Biomimetic IGA neuron growth modeling with neurite morphometric features and CNN-based prediction

Kuanren Qian[a], Ashlee S. Liao[a], Shixuan Gu[b], Victoria A. Webster-Wood[a,b,c], Yongjie Jessica Zhang[a,b]

[a]*Department of Mechanical Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA*
[b]*Department of Biomedical Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA*
[c]*McGowan Institute for Regenerative Medicine, University of Pittsburgh, 450 Technology Drive, Pittsburgh, PA 15219, USA*

Abstract

Neuron growth is a complex, multi-stage process that neurons undergo to develop sophisticated morphologies and interwoven neurite networks. Recent experimental research advances have enabled us to examine the effects of various neuron growth factors and seek potential causes for neurodegenerative diseases, such as Alzheimer's disease, Parkinson's disease, and amyotrophic lateral sclerosis. A computational tool that studies the neuron growth process could shed crucial insights on the effects of various factors and potentially help find a cure for neurodegeneration. However, there is a lack of computational tools to accurately and realistically simulate the neuron growth process within reasonable time frames. Bio-phenomenon-based models ignore potential neuron growth factors and cannot generate realistic results, and bio-physics-based models require extensive, high-order governing equations that are computationally expensive. In this paper, we incorporate experimental neurite features into a phase field method-based neuron growth model using an isogeometric analysis collocation (IGA-C) approach. Based on a semi-automated quantitative analysis of neurite morphology, we obtain relative turning angle, average tortuosity, neurite endpoints, average segment length, and the total length of neurites. We use the total neurite length to determine the evolving days *in vitro* (DIV) and select corresponding neurite features to drive and constrain the neuron growth. This approach archives biomimetic neuron growth patterns with automatic growth stage transitions by incorporating corresponding DIV neurite morphometric data based on the total neurite length of the evolving neurite morphology. Furthermore, we built a convolutional neural network (CNN) to significantly reduce associated computational costs for predicting complex neurite growth patterns. Our CNN model adopts a customized convolutional autoencoder as the backbone that takes neuron growth simulation initializations and target iteration as the input and predicts the corresponding neurite patterns. This approach achieves high prediction accuracy (97.77%) while taking 7 orders of magnitude less computational times when compared with our IGA-C neuron growth solver.

Keywords: Neuron growth, Neurite morphometric features, Isogeometric collocation, Phase field method, Deep learning, Convolution neural network, Auto-encoder

1. Introduction

In recent years, we have witnessed significant outgrowth in the neuron growth research field, and multiple attempts to prevent neurodegenerative diseases with neuron growth factors have been made. Understanding nervous system development is critical in searching for potential neurodegenerative disease treatments. Recent molecular biology research has revealed many possible effects of neuron growth factors during disease pathogenesis [1, 2]. Alzheimer's disease, Parkinson's disease, and amyotrophic lateral sclerosis are three devastating neurodegenerative disorders with high morbidity and mortality for patients [3]. These diseases arise when neurons progressively lose connections or functions due to alterations of certain neuron growth factors [4, 5]. Because most mammalian neurons cannot intrinsically regenerate, they can not repair or replace themselves, making traditional treatments ineffective against neurodegeneration [6, 7]. Although evidence shows that certain factors from developing neurons can help protect mature neurons from degeneration [5], there is a lack of study of the specific functional role of these factors [8]. Therefore, understanding the factors involved during neuron growth is vital for seeking potential neurodegeneration treatments. Neuron growth is a complex multi-stage process consisting of multiple stages of development, each with

unique growth behaviors [9]. Primary rat hippocampal neurons *in vitro* have exhibited five key developmental stages in their morphology [10]. In Stage 1, lamellipodia protrude from the initial neuron cell and extend into neurites of similar length in Stage 2. The longest neurite will differentiate into a longer axon in Stage 3, while other neurites slowly grow in Stage 4. Stage 5 is neuron maturation, in which neurons grow intricate patterns and form complex neurite networks. This process spans several days *in vitro* (DIV), and each stage exhibits drastically different growth behaviors driven by intracellular and extracellular biophysics processes that involve a wide range of neuron growth factors [11, 12].

Considering the aggregating costs and arduous procedures needed for extensive cell culture experiments, mathematical modeling of early neuron growth stages has been proposed to study neuron growth factor during the initial neurite outgrowth [13], axon differentiation [14, 15], and axon guidance [16]. Among different neuron growth models, there exist two major schools of thought. One approach models neuron growth using stochastic methods that follow phenomenological results [17, 18]. Along this direction, some literature models the growth process using axon steering theory based on filopodia [19], external repulsive cues [20], and stochastic mechanism [21]. Generalized neurite characteristics are utilized based on morphology [22, 23], and the surrounding substrate is incorporated [24]. While efficient to compute, these methods take limited biophysics into account. In contrast, models that attempt to incorporate the effects of different neuron growth factors based on biophysical mechanisms behind neurite elongations are prone to high computational cost and numerical instability [25, 26]. Despite the aforementioned approaches to model the neuron growth process, most of them lack the ability to efficiently and accurately model realistic neuron growth. Many rely heavily on broad assumptions or are associated with expensive computational costs, yet are still unable to capture complex neurite morphology changes throughout the growth process.

Inspired by [27], we developed a phase field method-based neuron growth model with distinctive growth stage transitions [28] to address these limitations in neuron growth modeling. Our isogeometric analysis collocation (IGA-C) neuron growth model archives multi-stage neuron growth by considering the effect of intracellular tubulin concentration. By adjusting the assembly and disassembly rate of the tubulin transport coupled with the phase field model, the model could simulate axon differentiation through different neurite elongation rates based on tubulin concentration. Yet, despite our implementation of biophysics-based approaches, our neuron growth model relies on arbitrarily set growth stage transitions and can not capture similar biomimetic growth behaviors without extensive manual parameter adjustments for a particular type of neuron. As a follow-up, in this paper we incorporate experimentally observed neurite morphometric features into the biophysics-based IGA-C neuron growth model to simulate biomimetic neuron growth behaviors with intrinsic growth stage transitions determined by neurite morphology. Based on the semi-automated quantitative neurite morphometric evaluation [29], the developed neuron growth model uses the total neurite length to determine the evolving DIV and select neurite morphometrics of the corresponding DIV to constrain and drive neurite growth behaviors. The model can be easily adapted to different types of neurons, given the corresponding neurite morphometrics.

