volume
Han et al. [55]	STNet	low-resolution two end volumes	super-resolution intermediate volumes
Wurster et al. [167]		low-resolution volume	multi-resolution super-resolution volume
Guo et al. [47]	SSR-VFD	low-resolution field	super-resolution field
Jakob et al. [76]		low-resolution flow map	super-resolution flow map
Sahoo and Berger [126]	IA-VFS	low-resolution field	super-resolution field
An et al. [2]	STSRNet	low-resolution two end fields	super-resolution intermediate fields
Han and Wang [53]	TSR-VFD	two end volumes	intermediate volumes
Xie et al. [168]	tempoGAN	low-resolution volume	super-resolution volume
Werhahn et al. [162]		low-resolution volume	super-resolution volume
Wang et al. [156]	DeepOrganNet	3D/4D-CT projection or X-ray image	3D/4D lung model
Lu et al. [109]	neurcomp	voxel coordinate	scalar value
Weiss et al. [160]	fV-SRN	voxel coordinate	density or color
Shi et al. [131]	GNN-Surrogate	simulation parameters	output field with adaptive resolution
Han and Wang [54]	VCNet	incomplete volume	complete volume
Liu et al. [106]		volume patch	feature vector
Han et al. [49]		low-resolution field	high-resolution field
Gu et al. [45]	VFR-UFD	low-quality fields	high-quality fields
Han et al. [56]	V2V	source variable	target variable
Gu et al. [46]	Scalar2Vec	scalar field volume	vector field volume
Kim et al. [84]	Deep Fluids	velocity vector field, solver parameters	divergence-free velocity field
Chu et al. [27]		density field	velocity field
Wiewel et al. [163]	LSP	multiple steps of pressure fields	future steps of pressure fields
Wiewel et al. [164]	LSS	early timesteps of simulation	future timesteps of simulation

[●] (e.g., generating historical or future data given the current data) issues.

Visualization generation tasks synthesize rendering results conditioned on the input, such as viewpoint (e.g., [70]), transfer function (e.g., [12]), and other parameters (e.g., [63]). Their goal is to produce unseen visualization results given the new input parameters without going through the traditional rendering pipelines or rerunning simulations under different parameter settings.

Prediction tasks estimate class memberships, quality scores, future values, etc. The prediction results can assist users in making selections (e.g., [143]), recommendations (e.g., [169]), or planning (e.g., [71]) accordingly. These predictions can be data-relevant [○] (e.g., voxel value) or visualization-relevant [●] (e.g., viewpoint quality).

Objection detection and segmentation tasks are common in CV and biomedical imaging. The goal is to detect objects (e.g., animals, pedestrians, vehicles) from image or video data or segment different components (e.g., heart, lung, vessels) from biomedical data. Researchers also classify vortex boundaries [11] or vortical structures [81] from flow data.

Finally, feature learning and extraction tasks learn data representations in the latent space (e.g., [48]) and extract domain-specific features (e.g., [64]). The learned representations can guide downstream tasks such as clustering, filtering, and representative selection.

From Tables 4 to 8, we list the neural networks' inputs and outputs of these papers (one for each category) following the same order as shown in Table 1. The input and output refer to the inference stage whenever applicable. In the following, we describe these papers in detail.

3.2.1 *Data Generation* [○ ○ ● ●]

Under the category of data generation, we further group the related papers into four subcategories: *super-resolution*, *compression and reconstruction*, *translation*, and *extrapolation*.

Super-resolution [○]**.** Super-resolution is a class of techniques that aim to enhance or increase the image, video, or volumetric data resolution. In SciVis, the resolution encompasses the three spatial dimensions and the temporal dimension. Due to the limited storage space, generating super-resolution data from their low-resolution counterparts brings the immediate benefit of storage-saving via data reduction. This is because only the low-resolution data and the trained network model need to be stored to recover the super-resolution data. DL-based super-resolution techniques thus provide domain scientists an alternative to manage their simulation data cost-effectively.

For scalar field data, the work of Zhou et al. [182] is the first known one that utilizes a CNN for volume upscaling. Their CNN pipeline includes three stages: block extraction and feature representation, non-linear mapping, and reconstruction. They could upscale a single scalar volume by a factor of 2 (i.e., the super-resolution volume is 8× the size of the low-resolution input volume). To push the limit of data reduction, Wurster et al. [167] proposed a hierarchical super-resolution solution for volumetric data reduction. The resulting neural network hierarchy enables multi-resolution super-resolution generation at varying scaling factors (from 2× to 64×). Finally, their octree-based data representation solution can upscale multi-resolution data to a uniform resolution while minimizing artifacts along block boundaries. Han and Wang considered time-varying volumetric data and presented TSR-TVD [51] and SSR-TVD [50] for generating temporal super-resolution (TSR) and spatial super-resolution (SSR). Both works leverage GANs for network training and consider temporal coherence. TSR-TVD can achieve a maximal interpolation step of 11, while SSR-TVD can upscale the volumes by a factor of 4. More recently, Han et al. [55] introduced STNet, an end-to-end generative framework that achieves simultaneously spatiotemporal super-resolution (STSR) for time-varying volume data. They

argued that straightforward concatenating SSR and TSR solutions does not lead to high-quality STSR volumes due to error propagation and presented STNet results superior to those of SSR+TSR.

For vector field data, Guo et al. [47] designed SSR-VFD to upscale a 3D vector field by a scaling factor of 4 or 8. Due to the possibly vast differences among vector components, they employed three neural nets to train individual components. These networks jointly generate spatially coherent super-resolution of vector field data. Sahoo and Berger [126] presented IA-VFS, a solution for integration-aware vector field super-resolution. IA-VFS follows SSR-VFD [47] but considers integral streamlines in loss function to improve network optimization. Han and Wang [53] developed TSR-VFD that generates intermediate vector fields from temporally sparsely sampled vector fields. TSR-VFD utilizes two networks: InterpolationNet and MaskNet, to learn different scales of the data and can achieve a maximal interpolation step of 9. An et al. [2] proposed STSRNet, a joint space-time super-resolution framework for vector field visualization. STSRNet includes two stages: the first one synthesizes intermediate frames given the two end frames at the low spatial resolution, and the second one upscales these intermediate frames to high spatial resolution. Jakob et al. [76] adopted different versions of CNN to generate super-resolution flow maps, which are fundamental to Lagrangian transport analysis. They also provided the community with a large numerical 2D fluid flow dataset for ML.

For fluid simulation, Xie et al. [168] introduced tempoGAN, a generative solution that produces temporally coherent volumetric super-resolution fluid flow. Their newly designed temporal discriminator can yield consistent and high-quality temporal results. Later works such as SSR-TVD [50] follow this design. Werhahn et al. [162] presented a multi-pass GAN for generating fluid flow super-resolution. Their solution takes several orthogonal passes to decompose the space-time generative task so that each pass can solve an easily manageable inference problem.

Compression and reconstruction [●]**.** Another subcategory of data generation tasks focuses explicitly on data compression and reconstruction. For data compression, Lu et al. [109] followed the SIREN [136] design and introduced neurcomp, a coordinate-based MLP for compressive neural representations of scalar field volume data. Once learned, the network itself becomes the compressed representation of the underlying data. By quantizing network weights, neurcomp could achieve an impressive CR over 1,000× while preserving important volumetric features. Weiss et al. [160] presented fV-SRN, which improves neurcomp by leveraging GPU tensor cores to integrate the reconstruction task into on-chip raytracing kernels. They also supported random access reconstruction at arbitrary granularity for temporal reconstruction tasks. Liu et al. [106] developed an in situ compression technique based on GAN for computational fluid dynamics (CFD) data. Compared with discrete wavelet transform, their solution achieves a speedup of over 3× in compression time while allowing a tradeoff between CR and reconstruction accuracy.

For data reconstruction, Wang et al. [156] designed DeepOrganNet that reconstructs 3D/4D lung models from single-view CT projections or X-ray images. DeepOrganNet can reconstruct manifold meshes of lung models in high quality and high fidelity, which all previous DL-based shape reconstruction approaches cannot. Shi et al. [131] developed GNN-Surrogate that reconstructs simulation outputs given simulation parameters as input. They generated graph hierarchies from unstructured grids and used the cutting policy to steer the representation of the simulation output with adaptive resolutions. Han and Wang [54] designed VCNet for volume completion. The GAN-based neural network can synthesize missing subvolumes of various shapes (e.g., cuboid, cylinder, hyperboloid, sphere, tetrahedron, and ring) and sizes (up to 50% of the entire volume). Han et al. [49] addressed the issue of reconstructing vector field data from representative streamlines. Their solution encompasses two steps: initializing a low-resolution field based on the input streamlines and upscaling the low-resolution field using a CNN. Gu et al. [45] extended the work of Han et al. [49] to reconstruct unsteady vector fields from their streamline representations via diffusion and DL-based denoising. Their solution captures temporal coherence by considering multiple consecutive timesteps at a time and preserves spatial coherence through streamline-based network optimization.

Translation [●]**.** In SciVis, DL-based data translation was inspired by image colorization [25] and image-to-image translation (e.g., Pix2Pix [75], CycleGAN [184]). Han et al. [56] designed a DL solution for variable-to-variable (V2V) translation in the context of multivariate time-varying volumetric data. Their work first utilizes U-Net to learn features from variable sequences and identify suitable pairs for translation. Then, V2V leverages a GAN to achieve the translation (i.e., inferring the target variable sequence given the source variable sequence). Gu et al. [46] presented Scalar2Vec that translates scalar fields to vector fields via DL. They followed the same approach as Han et al. [56] to pick suitable scalar variables for the translation. The CNN-based network takes a set of sampled scalar field volumes as input and extracts their multi-scale information to synthesize the corresponding vector field volumes. Kim et al. [84] developed Deep Fluids for parameterized fluid simulations. Their generative model uses physics-informed loss functions to generate divergence-free velocity fields given the input velocity vectors and solver parameters. Chu et al. [27] aimed to infer velocity fields from density fields (i.e., translating density fields to velocity fields) via a data-driven cGAN model [115]. Their work also provides multiple controls, such as physical parameters and kinetic energy, for fluid generation. We point out that some of these works (i.e., [27], [56]) also imply data reduction. The trained neural nets can infer a previously unseen target variable sequence of later timesteps from the corresponding source sequence or a vector field from a scalar field, thus omitting the need to store the target variable or vector field.

Extrapolation [●]**.** Extrapolation aims to generate historical or future data values based on the current values. For fluid simulation, Wiewel et al. [163] presented latent space physics (LSP), an LSTM-CNN hybrid approach to predict the changes of pressure fields over time for fluid flow simulation. LSP can achieve 150× speedups compared with a regular pressure solver, a significant boost in simulation performance. Wiewel et al. [164] proposed latent space

TABLE 5: Neural networks' inputs and outputs for visualization generation [○] papers.

paper	name	input	output
Berger et al. [12]		new viewpoint and transfer function	synthesized rendering conditioned on input
Hong et al. [70]	DNN-VolVis	original rendering, goal effect, new viewpoint	synthesized rendering conditioned on input
He et al. [63]	InSituNet	ensemble simulation parameters	synthesized rendering conditioned on input
Weiss et al. [159]		low-resolution isosurface maps, optical flow	high-resolution isosurface maps
Weiss et al. [161]		low-resolution image	high-resolution image
Weiss and Navab [158]	DeepDVR	volume, viewpoint	rendering image

TABLE 6: Neural networks' inputs and outputs for prediction [○ ●] papers.

paper	name	input	output
He et al. [62]	CECAV-DNN	sequence of ensemble pairs	likelihood each member from one ensemble
Tkachev et al. [143]		local spatiotemporal patch	future voxel value at patch center
Hong et al. [71]		movement sequence	probability vector of next movement
Kim and Günther [85]		unsteady 2D vector field	reference frame transformation
Han et al. [57]		particle start location, file cycles	particle end location
Yang et al. [169]		volume rendering under viewpoint	viewpoint quality score
Shi and Tao [130]		volume rendering image	estimated viewpoint
Engel and Ropinski [35]	DVAO	intensity volume, opacity volume or transfer function	AO volume

subdivision (LSS), an end-to-end DL-solution for robust prediction future timesteps of complex fluid simulations with high temporal stability. Using CNN and stacked LSTM, LSS achieves both spatial compression and temporal prediction.