Note that our neuron growth simulations with incorporated neurite features require days to complete, and the associated computational costs arise rapidly for multiple neuron cases as the domain increases in size. In addition, the neurite morphometric feature implementation requires evaluating each incorporated neurite feature within each iteration. In recent years, deep learning has been used in solving nonlinear high-dimensional partial differential equations (PDEs) [30], and physics-informed neural networks [31] were developed. They have significantly alleviated computational costs without undermining analysis accuracy when studying a 2D reaction-diffusion system [32], intracellular material transport process [33], and traffic jams in complex 3D neuron structures [34]. In this paper, we propose to use a convolutional neural network-based (CNN) surrogate model to accurately predict intricate neurite growth patterns with significantly lower computational costs than conventional simulations. The main contributions of the presented neurite morphometric feature-driven IGA-C growth model and CNN-based prediction include:

- Incorporating experimentally observed neurite features into the IGA-C phase field neuron growth model to drive and constrain biomimetic neuron growth behaviors with seamless intrinsic growth stage DIV transitions;
- An IGA-C neuron growth model that simulates rat hippocampal neuron growth process and is adaptable to different types of neurons given neurite morphometric data of the growth behaviors; and
- A deep learning model based on CNN architecture, capable of accurate and fast predictions of complex neurite networks. The model predicts neurite growth across different growth stages without an iterative process.

The remainder of this paper is organized as follows. Section 2 reviews IGA-C-based phase field neuron growth. Section 3 elaborates on the neurite morphometric features-driven neuron growth model that simulates biomimetic neurite growth with intrinsic growth stage transition. Section 4 presents a CNN-based surrogate model for neuron growth and showcases prediction results. Finally, we draw conclusions and discuss future work.

2. Review of IGA-C-based phase field neuron growth modeling

In this section, we review the phase field neuron growth model, IGA, and collocation method. IGA and phase field methods are powerful numerical techniques utilized in simulating complex engineering problems. IGA is a high-order numerical technique used in computational mechanics to approximate governing equations continuously, eliminating the need for discretization in traditional finite element methods [35]. The phase field method solves evolving boundary problems like fracture and dendrite solidification [36]. Combining these two techniques can effectively simulate neurite growth with high accuracy.

2.1. Phase field neuron growth model

In our previous work, we developed an IGA-C-based phase field model to simulate the multi-stage neuron growth process by introducing the effect of tubulin concentration [28]. By considering intracellular transport during the growth, our model differentiates the longest neurite into an axon. We simulated different growth behaviors during each stage to reproduce growth cone behaviors at neurite tips. By treating the neuron growth domain as a binary phase field ϕ with phase "1" representing the neuron and phase "0" representing the surrounding medium, we solved the phase field governing equations using the IGA-C method to simulate neurite growth patterns based on given neuron configurations and parameters. In particular, we used IGA-C to solve the phase field equation (Eq. 1) coupled with the tubulin equation (Eq. 2) and the neuroattractant equation (Eq. 3) through an energy term E (Eq. 4) and competitive tubulin consumption term $\frac{dL}{dt}$ (Eq. 5). In the phase field models, heat conduction was used to model dendrite solidification at the interface in spherulites [37]. We borrow this idea to model the diffusion process of releasing synaptogenesis particles such as neuroattractants, and use its gradient field for evaluating the energy term (the driving force) to evolve the boundary based on $\frac{\partial \phi}{\partial t}$. The diffusion of neuroattractants is responsible for growth cone pathfinding in the nervous system during synaptogenesis [38].

We provide a detailed list of variables in Table 1 for clarity. The main governing equations are defined as:

$$\frac{\partial \phi}{\partial t} = M_\phi[\nabla \cdot (a(\Psi)^2 \nabla \phi) - \frac{\partial}{\partial x}(a(\Psi)\frac{\partial a(\Psi)}{\partial \Psi}\frac{\partial \phi}{\partial y}) + \frac{\partial}{\partial y}(a(\Psi)\frac{\partial a(\Psi)}{\partial \Psi}\frac{\partial \phi}{\partial x}) + \phi(1-\phi)(\phi - 0.5 + E + 6H|\nabla \theta|)], \tag{1}$$

$$\frac{\partial(\phi\, c_{tub})}{\partial t} = \delta_t \nabla \cdot (\phi \nabla c_{tub}) - \alpha_t \cdot \nabla(\phi\, c_{tub}) - \beta_t(\phi\, c_{tub}) + \epsilon_0 \frac{|\nabla(\phi_0)|^2}{\int |\nabla(\phi_0)|^2 \, d\Omega}, \tag{2}$$

$$\frac{\partial c_{syn}}{\partial t} = D_c \nabla^2 c_{syn} + K\frac{\partial \phi}{\partial t}, \tag{3}$$

$$E = \frac{\alpha}{\pi}\tan^{-1}(H_\epsilon(\frac{dL}{dt})\gamma \triangle c_{syn}), \tag{4}$$

$$\frac{dL}{dt} = r_g\, c_{tub} - s_g, \tag{5}$$

where M_ϕ is the phase field mobility coefficient, $a(\Psi)$ is the anisotropy coefficient for the gradient, E is the energy term for phase field growth, H is a constant value, and θ is the orientation term. In Eq. 2, δ_t is the tubulin diffusion rate, α_t is the tubulin active transport coefficient, β_t is the tubulin decay coefficient, and $\epsilon_0 \frac{|\nabla(\phi_0)|^2}{\int |\nabla(\phi_0)|^2 \, d\Omega}$ is the constant tubulin production source term, where ϕ_0 is the initial phase field variable and ϵ_0 is the dimensionless production coefficient term. In Eq. 3, c_{syn} is the concentration of synaptogenesis particles such as neuroattractants, and K is the dimensionless latent neuroattractant coefficient. In Eq. 4, E is the energy term responsible for driving cell growth at the interface between phases. $\frac{\alpha}{\pi}$ is a scaling coefficient, H_ϵ is the Heaviside step function, and γ is the phase field interfacial energy constant. In Eq. 5, r_g and s_g are the assembly and disassembly rate of tubulin concentration within the cell boundary [39] to incorporate the effect of intracellular tubulin concentration.

Table 1: Parameters utilized in the phase field neuron growth model.

Parameter	Description	Value	Parameter	Description	Value
ϕ	phase field variable	-	γ	phase field interface energy constant	10
M_ϕ	Mobility coefficient	60	c_{tub}	Tubulin concentration	-
$a(\Psi)$	Neurite morphology gradient coefficient	-	δ_t	Tubulin diffusion rate	4 ($\mu m^2/h$)
H	Orientation constant coefficient	0.007	α_t	Tubulin active transport rate	0.001 ($\mu m/h$)
θ	Neurite growth orientation angle	[0, 1]	β_t	Tubulin decay coefficient	0.001 ($1/h$)
c_{syn}	Neuroattractant concentration	-	ϵ_0	Tubulin production coefficient	15
D_c	Neuroattractant diffusion coefficient	-	ϕ_0	Initial phase field variable	-
K	Dimensionless latent neuroattractant	2	$\frac{dL}{dt}$	Competitive tubulin consumption	-
E	Energy	-	r_g	Tubulin assembly rate	5
$\frac{\alpha}{\pi}$	Scaling coefficient	0.2865	s_g	Tubulin disassembly rate	0.1
H_ϵ	Heaviside function	-			

Note: Default value is given if the parameter requires initialization. For dimensionless parameters, we show the default value without units.

The neuron growth model achieves neurite outgrowth by iteratively solving interface evolution using the phase field governing equation (Eq. 1). The first few terms involving $a(\Psi)$ introduce anisotropy into the neuron growth model [27]. The following term involving E is a double well function with a coupling orientation term, $6H|\bigtriangledown\theta|$, introduced in [40] to break dendrite growth symmetry. In the phase field model, θ indicates the change in the direction of the extending neurites. The initial orientation θ is set as random values between $[0, 1]$ throughout the domain and the cell is initialized as a filled circle at the center of the binary domain ϕ, where we consider $\phi_0 = 1$ in the cell and $\phi_0 = 0$ in the medium. The seed radius of the cell is set as $r_0 = 20\,\triangle x$. We initialize the tubulin concentration in the cell as $c_{tub} = \frac{1}{2}(1 + tanh((r_0 - r)/2))$. In our earlier neuron growth model [28], we

- Developed a phase field model coupled with intracellular tubulin concentration to simulate the first four stages of neuron growth, including lamellipodia formation, initial neurite outgrowth, axon differentiation, and dendrite formation; and
- Implemented relative turning angle from experimental observations to guide neurite growth and demonstrated similar end-stage neuron morphology reproduction; and
- Extended the model to simulate the growth of neural circuits, where multiple neurons build neurite interactions and form neural networks.

Discussion. Despite the attempts made in the aforementioned model [28] to simulate growth behaviors, the model still relies on arbitrarily-set iterations for each growth stage and can not accurately depict intrinsic growth stage transition. In addition, the model develops excessive branchings at later stages not observed in experimental cultures. Moreover, parameters used in the model were manually tuned for rat hippocampal neuron behaviors, making it less applicable to other types of neurons. These limitations undermine the practical implementations of the model. In this paper, we propose incorporating neurite morphometric features from experimental cultures as a modular component into the neuron growth model to overcome these limitations. Our proposed approach will improve the model automation and its ability to simulate more biomimetic growth behaviors of a diverse range of neurons.

2.2. Isogeometric-collocation (IGA-C) method

In this work, we utilize the IGA-C method to solve the phase field neuron growth model for its flexibility and accuracy [41, 42]. High-fidelity modeling for complex neuron morphologies is a significant challenge in computational biology [43, 44]. Using high-order spline basis functions, we can generate an accurate analysis of the phase field model. Together with the collocation method, we directly solve the strong form of the phase field equation on Greville Abscissae collocation points [45, 46].

B-splines basis function. In this paper, we choose cubic B-splines as the basis functions for our IGA-C solver [47, 48]. For a univariate B-spline curve of degree p, it is defined using a sequence of nondecreasing real numbers, u_i, to construct an open knot vector $U = \{u_1, u_2, \cdots u_{n+p+1}\}$, in which p is the B-spline order, and n is the number of

basis functions. The basis function $N_{i,p}(u)$ is defined based on the knot vectors, we have:

$$N_{i,0}(u) = \begin{cases} 1, & \text{if } u_i \leq u \leq u_{i+1} \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u), \quad p = 1, 2, ... \tag{7}$$

where $N_{i,0}(u)$ is piece-wise linear constant. $N_{i,p}(u)$ is recursively defined based on a combination of preceding basis functions. With control points $P = \{P_i\}_{i=1}^{n}$, we can construct a p^{th}-order B-spline curve $C(u)$ as:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i, \quad 0 \leq u \leq 1 \tag{8}$$

where $N_{i,p}(u)$ is the B-spline basis function of p degree, defined over the knot vector U. Given an arbitrary u value in the knot vector U, we can compute the point on the B-spline curve by multiplying the value of every non-zero basis function $N_{i,p}(u)$ with its control point P_i and taking a summation. For a 2D surface, we define the basis functions as the tensor product of two univariate B-splines.

Greville abscissae collocation points. The Greville abscissae collocation method is a numerical technique for solving strong form of PDEs [49, 46]. Using Greville abscissae points, it is relatively straightforward to implement and accurate for problems with smooth solutions. It is also adaptable to a wide range of high-order differential equations. For IGA, collocation methods have been recently shown to perform well as an alternative to Galerkin methods [50, 51] while maintaining higher order convergence rates [52]. The Greville abscissae point $\hat{u}_i$ is defined based on knots in the knot vector $U = \{u_1, u_2, \cdots u_{n+p+1}\}$ as:

$$\hat{u}_i = \frac{u_{i+1} + ... + u_{i+p}}{p}, \quad i \in [1, n] \tag{9}$$

which can be directly computed on given B-spline surfaces [52]. In the context of neuron growth, we can rewrite each collocation point as $\hat{\phi} = \{\hat{\phi}_u, \hat{\phi}_v\}$:

$$\hat{\phi}_u = \frac{\sum_{i+1}^{i+p} u}{p} \text{ and } \hat{\phi}_v = \frac{\sum_{j+1}^{j+p} v}{p}, \tag{10}$$

where $\hat{\phi}_u$ and $\hat{\phi}_v$ are the components along each parametric direction of the collocation point $\hat{\phi}$. By directly solving the strong form of the phase field neuron growth model, we obtain accurate and smooth results.