3.2.2 *Visualization Generation* [○]

All existing DL4SciVis works in visualization generation are related to either direct volume rendering or isosurface rendering. Berger et al. [12] presented a generative model for volume rendering where a GAN was trained on a large collection of volume rendering images under different viewpoints and transfer functions. Once trained, the model can infer novel rendering conditioned on new viewpoints and transfer functions without following the traditional rendering pipeline. Hong et al. [70] designed DNN-VolVis, a DNN for volume visualization. Their goal is to synthesize volume rendering results from the original input rendering under a given target effect and new viewing parameters. Thus, without knowing the underlying transfer function, their generative framework supports volume exploration in a reverse manner. Weiss and Navab [158] trained DeepDVR, an end-to-end DNN that explicitly models the direct volume rendering process, including feature extraction, classification, and composition. Their solution generates similar direct volume rendering results from examples in the image space, eliminating the need for explicit feature design and manual transfer function specification.

He et al. [63] developed InSituNet for parameter-space exploration of ensemble simulations. The training data (i.e., visualization images conditioned on visual mappings and view parameters) were collected in situ. Then, they trained a convolutional regression model offline that learns the mapping from simulation parameters to visualization outputs. The trained model supports interactive post hoc exploration and analysis by synthesizing images from novel parameter settings. Weiss et al. [159] generated super-resolution isosurface rendering images from their low-resolution counterparts using FRVSR-Net (a fully convolutional frame-recurrent neural network). The network takes low-resolution isosurface maps (mask, normal, and depth) and optical flow as input and outputs high-resolution isosurface maps, including mask, normal, depth, and ambient occlusion (AO) maps. Weiss et al. [161] aimed to learn the correspondence between the data, sampling patterns, and generated images. To achieve the goal, they introduced an end-to-end neural rendering framework consisting of two networks (i.e., importance network and reconstruction network). The former infers the importance map from low-resolution rendering images, and the latter recovers high-resolution images from sparse samples.

3.2.3 *Prediction* [○ ●]

Under the category of data prediction, we further group the related papers into two subcategories: *data-relevant prediction* and *visualization-relevant prediction*.

Data-relevant prediction [○]**.** For scalar field data, He et al. [62] designed CECAV-DNN that predicts ensemble similarity for collective ensemble comparison and visualization (CECAV). Given a sequence of ensemble pairs (each ensemble is a collection of scalar fields), they trained the DNN to assign a likelihood score to each scalar field, indicating the probability that the field is from one ensemble rather than the other. After training, three levels of comparison: dimensionality comparison, member comparison, and region comparison, are provided for ensemble comparison and visualization. Tkachev et al. [143] developed a local prediction model for spatiotemporal volume visualization. Their goal is to detect irregular processes (i.e., outliers) in the space-time data. To this end, they designed a neural network that takes local spatiotemporal patches and predicts future voxel values at patch centers. The predicted values' deviation from the ground-truth values suggests mispredicted spatiotemporal regions for further study.

For vector field data, Hong et al. [71] aimed to predict the access pattern for parallel particle tracing. Their LSTM-based model learns the access pattern from a small set of pathline samples. Such prediction results can assist workload balancing by prefetching data blocks to reduce I/O costs and improve time efficiency. Kim and Günther [85] designed a CNN to predict reference frame transformations for 2D unsteady vector fields. Their solution can tackle noisy inputs and data with resampling artifacts by performing filtering and reference frame extraction end-to-end. Han et al. [57] predicted particle end locations for Lagrangian-based particle tracing. Their MLP-based model learns particle end locations given their start locations and file cycles.

TABLE 7: Neural networks' inputs and outputs for object detection and segmentation [●] papers.

paper	name	input	output
Wang et al. [154]	VC-Net	3D volume patch, multislice composited 2D MIP	vessel mask
Nguyen et al. [116]		cryo-EM image, dense pseudo labels	soft labels
Ghahremani et al. [40]	NeuroConstruct	batch of grayscale images	probability map
He et al. [65]		super-voxel graph with neighborhood relations	feature classification per super-voxel
Deng et al. [31]	Vortex-Net	sample local patch	hard labels
Berenjkoub et al. [11]		velocity patch	binary classification of vortex boundary
Kashir et al. [81]		input map (velocity, vorticity)	binary segmentation of vortical structure
Borkiewicz et al. [15]	CloudFindr	image patch	predicted mask

The trained model can predict new particle trajectories with a small memory cost and fast inference.

Visualization-relevant prediction [●]. This subcategory of prediction tasks estimates visualization-related quality or parameters. Yang et al. [169] designed a CNN-based model to estimate the viewpoint quality given a volume rendering image. The aim is to mimic the traditional scoring method and user preference and predict viewpoint quality close to human judgment. Shi and Tao [130] attempted to estimate the viewpoint given a volume rendering image. Their CNN-based viewpoint estimation framework features an overfit-resistant image rendering strategy for training data generation and a geometric structure-aware loss design. Engel and Ropinski [35] presented DVAO, deep volumetric AO, that predicts the AO volume given the original intensity volume and opacity information specified in the form of opacity volume or transfer function descriptor. DVAO supports real-time volume interaction with per-voxel AO estimation.

3.2.4 *Object Detection and Segmentation* [●]

Existing DL4SciVis works on object detection and segmentation heavily utilize U-Net [122], initially designed for biomedical image segmentation. For scalar field data, Wang et al. [154] designed VC-Net, a deep volume composition network for segmenting vessels from highly sparse and noisy biomedical image data. Their paradigm includes a dual-stream component and the bi-directional operations between them. The 3D volume segmentation stream follows a 3D U-Net design, and the 2D composited maximum intensity projection (MIP) segmentation stream uses a half 2D U-Net. To achieve effective exploration, VC-Net combines direct 3D volume processing (3D stream) and volume-rendered clues (2D stream). Nguyen et al. [116] presented a semi-supervised volume visualization solution for cryo-EM data. Their solution includes two segmentation algorithms (a weak one and a powerful DL-based one) to produce soft segmentation results guiding the transfer function design. They compared three models: 3D U-Net, 3D U-Net+ResNet, and 3D DenseNet, along with three losses: BCE, MSE, and AWL, and reported that 3D U-Net+ResNet with MSE loss works best. Ghahremani et al. [40] developed NeuroConstruct to reconstruct 3D neurites from optical microscopy brain images. Their 3D CNN-based segmentation model consists of multiple stages of residual U-block (RSU) connected in the big U-structure.

For vector field and image data, Deng et al. [31] designed Vortex-Net, a CNN-based method for vortex identification. Their binary classification solution benefits from global and local vortex identification methods to achieve better performance (both speed and accuracy). Berenjkoub et al. [11] presented a CNN to identify vortex boundary. They experimented with three CNN architectures: conventional CNN, ResNet, and U-Net, and reported that U-Net achieves the best performance in the binary classification task. Kashir et al. [81] utilized an FCN to identify vortical structures in 2D fluid flow. The model takes velocity and vorticity maps as input and produces pixel-wise semantic segmentation results. In addition, they investigated the symmetric U-shaped network structure to find the optimal settings to best extract vortical structures. Borkiewicz et al. [15] developed CloudFindr to detect cloud from satellite image data. Their U-Net-based solution produces a predicted mask given the input image patch.

Beyond U-Net, researchers also explored the use of GNN for volume classification. He et al. [65] generated a super-voxel graph from a scalar volumetric dataset where a node represents a super-voxel (i.e., a group of voxels with similar spatial locations and properties), and an edge represents the neighborhood relation between the corresponding super-voxels. They then utilized a GCN to learn node embedding. Finally, the output of the GCN goes through an MLP to predict the label of each node for volume classification.

3.2.5 *Feature Learning and Extraction* [●]

For scalar field data, many solutions use CNN-based neural networks for feature learning and extraction. Raji et al. [120] trained a deep Siamese network to extract feature vectors from image pairs (real-world photographs and volume rendering images) and judge whether nor not the input images are similar. The goal is to optimize rendering parameters via an evolutionary process to match the features of rendering images with those in the photographs. Cheng et al. [24] applied a pre-trained CNN to learn voxel neighborhood information. They then employed vector quantization to the high-level features extracted from volume patches to generate the characteristic feature vector to support the hierarchical exploration of complex volumetric structures. Porter et al. [117] leveraged an AE to encode each timestep of a time-varying volumetric dataset into a feature vector, which was then projected to an abstract 2D space for identifying representative timesteps. Their approach can naturally handle multivariate datasets using a multichannel input which previous works cannot. Tkachev et al. [144] designed S4, self-supervised learning of spatiotemporal similarity, for supporting explicit similarity queries of scientific datasets. They employed a Siamese network to extract feature vectors from local spatiotemporal patches and judge if they are from the same neighborhood.

Beyond CNNs, researchers have also investigated GNN-based solutions for feature learning and extraction. He et al. [64] designed ScalarGCN, a GNN-based solution for scalar-value association analysis of volumes. ScalarGCN aims to

TABLE 8: Neural networks' inputs and outputs for feature learning and extraction [●] papers.

paper	name	input	output
Raji et al. [120]		image pairs	feature vectors
Cheng et al. [24]		volume patch	feature vector
Porter et al. [117]		volume	feature vector
Tkachev et al. [144]	S4	local spatiotemporal patches	feature vectors
He et al. [64]	ScalarGCN	Scalar-Graph with local and global connections	feature vector per variable
Han et al. [48]	FlowNet	streamline or stream surface	feature vector
Han and Wang [52]	SurfNet	isosurface or stream surface	node features
Chu and Thuerey [26]		flow patch pairs	feature vectors
Liu et al. [103]		density field	feature vector
Li and Shen [97]		particle patch	feature vector
Zhu et al. [183]		feature position in initial scatterplot	inferred feature position in new scatterplot

learn the high-order topological structural relationships of multiple variables using a multilayer GCN with the self-attention mechanism. The input to ScalarGCN is Scalar-Graph, where nodes represent sampled scalar values from multivariate data, and edges encode local (within the same variable) and global (across different variables) connections. Node features consider context, spatial, and gradient distributions. ScalarGCN performs local learning for node embedding and global learning for variable embedding.

For vector field data and fluid simulation, Han et al. [48] introduced FlowNet, which is an AE for learning the latent features of streamlines and stream surfaces implicitly. They compared three losses: BCE, Dice, and MSE, and reported that BCE achieves the best results. The learned features are projected into a low-dimensional space to support clustering, filtering, and selection of representatives. Their voxel-based representation makes the solution generally applicable to any 3D data or their visual representations (e.g., [117]), albeit not necessarily efficient when the representations are sparse in the 3D domain. Han and Wang [52] designed SurfNet, a GCN-based solution for learning node and surface features for isosurfaces and stream surfaces. Compared with FlowNet [48], training SurfNet is 10× to 20× faster per epoch and inferring is 70× to 170× faster while the model reduction is 300× to 1,300×. Chu and Thuerey [26] employed two identical CNNs similar to the Siamese network to extract feature vectors from two flow patches: a coarse approximation and a refined version of two flow simulations of the same effect. Their end goal is to identify the best-matched flow patch from a fluid repository of pre-computed data to refine a new coarse input for volumetric synthesis.

For particle and image data, Liu et al. [103] investigated the use of a residual AE for feature learning of a particle dataset. Their goal is to achieve in situ data reduction that preserves features (i.e., gas bubbles in a fluid). Li and Shen [97] leveraged a Geo-CNN [93] to extract features from particle data that capture their spatial-physical attribute relationships without explicitly knowing their spatial connectivity information. The learned feature information was utilized in the subsequent feature tracking. Zhu et al. [183] designed a cascade neural network that takes hyperspectral image features as input and infers a scatterplot where domain experts customized the cluster centers. Once trained, the same network can be used for studying time-varying hyperspectral images without retraining.

3.2.6 Summary

Comparing the five research tasks, we see many more works in data generation [● ● ● ●] than visualization generation [●]. We reason that data generation tasks in SciVis share significant similarities with those in CV, making it relatively easier to work on even with the apparent challenge of handling 3D volume data instead of 2D image data. On the other hand, visualization generation tasks must consider different parameters (e.g., viewpoint, transfer function, ensemble parameters, etc.) and demand new solutions to assimilate such heterogeneous information into network design and training. Recent advances in CG, such as neural rendering [139] and differentiable rendering [82], provide good opportunities for SciVis researchers to expand the current research in visualization generation.

Within the category of data generation, there are more super-resolution [●] works than compression and reconstruction [●], translation [●], and extrapolation [●]. We can contribute this difference to the fact that data reconstruction may consider their visual representations as input, data translation adds extra complexity from multivariate relationships and multichannel variations, and data *extrapolation* is intrinsically more challenging than super-resolution (which can be treated as a form of data *interpolation*). Reconstruction, translation, and extrapolation could become a growth point for future research in data generation.

Feature learning and extraction [●] is a resounding theme for CV, CG, and VIS. In SciVis, feature definitions are usually application-specific, and in many cases, they are vague or even unknown. Therefore, explicitly or implicitly, learning features is the necessary first step toward effective analysis and visualization. By replacing manual feature engineering with automatic feature discovery, representation learning can help accomplish a wide variety of subsequent tasks critical to SciVis, such as dimensionality reduction, data clustering, representative selection, anomaly detection, data classification, and data generation. Due to the general need and the variety of data (scalar and vector, time-varying and multivariate) and their visual representations (line, surface, volume), we expect a strongly growing trend in DL-based solutions for feature or representation learning.

Prediction tasks [● ●] commonly serve as an intermediate step of a large problem which yields critical prediction results to assist downstream tasks. This makes unsupervised learning, particularly self-supervised learning, a suitable candidate for accomplishing such tasks. Thus, investigating the underexplored self-supervised learning solutions for making predictions or recommendations will certainly boost

TABLE 9: All surveyed papers and their learning types organized under the five research tasks. The * sign indicates the work includes a pre-training step. The + sign indicates the work uses a pre-trained model.

learning	task				
	data gen [○ ○ ● ●]	vis gen [○]	prediction [○ ●]	obj det & seg [●]	feat lrn & ext [●]
supervised	○ [2], ● [27], ● [45], ● [46], ○ [47], ● [49], ○ [50] ○ [51], ○ [53], ● [54], ○ [55]*, ● [56], ○ [76], ● [84] ○ [126], ● [131], ● [156]+, ○ [160], ○ [162] ● [163], ● [164], ○ [167], ○ [168], ○ [182]	[12], [63] [70], [158] [159] [161]	● [35], ○ [57] ○ [71], ○ [85] ● [130], ○ [143] ● [169]	[11], [15] [31], [40] [81], [154]	[24]+
weakly-supervised	—	—	—	—	—
semi-supervised	—	—	—	[65] [116]*	[183]
unsupervised					
distributed	● [106], ● [109]	—	○ [62]	—	[48], [52], [64] [97], [103], [117]
disentangled	—	—	—	—	—
self-supervised	—	—	—	—	[26], [120], [144]

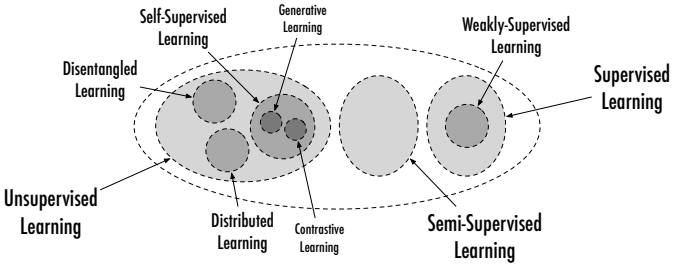


Fig. 1: The relationships among different learning types.

DL4SciVis research.

Finally, object detection and segmentation papers [●] are often published in CV and biomedical imaging venues, even though medical visualization is a long-standing topic in SciVis. Nevertheless, we see that popular networks such as U-Net, initially designed for biomedical image segmentation, have been widely adopted and utilized for solving SciVis problems.

3.3 Learning Types

As illustrated in Figure 1, depending on how much labeled data are used for neural network training, DL tasks can be *supervised* (full labels), *semi-supervised* (partial labels), or *unsupervised* (no labels) [77]. As a subset of supervised learning, *weakly-supervised* learning learns feature representations with coarse-grained or inaccurate labels. Under the umbrella of unsupervised learning, there are three types of learning: *distributed learning*, *disentangled learning*, and *self-supervised learning*.

Distributed learning denotes the same data features across multiple *scalable* and *interdependent* layers, which are learned concurrently but non-linearly. Each layer describes the information with the same accuracy level while adjusted for the scale level.

Unlike distributed learning, disentangled learning represents the data with *independent* features, each describing partial information, such as *content* and *style*. It is possible to learn disentangled representations from distributed representations by appropriate transformations.

Self-supervised learning automatically generates the labels from the underlying data itself by leveraging its structure [77]. This learning approach consists of two stages. The first stage trains unlabeled data in a *pretext task* (e.g., jigsaw puzzle) to generate representations. The second stage applies these representations to a *downstream task* (e.g., classification or segmentation) using only a small amount of labeled data. Self-supervised learning can be *generative* or *contrastive* [105]. Generative learning learns representations from data by fitting the data distribution. Contrastive learning aims at "learning to compare" through a noise contrastive estimation objective.

Table 9 classifies all surveyed papers into different learning types. Overall, we can see that supervised learning is dominant across the four categories of research tasks: data generation [○ ○ ● ●], visualization generation [○], prediction [○ ●], and object detection and segmentation [●]. Supervised learning is common for data or visualization generation tasks as the ground-truth data or visualizations are usually provided for loss computation during training. It is also often used for prediction and object detection and segmentation tasks as the ground-truth results (e.g., future values, user-voted quality scores, segmentation masks) are given for network training. On the contrary, unsupervised learning is mostly applied for feature learning and extraction [●]. AEs are often utilized for implicit feature learning from the input data in an unsupervised manner. In this case, these works exclusively belong to distributed learning, to be precise.

Along the research task dimension, visualization generation tasks [○] are exclusively supervised. Data generation [○ ○ ● ●], prediction [○ ●], and object detection and segmentation [●] each occupy two learning types. Under the data generation category, two exceptions (i.e., [106], [109]) of compression and reconstruction [●] fall into the category of distributed learning (a subtype of unsupervised learning). Under the prediction category, one exception (i.e., [62]) of data-relevant prediction [○] falls into distributed learning. Under the object detection and segmentation category, there are two exceptions (i.e., [65], [116]) of semi-supervised learning. Finally, feature learning and extraction tasks [●] are most diverse across the learning types, covering all but weakly-supervised and disentangled learning.

Only six examples fall into the categories of semi-supervised (i.e., [65], [116], [183]) and self-supervised (i.e., [26], [120], [144]) learning. Under self-supervised learning, all three works (i.e., [26], [120], [144]) are contrastive learning, not generative learning. Furthermore, two works (i.e., [55], [116]) employ pre-training (which aims to improve the network's generalization ability from the training datasets). Finally, two works (i.e., [24], [156]) use the pre-trained models directly to infer feature vectors from data. We find no existing DL4SciVis works in the disentangled learning, generative learning, and weakly-supervised learn-

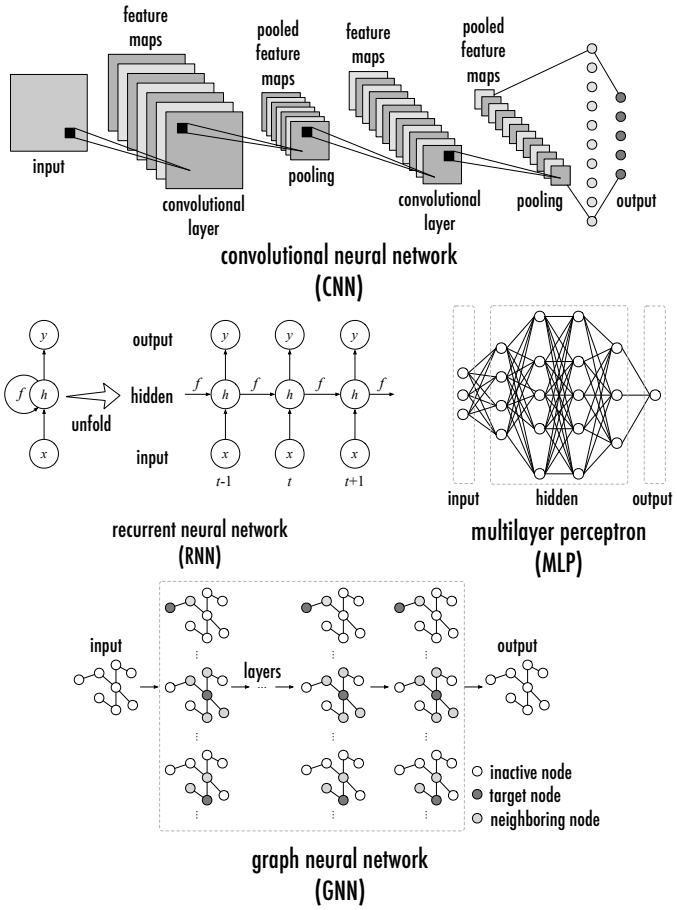


Fig. 2: Illustrative examples of CNN (redrawn from [94]), RNN (adapted from [41]), MLP, and GNN network architectures.

ing types. This may be due to the challenges of defining and interpreting content and style for SciVis data, the high training and memory cost for the generative tasks, and the lack of weakly-supervised learning scenarios for SciVis, respectively.

3.4 Network Architectures

We categorize all surveyed papers based on their respective network operations and structures. From the network operation or connection perspective, there are four basic ones: *convolution operation*, *full connection*, *recurrence operation*, and *graph connection*, which lead to CNN, MLP, RNN, and GNN, respectively (refer to Figure 2). CNNs [94] operate on grid-like data (e.g., images or volumes), where the convolution operation uses a convolution kernel or filter that slides along the input data to generate feature maps. The hidden layers of a CNN include several stages of convolutional and pooling layers, followed by one or several fully-connected layers. The *convolutional layer* detects local combinations of features from the previous layer. The *pooling layer* merges semantically similar features into one by computing a summary (e.g., maximum). The *fully-connected layer* learns global features from local ones by connecting neurons to all activations in the previous layer. RNNs [125] operate on sequence data, where network connections between nodes form a directed graph along a temporal sequence. One of the most frequently used deep RNN architectures is LSTM [68], which was developed to address the vanishing gradient problem in traditional RNNs by adding forget gates in the units. MLPs [123] are networks composed of multiple layers of perceptrons. The input layer, one or multiple hidden layers, and the output layer are all fully connected (i.e., every node in one layer connects with a certain weight to each node in the following layer). Finally, GNNs [43] naturally operate on graph-like data (e.g., surfaces). As a generalized version of CNN, GCN [88] works on data with underlying non-regular structures. Each layer loops each node as a target node and applies a convolution operation on its neighboring nodes to learn node features.