3. Neurite morphometric features-driven neuron growth

In this section, we first review experimental neuron culture procedures and semi-automated neurite morphometric evaluation. Then, we present the computational pipeline for the feature-driven neuron growth model and go through each implemented module in detail. Finally, we discuss our simulation results.

3.1. Experimental neuron culture and semi-automated neurite morphometric evaluation

Neuron cell culture. Neuron growth exhibits complex morphological characteristics as illustrated in Figure 1A, and behavior changes throughout the process. For the work presented here, data was used from an existing experimental neuron growth dataset of bright-field microscopy images [29]. Briefly, embryonic rat hippocampal neurons were cultured over 6 DIV, and the first week of *in vitro* morphological development was imaged and analyzed. Cryopreserved primary, embryonic-day 18 (E18) rat hippocampal neurons (A36513, Gibco, USA) were cultured in dishes coated with poly-D-lysine (P6407, Sigma-Aldrich, USA) following manufacturer protocol [53]. Neurons at a density of $10,000$ cells/cm^2 were seeded in Neurobasal Plus (A3582901, Gibco, USA) supplemented with 2% B-27 Plus (A3582801, Gibco, USA). Except during the media change and microscopy imaging, the incubation was conducted at 37 $^{\circ}C$ with 5% CO_2. 50% culture media was replenished with fresh media after 24 hours. The imaging process utilized the Echo Revolve Microscope (Echo Revolve | R4, inverted, BICO - The Bio Convergence Company, USA)

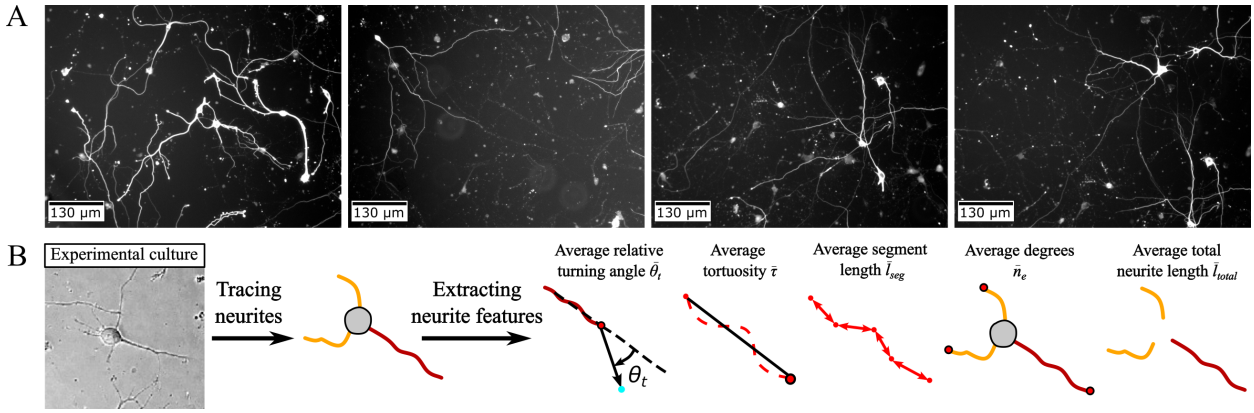

Figure 1: Neuron growth images and neurite morphometric features. (A) Fluorescent images of rat hippocampal neurons to illustrate the complexity of neuron structure. (B) Neurite morphometric features were extracted from experimental images of neuron cultures using semi-automated neurite morphometric evaluation [29], including average relative turning angle $\bar{\theta}_t$, average tortuosity $\bar{\tau}$, average segment length $\bar{l}_{seg}$, average degree $\bar{n}_e$, and average total neurite length $\bar{l}_{total}$.

with a 12-megapixel color camera in the inverted configuration. The magnification was set at 20X or 40X, and cells were imaged at 0.5, 1, 2, 3, 4, and 6 DIV.

Experimentally-derived neurite morphometric features. Neurite morphometric features used to drive the growth model in this work were taken from our prior neuron growth dataset [29]. We leveraged the distributions of representative neurite morphological features per cell (Figure 1B), including average relative turning angle $\bar{\theta}_t$, average tortuosity $\bar{\tau}$, average segment length $\bar{l}_{seg}$, neurite degrees (the number of neurite endpoints/tips $\bar{n}_e$), and the average total length $\bar{l}_{total}$ as a dataset shown in Table 2 [12, 29]. The relative turning angles $\bar{\theta}_t$ is the angle change between consecutive change point segments. The tortuosity $\bar{\tau}$ is calculated by dividing the length of the neurite tracing by the distance between its endpoints. The average segment length $\bar{l}_{seg}$ is the average distance between change points of all neurites for a given cell. Degree $\bar{n}_e$ is the number of neurite endpoints for a given cell. The total length $\bar{l}_{total}$ is the length of all traced neurites part of a given cell.

Table 2: Neurite morphometric features statistics from DIV 0.5 to 6 from rat hippocampal neurons [29].

DIV	$\bar{\theta}_t(^\circ)$		$\bar{\tau}$		$\bar{n}_e$		$\bar{l}_{seg}(\mu m)$		$\bar{l}_{total}(\mu m)$
	μ	σ	Q_1	Q_3	Q_1	Q_3	Q_1	Q_3	
0.5	22.18	11.65	1.0225	1.0776	1	2.5	5.05	7.82	27.53
1	22.82	10.82	1.0161	1.0757	1	3	5.58	8.36	36.54
1.5	21.34	9.69	1.0254	1.0507	1	4	5.64	9.39	53.19
2	22.32	8.01	1.0283	1.0685	1	4	6.39	10.35	84.34
3	22.88	6.78	1.0300	1.0725	2	6	8.35	10.95	155.13
4	21.28	4.91	1.0341	1.0623	3	7	6.65	10.86	218.74
6	20.32	2.65	1.0302	1.0498	6	10	10.24	12.36	554.73

Note: μ and σ are the mean and standard deviation, respectively. Q_1 and Q_3 are the 1st and 3rd quartiles of the features distribution.

3.2. Incorporating neurite morphometric features

Our previous neuron growth model [28] only utilizes relative turning angles θ_t extracted from experimental data without adjustments throughout the entire simulation process. Therefore, it can not intrinsically transition between adjacent neuron growth stages (DIV), leading to unnecessarily dense neurite branching during the simulation. In this paper, we incorporate DIV-based neurite morphometric features from embryonic rat hippocampal neuron cultures into our neuron growth model to better capture neurite morphological characteristics that vary across multiple growth stages intrinsically. We provide a detailed explanation of parameters used in the feature-driven neuron growth procedure in Table 3, and show the overall computational pipeline in Figure 2. Our feature-driven neuron growth consists of five modules: IGA-C neuron growth solver (Module A), growth cone calculation (Module B), DIV determination

(Module C), potential tip detection (Module D), and feature-based tip adjustment (Module E). Module A has been reviewed in Section 2.1. We explain Modules B-E in detail below. Parameters in Table 3 are categorized based on these four modules.

Table 3: Parameters utilized in feature-driven neuron growth procedure.

Module	Parameter	Description	Module	Parameter	Description
Module B	S_{neu}	Sites of detected neurons in ϕ	Module D	ζ_{tip}	Tip intensity threshold
	S_{gc}	Site of growth cones in ϕ		n_{tips}	Number of neurite tips detected
	n_{neu}	Number of neurons	Module E	l_{seg}	Evolving neurite segment length
	S_{tips}	Sites of detected tips		S_{tr}	Site of neurite tracings
	l_{gc}	Size of growth cone		l_{neu}	Neurite length
	$\mathcal{H}_{conv}$	Convolution box filter		d_{geo}	Calculated geodesic distance
Module C	$P_{initial}$	Initial neuron cell center		θ_t	Calculated relative turning angle
	l_{total}	Evolving total neurite length		τ	Calculated tortuosity
	$\bar{l}_{total}$	Average total neurite length		(x_{tip}, y_{tip})	Neurite tip coordinates
	$\bar{\theta}_t$	Average relative turning angle		(x_{root}, y_{root})	Neurite root coordinates
	$\bar{\tau}$	Average tortuosity	Algorithms 1&2	ζ_{soma}	Soma geodesic threshold
	$\bar{n}_e$	Average number of neurite tips		I	Intensity field
	$\bar{l}_{seg}$	Average neurite segment length		A_{tip}	Area of the tip
				γ_{tip}	Tip area threshold

Note: Parameters ($\bar{l}_{total}$, $\bar{\theta}_t$, $\bar{\tau}$, $\bar{n}_e$, and $\bar{l}_{seg}$) with a bar on top are obtained from experimental neuron cultures.

Growth cone calculation (Module B). Based on the IGA-C neuron growth solver (Module A), we incorporate the effect of various neurite morphometric features into the model by adjusting competitive tubulin consumption $\frac{dL}{dt}$ at growth cones of neurite tips. This is achieved by adjusting M_ϕ, r_g, and s_g values to change tubulin concentration balance at neurite growth cones S_{gc}. S_{gc} is determined using a feature-driven neuron growth procedure (Modules B-D) that takes ϕ as input and outputs corresponding S_{gc} for each neuron based on neurite morphometric features of the corresponding DIV in Table 2. During the neuron growth simulation, we run the IGA-C neuron growth solver until the final DIV is reached by iteratively determining the evolving DIV based on the total neurite length $\bar{l}_{total}$ (Figure 3). In each iteration, we apply the feature-driven neuron growth procedure to adjust $\frac{dL}{dt}$, which affects the magnitude of E (Eq. 4) for the phase field governing equation (Eq. 1) to drive neurite outgrowth. During the procedure, we first use MATLAB's connected component analysis to identify neurons S_{neu} within the phase field ϕ. Then, for each identified neuron, we search for potential neurite tips and identify their locations S_{tips} using a convolution operation with a box filter $\mathcal{H}_{conv}$, which will be explained later in Module D. With identified neurite tips S_{tips}, we can generate growth cones S_{gc} by selecting S_{tips} neighboring area based on the growth cone size l_{gc} (Module E). To simulate axon differentiation, we adjust mobility term M_ϕ on S_{gc} furthest from the initial neural cell center $P_{initial}$ to 100 while keeping M_ϕ on the rest S_{gc} as default 60. The M_ϕ magnitude difference among neurite tips allows the longest neurite to grow faster than the rest, therefore achieving axon differentiation. We then increase the r_g value to 50 and decrease the s_g value to 0 to drive neurite outgrowth at S_{gc} as shown in Module B. Based on semi-automated neurite morphometric features [12, 29] shown in Table 2, we can obtain relative turning angle $\bar{\theta}_t$, average tortuosity $\bar{\tau}$, neurite endpoints $\bar{n}_e$, average segment length $\bar{l}_{seg}$, and the total neurite length, $\bar{l}_{total}$ to drive and constrain the neuron growth model.

DIV determination (Module C). To achieve intrinsic DIV transition during simulation, we need to correctly determine the evolving DIV for S_{neu} during the simulation. As shown in Module C, we incorporate an automatic neurite tracing algorithm (Algorithm 1) into the computational pipeline to calculate the total neurite length, l_{total}, and use it to determine the current DIV. Algorithm 1 takes neuron growth phase field ϕ as input and outputs all primary, secondary, and tertiary neurite tracings based on their morphology connections in the neurite structure. In this process, we first calculate the corresponding neuron cell initial coordinate $P_{initial}$ based on centroids of detected S_{neu}. Then, the algorithm calculates the geodesic distance, d_{geo}, within the neuron cell from $P_{initial}$. To correctly differentiate neurites from the soma in a binary domain, we leverage a soma geodesic threshold ζ_{soma}. Neurite tracing is then achieved by propagating from the maximum geodesic value, $max(d_{geo})$, to the minimum geodesic value, $max(d_{geo})$, in the same connected neurite. As shown in Figure 3, once primary neurite tracing is complete, the algorithm repeats the tracing procedure and removes traced neurites to differentiate among primary, secondary, and tertiary tracings. Finally, based on obtained neurite tracings, we can calculate the total neurite length, l_{total}, by summing up the geodesic distance of