From the network structure perspective, there are three basic ones: *encoder*, *decoder*, and *encoder+decoder*. An encoder is a neural network that takes the data or rendering as input and outputs a reduced representation (e.g., feature vector) or estimated information (e.g., viewpoint) in a designated format. A decoder is also a neural network, usually the same as the encoder but in the opposite orientation. The decoder takes the coded message (i.e., reduced representation or estimated information) as input and outputs the data or infers the rendering result. An encoder+decoder includes both an encoder and a decoder. In the AE, the encoder is trained with the decoder. The output feature vector from the encoder implicitly represents the information of the input data in the latent space. The decoder takes the feature vectors as input and produces the reconstructed data best matching the original data. In other settings, it can complete a different task other than data reconstruction, such as super-resolution generation or data translation. The same technique has been used in various applications such as speech translation, generative models, etc.

Table 10 shows our categorization of all surveyed papers. In addition, other or specific networks (e.g., GAN, deformable CNN) employed different from their categorical names are provided in the table. As an exception, only three works (i.e., [84], [163], [164]) are placed in two categories due to their use of separate networks.

In terms of network operation or connection, we can see that CNN is the most popular category, followed by CNN+MLP. On the other hand, the least popular ones are GNN, CNN+RNN, MLP+RNN, and GNN+MLP. All four RNN-related works use only either LSTM [71], [163], stacked LSTM [164], or ConvLSTM [51]. The only four GNN-related works [52], [64], [65], [131] use GCN.

In terms of network structure, encoder+decoder is most popular, and encoder and decoder are similarly popular. Among these three structures, encoder is mostly used for prediction [○ ●] and feature learning and extraction [●] tasks, decoder is exclusively used for generation tasks (i.e., data generation [○ ● ●], visualization generation [○]), while encoder+decoder serves all different categories of tasks. These are expected due to the nature of their respective functional roles of network structures: encoder performs compression, decode performs decompression, and as the combination of encoder and decoder, encoder+decoder is capable of accomplishing a variety of tasks.

A close look at the categorization under CNN shows that encoder+decoder is the most diverse, covering all five research tasks, while decoder covers two tasks (i.e., data

TABLE 10: All surveyed papers and their neural network operations/connections and structures. The ▷ sign indicates the work adds a discriminator. Specific network names other than their category names, if available, are listed after each paper. Color coding is for data generation [○ ○ ● ●], visualization generation [●], prediction [● ●], object detection and segmentation [●], and feature learning and extraction [●].

operation/ connection	structure		
	encoder	decoder	encoder+decoder
CNN	● [130], ● [169]	○ [2](U-Net, deformable CNN), ○ [47] ● [49] ○ [50](GAN), ○ [55]▷(GAN) ○ [76](ESPCN), ● [106]▷(GAN), ○ [126] ● [159]▷(FRVSR-Net), ● [161](EnhanceNet) ○ [162]▷(multi-pass GAN) ○ [167]▷(ESRGAN, WGAN) ○ [168]▷(GAN), ○ [182]	● [11](CNN vs. ResNet vs. U-Net), ● [15](U-Net), ● [27]▷(GAN) ● [35], ● [40](nested encoder+decoder), ● [45], ● [46], ○ [53] ● [54]▷(GAN), ● [56]▷(GAN), ● [70]▷(GAN) ● [81](symmetric FCN), ● [84](AE), ● [103](residual AE) ● [116](U-Net vs. U-Net+ResNet vs. DenseNet) ● [117](AE), ● [154](multi-stream CNN) ● [158](U-Net, V-Net), ● [163], ● [164]
MLP	● [57], ● [183](FCCNN)	—	● [84], ● [109], ● [160]
RNN	● [164](stacked LSTM)	● [163](LSTM)	—
GNN	● [52](GCN), ● [64](GCN)	○ [131](GCN)	—
CNN+MLP	● [24], ● [26](Siamese) ● [31], ● [62], ● [85] ● [120](Siamese), ● [143] ● [144](Siamese), ● [156]	● [63]▷	● [12]▷(GAN), ● [48](AE) ● [97](Geo-CNN)
CNN+RNN	—	—	○ [51]▷(GAN+ConvLSTM)
MLP+RNN	● [71](LSTM)	—	—
GNN+MLP	● [65](GCN)	—	—

generation [○ ○ ●], visualization generation [●]), and encoder only covers the prediction task [●]. Many networks are either variants of CNNs or based on CNNs, including deformable CNN [152], DenseNet [73], EnhanceNet [127], ESPCN [132], FRVSR-Net [128], multi-stream CNN [154], ResNet [61], symmetric FCN [81], U-Net [122], and V-Net [114].

For the categorization under CNN+MLP, encoder is the most diverse, covering four research tasks (i.e., data generation [●], prediction [●], object detection and segmentation [●], feature learning and extraction [●]). In comparison, encoder+decoder covers two tasks (i.e., visualization generation [●], feature learning and extraction [●]), and decoder only covers the visualization generation task [●].

Explicitly set up to optimize for generative tasks, GAN covers both decoder and encoder+decoder categories, but not encoder. A GAN consists of two networks: a *generator* and a *discriminator*, which contest with each other in a zero-sum game. The generator maps from a latent space to a particular data distribution of interest. The discriminator discriminates between instances from the true data distribution and candidates produced by the generator. All works using a discriminator utilize GAN, except for one work [62] which employs a discriminative network instead of a GAN.

3.5 Loss Functions

An essential aspect of training a neural network lies in the design of the objective functions. Usually, the training goal is to minimize the objective functions over multiple iterations or epochs until the network converges. In this scenario, the objective function is often called the loss function (a.k.a. cost function or error function).

We categorize the optimization targets into five levels: *data*, *image*, *feature*, *probability*, and *parameter*. Data-level targets minimize the differences between model-inferred and ground-truth data (e.g., scalar or vector fields). The corresponding loss functions typically operate on individual voxels (3D data) or pixels (2D data) to accumulate the errors or compare the errors between two probability distributions (one from the mode-inferred data and another from the ground-truth data). Image-level targets minimize the differences between the synthesized and ground-truth rendering results (e.g., volume rendering, isosurface rendering). Like data-level targets, their corresponding loss functions loop through pixels or compare the probability distributions or statistical quantities (e.g., mean, standard deviation). Feature-level targets look to minimize the differences between features, properties, or attributes (e.g., streamlines, gradients, AO) derived from the inferred and ground-truth data. Probability-level targets aim to minimize the differences of predicted probabilities (e.g., segmentation masks) between the inferred and ground-truth results, typically in object detection and segmentation tasks. The loss functions often take the form of CE (multi-class classification or segmentation) or BCE (binary classification or segmentation), based on the framework of maximum likelihood. Finally, parameter-level targets directly minimize the neural network parameters during training. The corresponding loss functions are generally related to regularization terms or gradient penalties.

As shown in Table 11, we group all surveyed papers based on their optimization targets (organized in primary rows). For loss functions, we single out four categories: L1 (MAE), L2 (MSE), BCE, and Wasserstein, and leave all remaining ones under the category of "others".

L1 (MAE) and L2 (MSE) are the most popular loss functions. L1 loss minimizes the *absolute* differences between the inferred values and the ground-truth values, while L2 loss minimizes the *squared* differences between them. Because the difference between an incorrectly predicted value and the ground-truth value could be fairly large, squaring it would significantly amplify the difference. Therefore, compared with L2 loss, L1 loss is more stable and less susceptible to outliers.

For training a generative model using GANs, standard adversarial loss functions are based on L2, BCE, or Wasserstein. In the binary classification or segmentation setting, BCE compares the predicted probabilities with the actual binary class output and penalizes the probabilities if the distances from the expected ones are large. Unlike divergence-based loss functions (e.g., Kullback-Leibler divergence [91], Jensen-Shannon divergence [100]), Wasserstein loss [38] considers optimal transport by utilizing EMD as a natural

TABLE 11: All surveyed papers and their optimization targets and loss functions. Specific loss function names other than their category names, if available, are listed after each paper. In the "others" category, loss function names are listed before each paper. "adversarial" and "compression" are abbreviated as "adv" and "comp", respectively. Color coding is for data generation [○ ○ ● ●], visualization generation [●], prediction [○ ●], object detection and segmentation [●], and feature learning and extraction [●].

target	loss				
	L1 (MAE)	L2 (MSE)	BCE	Wasserstein	others
data	○ [2](magnitude) ○ [2](temporal), ● [56](feature) ● [70], ● [84](stream function) ● [103], ○ [106], ○ [131] ○ [162], ● [163] ● [164](AE), ● [164](split) ○ [167](reconstruction), ○ [168]	● [46](magnitude), ○ [47](magnitude) ○ [50](adv), ○ [50](content), ○ [50](feature) ○ [51](feature), ○ [51](volumetric), ○ [53], ● [54] ○ [55](adv), ○ [55](cycle), ○ [55](volumetric) ● [56](adv), ● [56](volumetric) ○ [109](reconstruction), ● [117] ○ [126](vector), ○ [143], ● [163], ○ [182]	○ [51](adv) ○ [54](adv) ● [70](adv) ○ [106](adv) ○ [168](adv)	● [27](adv) ○ [162](adv) ○ [167](adv)	CD^2 ○ [2](angle) CD^2 ● [46](angle) CD^2 ○ [47](angle)
image	● [12], ○ [160]	● [158]	● [12](adv) ● [63](adv)	—	SSIM ● [158]
feature	○ [2](Jacobian) ● [27](modified) ● [27](regularized), ● [57] ● [84](stream function) ● [84](velocity gradient) ● [144], ● [159](AO) ● [159](depth), ● [159](mask) ● [159](normal) ● [161], ● [164](AE)	● [12](AE), ● [26](hinge), ● [35] ○ [45](streamline), ● [46](Jacobian) ○ [49](streamline), ● [52] ● [63](reconstruction), ○ [76], ● [84], ● [85] ● [97](attribute), ○ [109](scalar gradient), ● [120] ○ [126](streamline), ○ [156](regularization) ○ [156](translation), ● [159](color-temporal) ● [161](bounds), ● [161](prior) ● [164](AE), ● [164](supervised), ○ [168]	● [161]	—	SSIM ● [35](AO) contrastive ● [64](local) MI ● [64](global) CD^1 ○ [156](deformation)
probability	—	● [48], ● [116], ● [169]	● [11], ● [15] ● [31], ● [40] ● [48], ● [81] ● [116] ● [144]	● [62]	CE ● [24], Dice ● [48] CE ● [65] log-likehood ● [71] AWL ● [116] CE ● [130](GSAL) Dice ● [154](MIP) Dice ● [154](voxel) CE ● [183]
parameter	—	● [62](gradient penalty), ● [144] ○ [162](gradient penalty)	—	—	—

distance for probability distributions over metric spaces. As an alternative to traditional GAN training, WGAN [3] can improve the stability of the network's optimization process.

Across the primary rows in Table 11, we can see that data- and feature-level optimization targets are the most widely used ones, followed by probability-level targets. Finally, image- and parameter-level optimization targets are the least employed ones. Across the columns, we see that L2 (MSE) and L1 (MAE) losses are the most popular ones. Between BCE and Wasserstein, BCE is more frequently used due to its simplicity and easy implementation. In the column of "others", different loss functions other than L1, L2, BCE, and Wasserstein include CE, Dice, CD^2, SSIM, etc.

Note that more than half of the surveyed papers employ more than one loss term. The goal is to consider different aspects to improve the overall inference quality. In addition, many papers coin specific loss names (e.g., adversarial, content, cycle, feature, reconstruction, temporal, and volumetric losses) reflecting their respective contexts. But in essence, the underlying loss function is mostly L1 or L2 loss.

Along the research task dimension, data generation tasks [○ ○ ● ●] employ losses across all levels except for probability-level targets. Visualization generation tasks [●] use losses across data-, image-, and feature-level targets. Prediction tasks [○ ●] and feature learning and extraction tasks [●] utilize losses across all levels except for image-level targets. Finally, object detection and segmentation tasks [●] only use the probability-level optimization target.