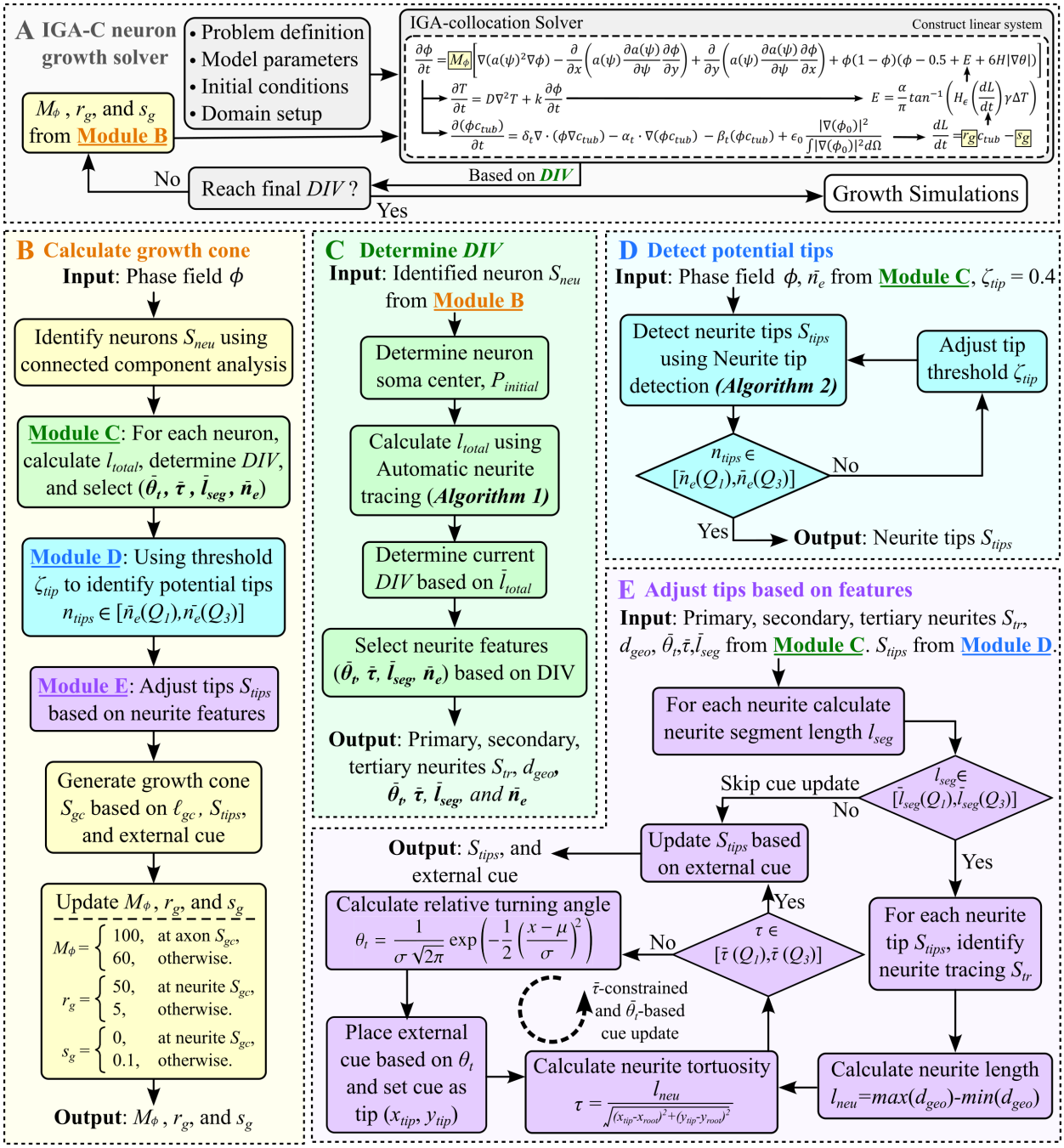

Figure 2: IGA-C neuron growth computational pipeline incorporating neurite morphometric features extracted from experiments. (A) IGA-C phase field neuron growth solver. (B) Updating neuron growth cone to guide neurite outgrowth. (C) Intrinsic growth stage transition based on Algorithm 1 and determined DIV. (D) Detecting potential tips using Algorithm 2. (E) Neurite morphometric features-based tip adjustments.

each neurite, l_{neu}, and determine the corresponding DIV.

Algorithm 1 Automatic Neurite Tracing (Figure 3)

Input: Neurite growth field ϕ
Output: Neurite tracing and total neurite length l_{total}
Procedure Primary neurite tracing
1: Obtain binary neurite growth, $\lfloor \phi \rceil$
2: Calculate the geodesic distance, d_{geo}, from the center of soma $P_{initial}$ to each tip
3: Remove soma region: Set $\lfloor \phi \rceil = 0$ where $d_{geo} > \zeta_{soma}$
4: **for** each neurite **do**
5: Propagate and trace from $max(d_{geo})$ to $min(d_{geo})$ to obtain neurite tracings S_{tr}
6: **end for**
7: Set d_{geo} along S_{tr} to 0 to remove traced neurites
Procedure Secondary and tertiary neurite tracing
8: Repeat Step 3&4 to obtain secondary and tertiary neurites
Procedure Total neurite length calculation
9: **for** each neurite tracing **do**
10: Calculate each neurite length: $l_{neu} = max(d_{geo}) - min(d_{geo})$
11: **end for**
12: Compute total neurite length: $l_{total} = \sum l_{neu}$