3.6 Evaluation Metrics

All papers we survey include qualitative results that show the visualizations generated from their solutions. In most cases, they also compare their works' results with other methods (including DL- and non-DL-based). Many of them also report the timing (including training and inference) performance of their neural networks. Besides qualitative results, many papers utilize quantitative metrics in their evaluations (we only find six exceptions [52], [62], [103], [120], [162], [168]). In the following, we discuss quantitative metrics these surveyed papers employ in the evaluation.

We categorize the evaluation metrics into six levels: *data*, *image*, *feature*, *probability*, *physics*, and *human*. Data-level metrics quantify the errors produced from synthesized or reconstructed data (e.g., raw scalars or vectors) compared with the ground-truth data. Image-level metrics compute the differences between visualization images (e.g., volume rendering, isosurface rendering, streamline visualization, pathline visualization) produced from synthesized and ground-truth data or produced from neural networks and traditional rendering processes. Feature-level metrics evaluate the gaps between visual representations (e.g., streamlines, pathlines, isosurfaces, stream surfaces) produced from inferred and original data. Probability-level metrics compare the differences between predicted probabilities (e.g., boundary maps, segmentation maps) and ground-truth ones. Physics-level metrics calculate the deviations of physics-related quantities (e.g., power spectra, kinetic energy) derived from synthesized and ground-truth data. Finally, human-level metrics ask human subjects to give ratings or scores to the results (typically visualization results) produced from synthesized data with ground-truth references or compare results (e.g., viewpoints) suggested by neural networks with those selected by humans.

As shown in Table 12, data-level metrics are most popular, followed by feature-, probability-, and image-level

TABLE 12: Categorization of the surveyed papers into the six levels of evaluation metrics. Metrics in each category are arranged based on popularity, followed by alphabetical order. Color coding is for data generation [○ ● ● ●], visualization generation [●], prediction [● ●], object detection and segmentation [●], and feature learning and extraction [●].

level	metric: papers
data	PSNR: ○ [2], ● [45], ● [46], ○ [47], ● [48], ● [49], ○ [50] ○ [51], ○ [53], ● [54], ○ [55], ● [56], ● [109], ● [117] ○ [126], ● [131], ● [163], ● [164], ○ [167], ○ [182] CR: ○ [2], ● [46], ● [49], ○ [53], ● [84], ● [106], ● [109] ● [160], ○ [167] AAD: ● [45], ○ [47], ● [48], ○ [53] RMSE: ○ [47], ● [106], ● [117] SSIM: ● [144], ○ [167], ○ [182] MAE: ● [27], ● [84] MSE: ● [143], ● [144] RAE: ● [46], ○ [53] CD^2: ● [64] EMD: ● [144] LSiM: ● [27] VGG metric: ● [144]
image	SSIM: ● [40], ○ [50], ○ [51], ○ [55], ● [56], ● [63], ● [131] ● [158], ● [159], ● [160], ● [161] LPIPS: ○ [53], ● [158], ● [160], ● [161] PSNR: ● [63], ● [70], ● [159], ● [161] EMD: ● [12], ● [63], ● [131] FID: ● [63], ● [158] MAE: ● [40] REC: ● [159] RMSE: ● [12]
feature	IS: ○ [50], ○ [51], ● [54], ○ [55], ● [56] MSE: ● [11], ● [24], ● [35], ○ [76], ● [183] MCPD: ○ [2], ● [45], ● [46], ○ [53] HD: ● [156], ● [163] IOU: ● [131], ● [156] accuracy: ● [40] ACPPE: ○ [2] AEDR: ● [57] ALP: ○ [126] association score: ● [64] CD^1: ● [156] EMD: ● [156] F-score: ● [156] feature deviation: ● [97] finger count: ● [97] global error: ● [57] local error: ● [57] ME: ● [24] PSNR: ○ [76] SC: ● [64] SSIM: ● [35]
probability	F-score: ● [11], ● [40], ● [48], ● [65], ● [81], ● [116], ● [154] PPV: ● [31], ● [40], ● [65], ● [81], ● [154] TPR: ● [31], ● [40], ● [65] IOU: ● [15], ● [40] Jaccard: ● [24], ● [81] classification accuracy: ● [130] classification error: ● [130] F_β: ● [40] FN: ● [11] FP: ● [11] FPR: ● [154] HR: ● [71] ROR: ● [26] TN: ● [11] TP: ● [11]
physics	kinetic energy: ● [27] power spectra: ○ [167] RMSE of vorticity: ○ [47] RMSE of wall shear stress: ○ [47] TPD: ● [85] vorticity ratio: ● [27]
human	MOS: ● [54], ○ [50] AER: ● [169] average top-3 match distance: ● [169] # hits: ● [169]

metrics. Physics- and human-level metrics are the least used ones. Across the research tasks, data generation tasks [○ ● ● ●] use metrics across all but probability-level metrics. However, visualization generation tasks [●] exclusively utilize image-level metrics. Data-relevant prediction tasks [●] use data-, feature-, probability, and physics-level metrics, while visualization-relevant prediction tasks [●] utilize feature-, probability-, and human-level metrics. Most object detection and segmentation tasks [●] employ probability-level metrics (with a few exceptions using image- and feature-level metrics). Finally, feature learning and extraction tasks [●] utilize data-, feature-, and probability-level metrics but no image- and human-level metrics.

PSNR, SSIM, IS/MSE, and F-score are the most widely used ones in the data-, image-, feature-, and probability-level metrics, respectively. In addition, several metrics (i.e., PSNR, SSIM, MSE, RMSE, MAE, EMD, F-score, IOU) are utilized across different categories. A closer look shows that many papers employ more than one evaluation metric, and in this case, several of them (e.g., [2], [27], [47], [51]) utilize metrics across categories for a comprehensive evaluation.

Several data- and feature-level metrics are primarily used for vector fields, flow lines, or critical points. AAD and CD^2 are data-level metrics for comparing individual vectors' angles and magnitudes. At the feature level, ALP and MCPD compare integral flow lines' endpoints or calculate the distances among sample points along flow lines. ACPPE considers critical point position deviations. In addition, three metrics (i.e., LSiM, VGG metric, LPIPS) are learned metrics based on neural networks.

Most data-level metrics evaluate the data error at individual voxel or pixel (i.e., 2D data slice) levels. Other than that, CR is often used to evaluate data compression or reduction performance. LSiM [89] was utilized to evaluate the accuracy of static and temporal restoration in fluid simulation [27]. VGG metric [135] was compared against self-supervised learning of spatiotemporal similarity [144].

For image-level metrics, PSNR, MAE, and RMSE examine the differences of two images at the pixel level. SSIM compares two images according to the patch-level means, variances, and covariances. As a network-based similarity metric, LPIPS [175] computes a weighted average of the activations at hidden layers to predict relative image similarities correlating well with perceptual judgments. EMD [124] and FID [66] quantify the distances between two images at the histogram and distribution levels.

Two feature-level metrics were used to evaluate surface similarities: IS [18] (which employs MI to identify the similarity between isosurfaces) and surface-based HD [163] (which uses HD to compute the error between the predicted and reference liquid surfaces represented in signed distance functions).

Several probability-level metrics are related to the confusion matrix in a supervised learning setting (e.g., classification or segmentation task). The basic terms of the confusion matrix are FN, FP, TN, and TP, and their derivations include accuracy, F_β, FPR, F-score, IOU, Jaccard, PPV, and TPR.

Finally, we notice that physics- and human-level metrics are far less employed in the evaluation. This is due to the lack of physics-informed DL works in SciVis and the missing of a rich set of human-level quantitative metrics.

4 Research Opportunities

DL4SciVis is a fast-growing area in SciVis, which may play a pivotal role in the future of SciVis. Reflecting on several dimensions (i.e., domain setting, research task, learning type, network architecture) used to classify the surveyed papers (refer to Tables 1, 9, and 10), we can identify gaps that suggest possible future research directions. This section examines the remaining gaps based on the surveyed papers and points out several research opportunities for DL4SciVis.

From single field to multiple fields. Table 1 shows that more than half of the surveyed papers are for scalar field data, which is about twice as many as those for vector field data. Although many papers deal with time-varying scalar fields (e.g., [50], [51], [55], [143], [144]) or unsteady vector fields (e.g., [2], [45], [71], [76], [85]), only a few works handle multivariate data [56], [117] or ensemble data [62], [63]. We expect the future growth of DL4SciVis by considering multi-field (scalar, vector, tensor) and multi-run (ensemble) data and the interplay among them. For example, Chu et al. [27] considered the data translation problem (from density scalar field to velocity vector field). In CV, DL-based image-to-image translation (e.g., Pix2Pix [75], CycleGAN [184]) and image colorization [174] works provide us good examples to design suitable solutions for SciVis data.

From data generation to visualization generation. In Table 1, we can see a starking contrast between the numbers

of papers on data generation and visualization generation (even including visualization-relevant prediction works). Data generation works have grown significantly in the subcategories of super-resolution generation and compression and reconstruction. Visualization generation, however, presents more challenges as we need to consider different parameters involved in the rendering process, including transfer function, viewpoint, lighting, etc. Therefore, the lagging of visualization generation behind data generation is reasonable. Nevertheless, the new surge of differentiable rendering [82] or neural rendering [139] in CG could be poised to become a new area in SciVis.

From images and volumes to graphs. We can observe from Tables 4 to 8 that most papers focus on processing SciVis data in their original forms (e.g., images and volumes) while derived forms (e.g., graphs) are seldom operated. So far, only four works use GNN [52], [64], [65], [131], where GCNs are employed to learn scalar value association, super-voxel features, surface node features, and feature map upsampling, respectively. GNN is not limited to GCN [176], and it also includes GAE, GRN, and STGNN [166]. SciVis data have rich graph-like representations or relationships [148], such as surfaces (e.g., isosurfaces, stream surfaces) and relationships (e.g., correlation, transition, topology). Advances in GNN techniques from CG (e.g., geometric DL [17] on non-Euclidean domains such as graphs and manifolds) and knowledge discovery and data mining (KDD) [166], [179], [181] will provide ample opportunities for us to develop GNN-based solutions for solving SciVis problems.

From supervised learning to self-supervised learning. In Table 9, we can see that the majority of the surveyed papers fall into the category of supervised learning, where a large amount of annotated data is needed for training. However, this requirement is not easy to meet when the annotation is time-consuming and difficult to collect. In CV, self-supervised learning was proposed to address this issue. Self-supervised learning designs a pretext task to learn hidden representations with a large amount of unlabeled data. It then fine-tunes a downstream task (e.g., classification and segmentation) with few annotated data. Although there are several self-supervised learning works (i.e., [26], [120], [143]) in SciVis, they all utilize the Siamese network [16] for pairwise contrastive learning. Furthermore, these works focus on feature extraction without extending the framework to fine-tune the downstream tasks. In addition, the relationship between pretext and downstream tasks is still unclear. We expect more works in this category, especially leveraging new frameworks (such as CMC [142], SimCLR [23], MoCo [60], and BYOL [44]) and novel pretext tasks. Besides contrastive learning, the unexplored generative learning [105] presents a unique opportunity.

From CNN and RNN to MLP. We can see from Table 10 that CNNs and RNNs have been well utilized in DL4SciVis research. They have also been extensively used in generative tasks, such as super-resolution generation and visualization generation. To our surprise, pure MLP was seldom employed in these tasks, and the only MLP work (i.e., [109]) is for data compression. Recent investigations in CV have demonstrated that a pure MLP model can outperform CNN- and RNN-based architectures across a diverse set of tasks, such as image classification [32], [107], image segmentation [180], object detection [19], image reconstruction [136], and view synthesis [21]. We expect more DL4SciVis works utilizing MLPs and newer mechanisms or architectures, such as transformer [147] and implicit neural representation [113].

From distributed learning to disentangled learning and style transfer. We observe that most unsupervised learning works in SciVis are distributed learning, e.g., an encoder+decoder network architecture is designed to extract unified features through reconstruction. However, the possibility of disentangled learning (refer to Figure 1) and style transfer is still unexplored. In CV, Gatys et al. [39] pioneered the use of CNNs to extract feature responses of a photo as the *content* and the feature statistics of a piece of artwork as the *style* for neural style transfer (i.e., rendering a content image in different styles). Huang et al. [74] built two encoders to learn the content (e.g., shape) and style (e.g., texture and pose) of an image, respectively. After that, fixing the content feature while switching different style features can render arbitrary images with the same content in different styles. Many styles can be considered, including semantic, instance, doodle, stereoscopic, portrait, video, character, photorealistic, attribute, fashion, and audio styles [78]. Although it is not as intuitive as images or videos to define content and style for SciVis data, it is not impossible. For example, given an ensemble fluid flow simulation, the content could be invariant information (e.g., vortices). The style could be the pattern (e.g., the number, size, and location of vortices) extracted by simulation output under varying Reynolds numbers, which indicate how turbulent the flow is. If such a definition is meaningful and the transferred results are interpretable, disentangled learning for data generation can play a role in DL4SciVis.

From heavyweight to lightweight. Currently, the DL models in DL4SciVis are built with tens or hundreds of layers to guarantee quality. However, this could result in a large model size and inefficient inference. In the ML community, researchers have already studied different techniques (e.g., weight quantization [58], [59], [101] and knowledge distillation [37], [67], [98], [185]) to build a lightweight model from a heavyweight one. For instance, Han et al. [59] pruned the network, quantized parameters and compressed them using Huffman coding. Li et al. [98] applied neural architecture search to find efficient architectures through combining the knowledge of multiple intermediate features extracted from the heavyweight model. Thomas et al. [140] presented QW-Net for image reconstruction, where about 95% of the computations can be implemented with 4-bit integers. We believe there is an opportunity to incorporate these techniques into DL models to improve training efficiency for large-scale scientific data analysis and visualization.

From centralized learning to federated learning. The success of DL models heavily relies on a large amount of data. However, due to practical issues such as confidentiality and privacy, SciVis data are often not publicly available. This prevents the DL models from gaining a strong learning capability from different sources. Instead of requiring data sets, researchers can release the trained models. Federated learning [79] can produce a shared model by collaborating with local models trained on different data

sets. The shared model does not need to access the trained data sets. Techniques in federated learning include weight averaging [112], momentum update [72], Bayesian nonparametric match [149], and model contrast [99]. Although these algorithms were designed for classification tasks, we expect researchers to explore this direction in SciVis-related tasks by designing new approaches when multiple local models are available.

From data-driven DL to physics-informed DL. Existing works in DL4SciVis are primarily data-driven, require a significant amount of data for network training. These works are often purely data-driven, seldom leveraging the underlying physics for physics-informed DL. Outside the SciVis community, researchers in CFD and fluid simulation have extensively investigated physics-informed DL [80], [141]. For instance, Raissi et al. [119] introduced a physics-informed neural network for solving supervised learning tasks involving nonlinear partial differential equations. Considering the data and physics scenarios, we have one extreme of big data with no physics and the other extreme of small data with lots of physics. Physics-informed DL applies to the middle regime with some data and some physics. Using differentiable physics and neural networks, physics-informed DL can integrate data and the governing physical laws to produce predictions conforming to the underlying physics, even for models with partially missing physics. As listed in Table 1, several surveyed papers in the fluid simulation domain (i.e., [27], [84], [162], [163], [164], [168]) have considered physical information or quantities in their works, sharing some flavor of physics-informed DL. Although there are practical gaps to be bridged, SciVis researchers may find it rewarding to work closely with physics and simulation researchers to jointly advance physics-informed DL.

5 Open Challenges

In a recent article published in *Communications of the ACM*, Bengio et al. [10] outlined some of the future challenges facing DL for AI, including training with little or no supervision, robustness to test samples out of training data distribution, and DL for tasks requiring a deliberate sequence of steps. Compared to the advances of DL in CV and CG, DL4SciVis is still in its early stage of development. Despite significant progress over the past few years, many breakthroughs still need to be made in DL4SciVis research. In this section, we identify and discuss several open yet pressing challenges facing us.

Model generalization. A universal neural network model trained on various images could effectively perform intended tasks on multiple image categories. However, this is not the case for SciVis data due to the lack of sufficiently large and diverse enough training data and the added training cost (2D images vs. 3D volumes). Current DL4SciVis works allow training a model on specific variables or ensemble runs and later applying the model to a different variable sequence [56] or ensemble run [63] of the same simulation. Beyond that, the performance often downgrades [48]. SciVis researchers have not seriously investigated the issue of generalizing the neural network model to adapt appropriately to new, previously unseen data. The current practice of "one training for one dataset" ought to be changed. Training a generalized model to work well from one dataset to different datasets poses an open challenge. Commonly used in CV, data augmentation techniques [134] can help improve model generalization. Their goal is to increase the amount of training data by adding modified copies of existing data (e.g., via cropping, transformation, noise injection, and random erasing) or newly created synthetic data from existing data using generative solutions [95], [177]. Besides training with large and diverse data, it is also possible to improve model generalization by introducing pre-training stages [36] or building connections between early and latter layers (e.g., residual connection [61] and dense connection [73]).

Benchmark dataset. CV and CG researchers have produced many benchmark datasets for reproducible research. The hugely successful benchmark databases such as ImageNet [30] demonstrate their value of benefiting the community and advancing the field. In SciVis, such an effort is scarce. SciVis researchers are often short of experimental data. Unlike CV and CG, where images, videos, and geometric models are readily available for use, SciVis researchers only obtain datasets from domain scientists they collaborate with or from publicly available sources (e.g., IEEE SciVis Contest), which are rather limited. Recently, Eckert et al. [33] produced a large-scale volumetric dataset of scalar transport flows for computer animation and ML. Jakob et al. [76] released a large numerical 2D fluid dataset, including laminar and turbulent flows, for ML. These encouraging initiatives will inspire more significant community efforts to propel DL4SciVis and SciVis research in general.

Few-shot learning. DL tasks are data-hungry. For example, generating super-resolution for time-varying volumetric data needs 40% of samples from the datasets for training [50], [51] and training a generative model for volume rendering requires 200,000 image samples [12]. In CV, Li et al. [96] introduced one-shot learning, a Bayesian approach for learning object categories, where much information about a new object category can be learned from a single or just a few training examples. Leverages prior knowledge, few-shot learning can generalize to new tasks involving only a small number of samples with supervised information [155]. This direction is especially appealing to DL4SciVis research due to the limited data and time-consuming training. However, it remains challenging to determine what prior knowledge can be obtained and how such knowledge should be utilized in few-shot learning.

Multi-task learning. All surveyed papers we survey only tackle a single task. Multi-task learning [20] aims to learn multiple related tasks *simultaneously* by sharing the knowledge obtained from different tasks to improve the generalization performance of all these tasks. Zhang and Yang [178] classified different multi-task learning algorithms into the following categories: feature learning, low-rank, task clustering, task relation learning, and decomposition. Existing DL works on multi-tasking learning are often based on sharing hidden layers of the neural network, which is vulnerable to noisy and outlier tasks. For DL4SciVis, open questions include defining and group tasks, exploiting unrelated tasks, and applying multi-task learning in non-supervised learning scenarios.

Interpretable DL. All the surveyed works treat the DL models as black boxes. This makes it difficult to interpret

or modify the model results when the predictions are inaccurate or do not meet particular constraints (e.g., the cell size in medical images and physical properties in simulation data). Interpretable DL [173] aims to study the role of each neuron in a DL model and understand the decision process. Recent works [6], [7], [8], [9] investigated the importance of each neuron in image classification and generation tasks (e.g., identifying the neurons that can control the generation or classification of church). With this interpretation, researchers can manipulate the model behavior to generate desired results. For example, a GAN model can produce images without a sofa by eliminating the sofa neurons. For DL4SciVis, open questions include discovering the neurons with different roles, controlling specific neurons, and rewriting the neurons to produce customized results.

Automated ML. Most DL4SciVis works entail great efforts on researchers. They need to be involved in every stage of the process, including problem definition, data collection, feature engineering, model selection, algorithm selection, evaluation, and deployment. Automated ML [170] can relieve us from intermediate steps (i.e., feature engineering, model selection, algorithm selection, and evaluation), minimizing human participation and improving efficiency. Of particular interest for SciVis researchers is neural architecture search [102], [186]. Designing network architectures has been the primary task for achieving good learning performance, often demanding a time-consuming and painstaking process. For example, a typical CNN design space includes many choices, such as the number of filters, filter width and height, stride width and height, and skip connections. The iterative nature of the architecture generation process makes reinforcement learning [137] a suitable choice for neural architecture search. Open questions for DL4SciVis researchers include determining the search space, the corresponding feedback, and the number of configurations for evaluation.

6 Conclusions

We have presented a state-of-the-art survey on DL4SciVis. The survey covers 59 papers published since 2017 along six dimensions, provides an in-depth discussion on their similarities and differences, identifies trends and gaps, and outlines research opportunities and open challenges. Despite the fantastic advances of DL4SciVis, we acknowledge that researchers need to address many practical issues to make the presented solutions robust to outliers, generalizable across datasets, and applicable in real-world settings. As DL4SciVis has grown out of its infancy stage, we anticipate future research will answer these challenges.

DL4SciVis is only a branch of AI4VIS. With AI4VIS and VIS4AI, the entire area of AI+VIS has quickly become the most vibrant research focus in VIS. The astonishing advancement of ML and DL, the interplay between AI4VIS and VIS4AI, and the interconnection across SciVis, InfoVis, and VA provide a myriad of thoughts and ideas for sustainable growth of AI+VIS research for years to come. We hope this survey can serve as a good source of reference for SciVis researchers and shed light on future directions.

Acknowledgements

This work was supported in part by the U.S. National Science Foundation through grants IIS-1455886, CNS-1629914, DUE-1833129, IIS-1955395, IIS-2101696, and OAC-2104158. The authors would like to thank the anonymous reviewers for their insightful comments.

References

[1] G. Alicioglu and B. Sun. A survey of visual analytics for explainable artificial intelligence methods. *Computers & Graphics*, 102:502–520, 2022.

[2] Y. An, H.-W. Shen, G. Shan, G. Li, and J. Liu. STSRNet: Deep joint space-time super-resolution for vector field visualization. *IEEE Computer Graphics and Applications*, 41(6):122–132, 2021.

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of International Conference on Machine Learning*, pages 214–223, 2017.

[4] D. H. Ballard. Modular learning in neural networks. In *Proceedings of AAAI Conference on Artificial Intelligence*, pages 279–284, 1987.

[5] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.

[6] D. Bau, S. Liu, T. Wang, J.-Y. Zhu, and A. Torralba. Rewriting a deep generative model. In *Proceedings of European Conference on Computer Vision*, pages 351–369, 2020.

[7] D. Bau, H. Strobelt, W. Peebles, J. Wulff, B. Zhou, et al. Semantic photo manipulation with a generative image prior. *ACM Transactions on Graphics*, 38(4):59:1–59:11, 2019.

[8] D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, et al. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, 2020.

[9] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, et al. GAN dissection: Visualizing and understanding generative adversarial networks. In *Proceedings of International Conference on Learning Representations*, 2019.

[10] Y. Bengio, Y. Lecun, and G. E. Hinton. Deep learning for AI. *Communications of the ACM*, 64(7):58–65, 2021.