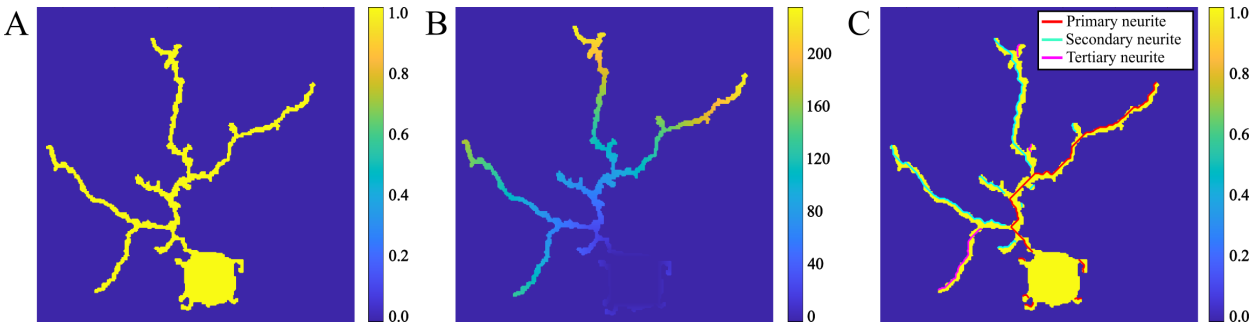

Figure 3: Neurite tracing procedure (Algorithm 1) that takes ϕ as input and traces three generations of neurites. (A) Input phase field ϕ. (B) Geodesic distance d_{geo} in neuron from $P_{initial}$ to neurite tips S_{tips}. Tracing is achieved by propagating along neurites based on d_{geo}. (C) Output all neurite tracings S_{tr} and the sum of all neurite lengths.

Potential tip detection (Module D). To simulate neurite outgrowth in the IGA-C phase field neuron growth model, we carry out a "growth-cone" like activation of the energy term E, as shown in Module D. This approach allows us to incorporate the number of neurite endpoints $\bar{n}_e$ from experiments into the neuron growth model. We develop a neurite tip detection algorithm (Algorithm 2) with an adjustable threshold value ζ_{tip} to constrain the number of tips S_{tips}. The tip detection algorithm can automatically detect neurite tips based on a given ζ_{tip} of 0.4. In Figure 4, the algorithm takes neurite growth pattern ϕ as input and outputs a list of potential tips in the domain. The neurite tips are detected as the locations of high intensity magnitude with fewer neighbors than the rest of the neurite regions (Figure 4C). First, the algorithm calculates an intensity field I by applying convolution on ϕ using a box filter $\mathcal{H}_{conv}$:

$$I(m,n) = \phi(m,n) \otimes \mathcal{H}_{conv}(m,n) = \sum_{i=1}^{l_{kl}} \sum_{j=1}^{l_{kl}} \mathcal{H}_{conv}(i,j) \cdot \phi(m-i, n-j), \qquad 1 \le m \le M, \quad 1 \le n \le N, \tag{11}$$

where $\mathcal{H}_{conv}$ is the convolution kernel, and l_{kl} is kernel size. M and N are the dimensions fo the domain. Then, we can identify tips by extracting sites with intensity magnitudes lower than $\zeta_{tip} \times max(I)$ (Figure 4D). Based on the 1^{st} and 3^{rd} quartiles of $\bar{n}_e$ given in Table 2, we can adjust ζ_{tip} to constrain the number of detected tips to fit within the range.

Algorithm 2 Neurite tip detection (Figure 4)

Input: Neurite growth phase field ϕ, $\zeta_{tip} = 0.4$, and $\gamma_{tip} = 10$
Output: Detected tips S_{tips}

Procedure Detect potential neurite tip locations S_{tips}
Goal: To obtain tips by extracting areas with intensity I lower than ζ_{tip}
Calculate intensity I using convolution on binary ϕ: $I = \lfloor \phi \rceil \otimes \mathcal{H}_{conv}$
if $I > \zeta_{tip} \times \max(I)$ **then**
    $I = 0$
end if
Procedure Filter out tips with small areas (noise)
Goal: To obtain tip centroids S_{tips} and tip areas A_{tip} using connected component analysis
for each detected tip **do**
    if $A_{tip} \leq \gamma_{tip}$ **then**
        $S_{tips} = 0$
    end if
end for

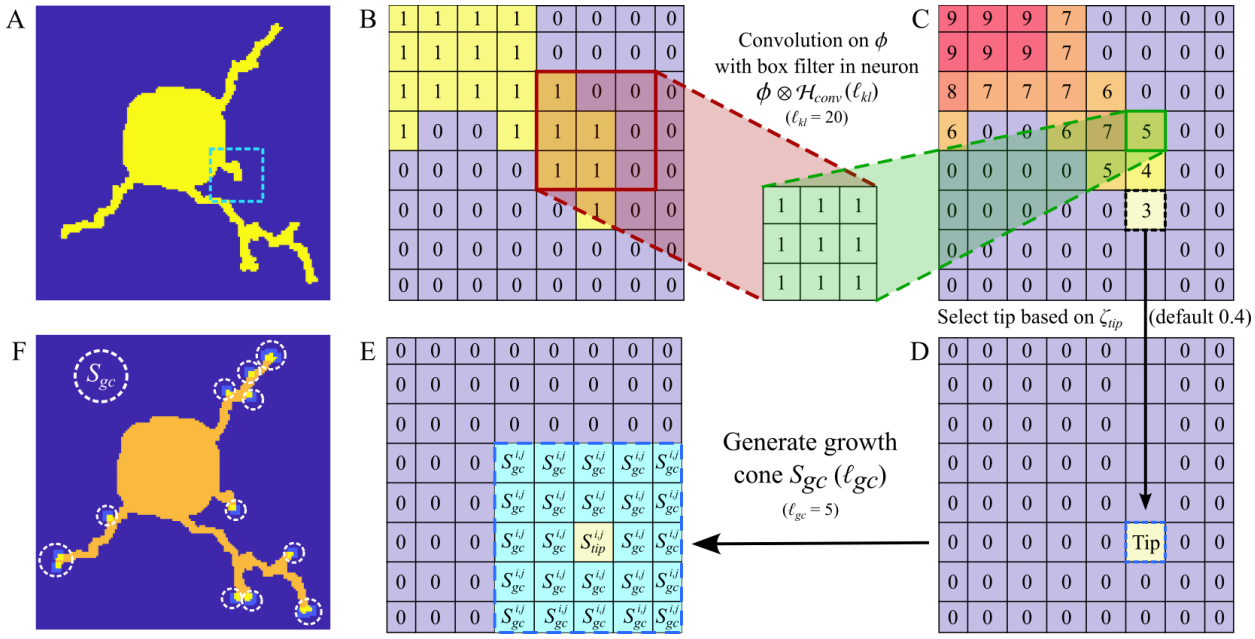

Figure 4: Tip detection procedure (Algorithm 2). (A) Input ϕ field. (B) A zoomed-in view of the tip area (the cyan dashed square in (A)). (C) Intensity field I is calculated using convolution to ϕ inside the neuron cell (where $\phi = 1$). For visualization, l_{kl} is set as 3 (default value is 20). (D) Detecting potential neurite tips S_{tips} based on tip intensity threshold $\zeta_{tip} \times max(I)$. ζ_{tip} is set to 0.4. (E) Growth cone S_{gc} generated based on S_{tips}. l_{gc} is set as 5. (F) Generated growth cone S_{gc} across the entire neuron.