[11] M. Berenjkoub, G. Chen, and T. Günther. Vortex boundary identification using convolutional neural network. In *Proceedings of IEEE Visualization Conference (Short Papers)*, pages 261–265, 2020.

[12] M. Berger, J. Li, and J. A. Levine. A generative model for volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 25(4):1636–1650, 2019.

[13] D. Berthelot, T. Schumm, and L. Metz. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[14] J. Bi and K. P. Bennett. Regression error characteristic curves. In *Proceedings of International Conference on Machine Learning*, pages 43–50, 2003.

[15] K. Borkiewicz, V. Shah, J. P. Naiman, C. Shen, S. Levy, et al. CloudFindr: A deep learning cloud artifact masker for satellite DEM data. In *Proceedings of IEEE Visualization Conference (Short Papers)*, pages 1–5, 2021.

[16] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a Siamese time delay neural network. In *Proceedings of Advances in Neural Information Processing Systems*, pages 737–744, 1993.

[17] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[18] S. Bruckner and T. Möller. Isosurface similarity maps. *Computer Graphics Forum*, 29(3):773–782, 2010.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, et al. End-to-end object detection with transformers. In *Proceedings of European Conference on Computer Vision*, pages 213–229, 2020.

[20] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[21] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 5799–5809, 2021.

[22] A. Chatzimparmpas, R. M. Martins, I. Jusufi, K. Kucher, F. Rossi, et al. The state of the art in enhancing trust in machine learning models with the use of visualizations. *Computer Graphics Forum*, 39(3):713–756, 2020.

[23] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of International Conference on Machine Learning*, pages 1597–1607, 2020.

[24] H.-C. Cheng, A. Cardone, S. Jain, E. Krokos, K. Narayan, et al. Deep-learning-assisted volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 25(2):1378–1391, 2019.

[25] Z. Cheng, Q. Yang, and B. Sheng. Deep colorization. In *Proceedings of IEEE International Conference on Computer Vision*, pages 415–423, 2015.

[26] M. Chu and N. Thuerey. Data-driven synthesis of smoke flows with CNN-based feature descriptors. *ACM Transactions on Graphics*, 36(4):69:1–69:14, 2017.

[27] M. Chu, N. Thuerey, H.-P. Seidel, C. Theobalt, and R. Zayer. Learning meaningful controls for fluids. *ACM Transactions on Graphics*, 40(4):100:1–100:13, 2021.

[28] D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, 2010.

[29] I. Corouge, S. Gouttard, and G. Gerig. Towards a shape model of white matter fiber bundles using diffusion tensor MRI. In *Proceedings of International Symposium on Biomedical Imaging*, pages 344–347, 2004.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, et al. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[31] L. Deng, Y. Wang, Y. Liu, F. Wang, S. Li, et al. A CNN-based vortex identification method. *Journal of Visualization*, 22(1):65–78, 2019.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, et al. An image is worth 16×16 words: Transformers for image recognition at scale. In *Proceedings of International Conference on Learning Representations*, 2020.

[33] M.-L. Eckert, K. Um, and N. Thuerey. ScalarFlow: A large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Transactions on Graphics*, 38(6):239:1–239:16, 2019.

[34] A. Endert, W. Ribarsky, C. Turkay, B. W. Wong, I. Nabney, et al. The state of the art in integrating machine learning into visual analytics. *Computer Graphics Forum*, 36(8):458–486, 2017.

[35] D. Engel and T. Ropinski. Deep volumetric ambient occlusion. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1268–1278, 2021.

[36] D. Erhan, Y. Bengio, A. C. Courville, P.-A. Manzagol, P. Vincent, et al. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010.

[37] Z. Fang, J. Wang, X. Hu, L. Wang, Y. Yang, et al. Compressing visual-linguistic model via knowledge distillation. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1428–1438, 2021.

[38] C. Frogner, C. Zhang, H. Mobahi, M. Araya-Polo, and T. A. Poggio. Learning with a Wasserstein loss. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2053–2061, 2015.

[39] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[40] P. Ghahremani, S. Boorboor, P. Mirhosseini, C. Gudisagar, M. Ananth, et al. NeuroConstruct: 3D reconstruction and visualization of neurites in optical microscopy brain images. *IEEE Transactions on Visualization and Computer Graphics*, 2021. Accepted.

[41] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. https://www.deeplearningbook.org/contents/rnn.html.

[42] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, et al. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[43] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings of IEEE International Joint Conference on Neural Networks*, pages 729–734, 2005.

[44] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, et al. Bootstrap your own latent - a new approach to self-supervised learning. In *Proceedings of Advances in Neural Information Processing Systems*, pages 21271–21284, 2020.

[45] P. Gu, J. Han, D. Z. Chen, and C. Wang. Reconstructing unsteady flow data from representative streamlines via diffusion and deep learning based denoising. *IEEE Computer Graphics and Applications*, 41(6):111–121, 2021.

[46] P. Gu, J. Han, D. Z. Chen, and C. Wang. Scalar2Vec: Translating scalar fields to vector fields via deep learning. In *Proceedings of IEEE Pacific Visualization Symposium*, 2022. Accepted.

[47] L. Guo, S. Ye, J. Han, H. Zheng, H. Gao, et al. SSR-VFD: Spatial super-resolution for vector field data analysis and visualization. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 71–80, 2020.

[48] J. Han, J. Tao, and C. Wang. FlowNet: A deep learning framework for clustering and selection of streamlines and stream surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 26(4):1732–1744, 2020.

[49] J. Han, J. Tao, H. Zheng, H. Guo, D. Z. Chen, et al. Flow field reduction via reconstructing vector data from 3D streamlines using deep learning. *IEEE Computer Graphics and Applications*, 39(4):54–67, 2019.

[50] J. Han and C. Wang. SSR-TVD: Spatial super-resolution for time-varying data analysis and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2020. Accepted.

[51] J. Han and C. Wang. TSR-TVD: Temporal super-resolution for time-varying data analysis and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):205–215, 2020.

[52] J. Han and C. Wang. SurfNet: Learning surface representations via graph convolutional network. *Computer Graphics Forum*, 41(3), 2022. In Press.

[53] J. Han and C. Wang. TSR-VFD: Generating temporal super-resolution for unsteady vector field data. *Computers & Graphics*, 103:168–179, 2022.

[54] J. Han and C. Wang. VCNet: A generative model for volume completion. *Visual Informatics*, 2022. Accepted.

[55] J. Han, H. Zheng, D. Z. Chen, and C. Wang. STNet: An end-to-end generative framework for synthesizing spatiotemporal super-resolution volumes. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):270–280, 2022.

[56] J. Han, H. Zheng, Y. Xing, D. Z. Chen, and C. Wang. V2V: A deep learning approach to variable-to-variable selection and translation for multivariate time-varying data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1290–1300, 2021.

[57] M. Han, S. Sane, and C. R. Johnson. Exploratory Lagrangian-based particle tracing using deep learning. *arXiv preprint arXiv:2110.08338*, 2021.

[58] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, et al. Deep compression and EIE: Efficient inference engine on compressed deep neural network. In *Proceedings of IEEE Hot Chips Symposium*, pages 1–6, 2016.

[59] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding. In *Proceedings of International Conference on Learning Representations*, 2016.

[60] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[61] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[62] W. He, J. Wang, H. Guo, H.-W. Shen, and T. Peterka. CECAV-DNN: Collective ensemble comparison and visualization using deep neural networks. *Visual Informatics*, 4(2):109–121, 2020.

[63] W. He, J. Wang, H. Guo, K.-C. Wang, H.-W. Shen, et al. InSituNet: Deep image synthesis for parameter space exploration of ensemble simulations. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):23–33, 2020.

[64] X. He, Y. Tao, S. Yang, C. Chen, and H. Lin. ScalarGCN: Scalar-value association analysis of volumes based on graph convolutional network. *Journal of Visualization*, 25(1):77–93, 2022.

[65] X. He, S. Yang, Y. Tao, H. Dai, and H. Lin. Graph convolutional network-based semi-supervised feature classification of volumes. *Journal of Visualization*, 25(2):379–393, 2022.

[66] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Proceedings of Advances in Neural Information Processing Systems*, pages 6627–6638, 2017.

[67] G. E. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[68] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[69] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2674–2693, 2019.

[70] F. Hong, C. Liu, and X. Yuan. DNN-VolVis: Interactive volume visualization supported by deep neural network. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 282–291, 2019.

[71] F. Hong, J. Zhang, and X. Yuan. Access pattern learning with long short-term memory for parallel particle tracing. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 76–85, 2018.

[72] T.-M. H. Hsu, H. Qi, and M. Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[73] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017.

[74] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of European Conference on Computer Vision*, pages 172–189, 2018.

[75] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[76] J. Jakob, M. Gross, and T. Günther. A fluid flow data set for machine learning and its application to neural flow map interpolation. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1279–1289, 2021.

[77] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4037–4058, 2021.

[78] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, et al. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3365–3385, 2020.

[79] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1-2):1–210, 2021.

[80] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, et al. Physics-informed machine learning. *Nature Reviews Physics*, 3:422–440, 2021.

[81] B. Kashir, M. Ragone, A. Ramasubramanian, V. Yurkiv, and F. Mashayek. Application of fully convolutional neural networks for feature extraction in fluid flow. *Journal of Visualization*, 24(4):771–785, 2021.

[82] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, et al. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020.

[83] M. Kawato, Y. Maeda, Y. Uno, and R. Suzuki. Trajectory formation of arm movement by cascade neural network model based on minimum torque-change criterion. *Biological Cybernetics*, 62:275–288, 1990.

[84] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, et al. Deep Fluids: A generative network for parameterized fluid simulations. *Computer Graphics Forum*, 38(2):59–70, 2019.

[85] B. Kim and T. Günther. Robust reference frame extraction from unsteady 2D vector fields with convolutional neural networks. *Computer Graphics Forum*, 38(3):285–295, 2019.

[86] I. Kimura, A. Yoke, A. Kaga, and Y. Kuroe. Neural-net based modeling of velocity and concentration fields. *Journal of Visualization*, 12(1):73–80, 2009.

[87] T. N. Kipf and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[88] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of International Conference on Learning Representations*, 2017.

[89] G. Kohl, K. Um, and N. Thuerey. Learning similarity metrics for numerical simulations. In *Proceedings of International Conference on Machine Learning*, pages 5349–5360, 2020.

[90] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1106–1114, 2012.

[91] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[92] F. Lan, M. Young, L. Anderson, A. Ynnerman, A. Bock, et al. Visualization in astrophysics: Developing new methods, discovering our universe, and educating the earth. *Computer Graphics Forum*, 40(3):635–663, 2021.

[93] S. Lan, R. Yu, G. Yu, and L. S. Davis. Modeling local geometric structure of 3D point clouds using Geo-CNN. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 998–1008, 2019.

[94] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[95] D. Li, J. Yang, K. Kreis, A. Torralba, and S. Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 8300–8311, 2021.

[96] F.-F. Li, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[97] H. Li and H.-W. Shen. Local latent representation based on geometric convolution for particle data feature exploration. *IEEE Transactions on Visualization and Computer Graphics*, 2022. Accepted.

[98] M. Li, J. Lin, Y. Ding, Z. Liu, J.-Y. Zhu, et al. GAN compression: Efficient architectures for interactive conditional GANs. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 5284–5294, 2020.

[99] Q. Li, B. He, and D. Song. Model-contrastive federated learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021.

[100] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[101] J. Lin, C. Gan, and S. Han. Defensive quantization: When efficiency meets robustness. In *Proceedings of International Conference on Learning Representations*, 2019.

[102] C. Liu, B. Zoph, J. Shlens, W. Hua, L.-J. Li, et al. Progressive neural architecture search. In *Proceedings of European Conference on Computer Vision*, pages 19–35, 2018.

[103] Q. Liu, S. Hazarika, J. M. Patchett, J. P. Ahrens, and A. Biswas. Deep learning-based feature-aware data modeling for complex physics simulations. *arXiv preprint arXiv:1912.03587*, 2019.

[104] S. Liu, X. Wang, M. Liu, and J. Zhu. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1):48–56, 2017.

[105] X. Liu, F. Zhang, Z. Hou, Z. Wang, L. Mian, et al. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021. Accepted.

[106] Y. Liu, Y. Wang, L. Deng, F. Wang, F. Liu, et al. A novel in situ compression method for CFD data based on generative adversarial network. *Journal of Visualization*, 22(1):95–108, 2019.

[107] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of IEEE International Conference on Computer Vision*, pages 10012–10022, 2021.

[108] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[109] Y. Lu, K. Jiang, J. A. Levine, and M. Berger. Compressive neural representations of volumetric scalar fields. *Computer Graphics Forum*, 40(3):135–146, 2021.

[110] K.-L. Ma. Machine learning to boost the next generation of visualization technology. *IEEE Computer Graphics and Applications*, 27(5):6–9, 2007.

[111] A. Marzouk, P. Barros, M. Eppe, and S. Wermter. The conditional boundary equilibrium generative adversarial network and its application to facial attributes. In *Proceedings of International Joint Conference on Neural Networks*, pages 1–7, 2019.

[112] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[113] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, et al. NeRF: Representing scenes as neural radiance

fields for view synthesis. In *Proceedings of European Conference on Computer Vision*, pages 405–421, 2020.

[114] F. Milletari, N. Navab, and S.-A. Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *Proceedings of International Conference on 3D Vision*, pages 565–571, 2016.

[115] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[116] N. Nguyen, C. Bohak, D. Engel, P. Mindek, O. Strnad, et al. Finding nano-Ötzi: Semi-supervised volume visualization for cryo-electron tomography. *arXiv preprint arXiv:2104.01554*, 2021.

[117] W. P. Porter, Y. Xing, B. R. von Ohlen, J. Han, and C. Wang. A deep learning approach to selecting representative time steps for time-varying multivariate data. In *Proceedings of IEEE Visualization Conference (Short Papers)*, pages 131–135, 2019.

[118] R. Raina, A. Madhavan, and A. Y. Ng. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of International Conference on Machine Learning*, pages 873–880, 2009.

[119] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[120] M. Raji, A. Hota, R. Sisneros, P. Messmer, and J. Huang. Photo-guided exploration of volume data features. In *Proceedings of Eurographics Symposium on Parallel Graphics and Visualization*, pages 31–39, 2017.

[121] K. Ren, D. Qu, S. Xu, X. Jiao, L. Tai, et al. Uncertainty visualization of transport variance in a time-varying ensemble vector field. *ISPRS International Journal of Geo-Information*, 9(1):19:1–19:23, 2020.

[122] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[123] F. Rosenblatt. Principles of neurodynamics: Perceptrons and the theory of brain mechanisms. Technical Report 1196-G-8, Cornell Aeronautical Laboratory, Inc., Cornell University, 1961.

[124] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[125] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical Report ICS Report 8506, Institute for Cognitive Science, University of California, San Diego, 1985.

[126] S. Sahoo and M. Berger. Integration-aware vector field super resolution. In *Proceedings of Eurographics Visualization Conference (Short Papers)*, pages 49–53, 2021.

[127] M. S. M. Sajjadi, B. Schölkopf, and M. Hirsch. EnhanceNet: Single image super-resolution through automated texture synthesis. In *Proceedings of IEEE International Conference on Computer Vision*, pages 4501–4510, 2017.

[128] M. S. M. Sajjadi, R. Vemulapalli, and M. Brown. Frame-recurrent video super-resolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018.

[129] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[130] N. Shi and Y. Tao. CNNs based viewpoint estimation for volume visualization. *ACM Transactions on Intelligent Systems and Technology*, 10(3):27:1–27:22, 2019.

[131] N. Shi, J. Xu, S. W. Wurster, H. Guo, J. Woodring, et al. GNN-Surrogate: A hierarchical and adaptive graph neural network for parameter space exploration of unstructured-mesh ocean simulations. *IEEE Transactions on Visualization and Computer Graphics*, 28(6), 2022. In Press.

[132] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, et al. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[133] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Proceedings of Advances in Neural Information Processing Systems*, pages 802–810, 2015.

[134] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:60:1–60:48, 2019.

[135] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of International Conference on Learning Representations*, 2015.

[136] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. In *Proceedings of Advances in Neural Information Processing Systems*, 2020.

[137] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[138] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

[139] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, et al. State of the art on neural rendering. *Computer Graphics Forum*, 39(2):701–727, 2020.

[140] M. M. Thomas, K. Vaidyanathan, G. Liktor, and A. G. Forbes. A reduced-precision network for image reconstruction. *ACM Transactions on Graphics*, 39(6):231:1–231:12, 2020.

[141] N. Thuerey, P. Holl, M. Mueller, P. Schnell, F. Trost, et al. Physics-based deep learning. *arXiv preprint arXiv:2109.05237*, 2021.

[142] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. In *Proceedings of European Conference on Computer Vision*, pages 776–794, 2020.

[143] G. Tkachev, S. Frey, and T. Ertl. Local prediction models for spatiotemporal volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(7):3091–3108, 2021.

[144] G. Tkachev, S. Frey, and T. Ertl. S4: Self-supervised learning of spatiotemporal similarity. *IEEE Transactions on Visualization and Computer Graphics*, 2021. Accepted.

[145] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. A novel interface for higher-dimensional classification of volume data. In *Proceedings of IEEE Visualization Conference*, pages 505–512, 2003.

[146] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):273–284, 2005.

[147] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al. Attention is all you need. In *Proceedings of Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[148] C. Wang and J. Tao. Graphs in scientific visualization: A survey. *Computer Graphics Forum*, 36(1):263–287, 2017.

[149] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni. Federated learning with matched averaging. In *Proceedings of International Conference on Learning Representations*, 2020.

[150] Q. Wang, Z. Chen, Y. Wang, and H. Qu. A survey on ML4VIS: Applying machine learning advances to data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2021. Accepted.

[151] X. Wang, L. Bo, and F. Li. Adaptive wing loss for robust face alignment via heatmap regression. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 6971–6981, 2019.

[152] X. Wang, K. C. K. Chan, K. Yu, C. Dong, and C. C. Loy. EDVR: Video restoration with enhanced deformable convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1954–1963, 2019.

[153] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, et al. ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of European Conference on Computer Vision Workshop*, pages 63–79, 2018.

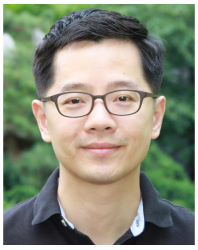
[154] Y. Wang, G. Yan, H. Zhu, S. Buch, Y. Wang, et al. VC-Net: Deep volume-composition networks for segmentation and visualization of highly sparse and noisy image data. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1301–1311, 2021.

[155] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):63:1–63:34, 2020.

[156] Y. Wang, Z. Zhong, and J. Hua. DeepOrganNet: On-the-fly reconstruction and visualization of 3D/4D lung models from single-view projections by deep deformation network. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):960–970, 2020.

[157] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[158] J. Weiss and N. Navab. Deep direct volume rendering: Learning visual feature mappings from exemplary images. *arXiv preprint arXiv:2106.05429*, 2021.
[159] S. Weiss, M. Chu, N. Thuerey, and R. Westermann. Volumetric isosurface rendering with deep learning-based super-resolution. *IEEE Transactions on Visualization and Computer Graphics*, 27(6):3064–3078, 2021.
[160] S. Weiss, P. Hermüller, and R. Westermann. Fast neural representations for direct volume rendering. *arXiv preprint arXiv:2112.01579*, 2021.
[161] S. Weiss, M. Işik, J. Thies, and R. Westermann. Learning adaptive sampling and reconstruction for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2020. Accepted.
[162] M. Werhahn, Y. Xie, M. Chu, and N. Thuerey. A multi-pass GAN for fluid flow super-resolution. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(2):1–21, 2019.
[163] S. Wiewel, M. Becher, and N. Thuerey. Latent space physics: Towards learning the temporal evolution of fluid flow. *Computer Graphics Forum*, 38(2):71–82, 2019.
[164] S. Wiewel, B. Kim, V. C. Azevedo, B. Solenthaler, and N. Thuerey. Latent space subdivision: Stable and controllable time predictions for fluid flow. *Computer Graphics Forum*, 39(8):15–25, 2020.
[165] A. Wu, Y. Wang, X. Shu, D. Moritz, W. Cui, et al. AI4VIS: Survey on artificial intelligence approaches for data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2021. Accepted.
[166] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, et al. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.
[167] S. W. Wurster, H.-W. Shen, H. Guo, T. Peterka, M. Raj, et al. Deep hierarchical super-resolution for scientific data reduction and visualization. *arXiv preprint arXiv:2107.00462*, 2021.
[168] Y. Xie, E. Franz, M. Chu, and N. Thuerey. tempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow. *ACM Transactions on Graphics*, 37(4):95:1–95:15, 2018.
[169] C. Yang, Y. Li, C. Liu, and X. Yuan. Deep learning-based viewpoint recommendation in volume visualization. *Journal of Visualization*, 22(5):991–1003, 2019.
[170] Q. Yao, M. Wang, H. J. Escalante, I. Guyon, Y.-Q. Hu, et al. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306*, 2018.
[171] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 3634–3640, 2018.
[172] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, et al. A survey of visual analytics techniques for machine learning. *Computational Visual Media*, 7:3–36, 2021.
[173] Q. Zhang and S.-C. Zhu. Visual interpretability for deep learning: A survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.
[174] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *Proceedings of European Conference on Computer Vision*, pages 649–666, 2016.
[175] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.
[176] S. Zhang, H. Tong, J. Xu, and R. Maciejewski. Graph convolutional networks: A comprehensive review. *Computational Social Networks*, 6:11:1–11:23, 2019.
[177] Y. Zhang, H. Ling, J. Gao, K. Yin, J.-F. Lafleche, et al. DatasetGAN: Efficient labeled data factory with minimal human effort. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 10145–10155, 2021.
[178] Y. Zhang and Q. Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021. Accepted.
[179] Z. Zhang, P. Cui, and W. Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):249–270, 2022.
[180] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021.
[181] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, et al. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.
[182] Z. Zhou, Y. Hou, Q. Wang, G. Chen, J. Lu, et al. Volume upscaling with convolutional neural networks. In *Proceedings of Computer Graphics International*, pages 38:1–38:6, 2017.
[183] F. Zhu, Y. Pan, T. Gao, H. Walia, and H. Yu. Interactive visualization of hyperspectral images based on neural networks. *IEEE Computer Graphics and Applications*, 41(5):57–66, 2021.
[184] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.
[185] Y. Zhu and Y. Wang. Student customized knowledge distillation: Bridging the gap between student and teacher. In *Proceedings of IEEE International Conference on Computer Vision*, pages 5057–5066, 2021.
[186] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In *Proceedings of International Conference on Learning Representations*, 2017.



Chaoli Wang is an associate professor of computer science and engineering at the University of Notre Dame. He received a Ph.D. degree in computer and information science from The Ohio State University in 2006. Dr. Wang's primary research interest is data visualization, particularly on the topics of time-varying multivariate data visualization, flow visualization, information-theoretic algorithms, graph-based techniques, and deep learning solutions for big data analytics. He is an associate editor of IEEE Transactions on Visualization and Computer Graphics.

Jun Han is a Ph.D. candidate at the University of Notre Dame. He received a BS degree in software engineering and an MS degree in computer software and theory in 2014 and 2017. Both degrees are from Xidian University. His Ph.D. research focuses on applying deep learning techniques to solve data visualization problems.