Because the phase field method solves interface movement based on energy balance, there exist small protrusions that could be incorrectly identified as tips by Algorithm 2. Therefore, we apply a clearing step at the end to filter out tips with small areas based on a tip area threshold γ_{tip} (the default value is 10). Furthermore, because Algorithm 2 is intensity-based with connected component analysis, neurites from the same neuron will not intersect as I will decrease when neurites approach each other, while I on neurites from different neurons will not be affected.

Feature-based tip adjustment (Module E). After determining evolving DIV and detecting potential neurite tips, we use the external cue-guided mechanism and energy activation zones introduced in the neuron growth model [28] to guide neurite growth. During neuron growth, an actin-rich protrusion at the neurite tip called the growth cone is

responsible for exploring the surrounding environment and guiding the neurite growth toward extracellular cues (Figure 5A). The extracellular medium will affect neurite growth because neurites consume various proteins and chemical concentrations during the growth and tend to grow towards extracellular attractive cues with high concentration. With the external cue-guided mechanism, we can specify the direction neurites should grow towards by changing the external cue placement (Figure 5B). Based on the evolving DIV, we first select and implement the corresponding neurite morphometric features ($\bar{\theta}_t, \bar{\tau}, \bar{l}_{seg}$) from Table 2. During the feature-based tip guidance procedure, we can selectively update external cues to guide the growing neurites so that neurite segments between updates satisfy the range set by the 1^{st} and 3^{rd} quartiles (Q_1, Q_3) of $\bar{l}_{seg}$ by monitoring the average segment length l_{seg}. l_{seg} was calculated by dividing l_{total} by the number of neurite tips recorded. To update external cue placement in Module E, we identify neurite tracing S_{tr} based on the neurite tip S_{tips}. Then, we calculate the neurite tortuosity τ by dividing the geodesic distances l_{neu} between the tip and the root by the corresponding Euclidean distance:

$$l_{neu} = max(d_{geo}) - min(d_{geo}), \tag{12}$$

$$\tau = \frac{l_{neu}}{\sqrt{(x_{tip} - x_{root})^2 + (y_{tip} - y_{root})^2}}, \tag{13}$$

where (x_{tip}, y_{tip}) and (x_{root}, y_{root}) are the coordinates of the neurite tip and root, respectively. Then, to implement the average experimental neurite tortuosity $\bar{\tau}$, we repeatedly generate new θ_t based on $\bar{\theta}_t$ to place the external cue, until the calculated neurite tortuosity $\tau \in [\bar{\tau}(Q_1), \bar{\tau}(Q_3)]$. To generate a new θ_t, we first calculate the direction of the neurite using endpoints of S_{tr}. Then, we generate a new θ_t using Gaussian distribution for each S_{tips} based on mean μ and standard deviation σ of experiment $\bar{\theta}_t$:

$$\theta_t(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right). \tag{14}$$

As shown in Figure 5B, the external cue (cyan dot) is then placed near the neurite tip (red dot) based on θ_t, and the tip section S_{tip} closest to the cue is selected as the energy activation zone to simulate the effect of the growth cone. Once all neurite tips are detected, the updated S_{tips} of each neuron are then passed back for growth cone generation in Module B.

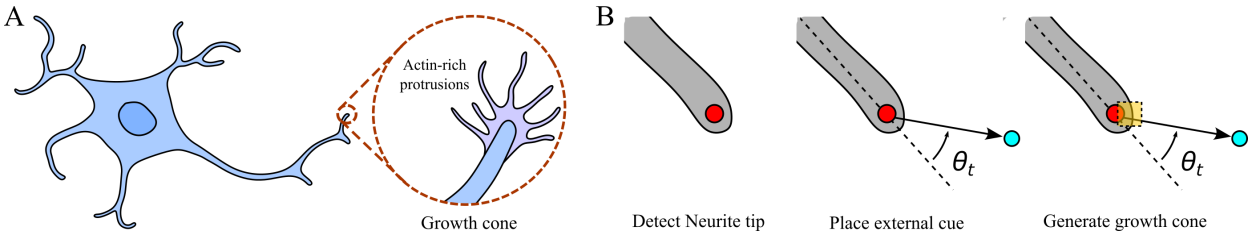

Figure 5: Neurite growth cone and external cue-guided mechanism. (A) The growth cone is an actin-rich protruding area at the neurite tip. (B) External cue-guided mechanism. The neurite tip is first detected, then an external cue (cyan dot) is placed based on θ_t, and finally a growth cone (orange square zone) is selected close to the external cue.

3.3. Neurite features driven neuron growth results

In this section, we study single- and multiple-neuron growth scenarios using the proposed feature-driven neuron growth model and summarize improvements based on a comparison with experimental neuron cultures and our previous neuron growth model [28]. Our feature-driven neuron growth IGA-C solver is developed using MATLAB (R2022a) on a desktop with 12 cores and 64GB memory. The batch computation used Bridges-2 Supercomputer at Pittsburgh Supercomputer Center through Advanced Cyberinfrastructure Coordination Ecosystem (ACCESS) [54, 55].

To compare the feature-driven IGA-C model with our previous neuron growth model, we set up scenarios for 1-7 neurons. In Figure 6A, we first show a selection of neuron culture images with different numbers of neurons as a reference. In Figure 6B-D, we compare our previous neuron growth model (top row) and our feature-driven model

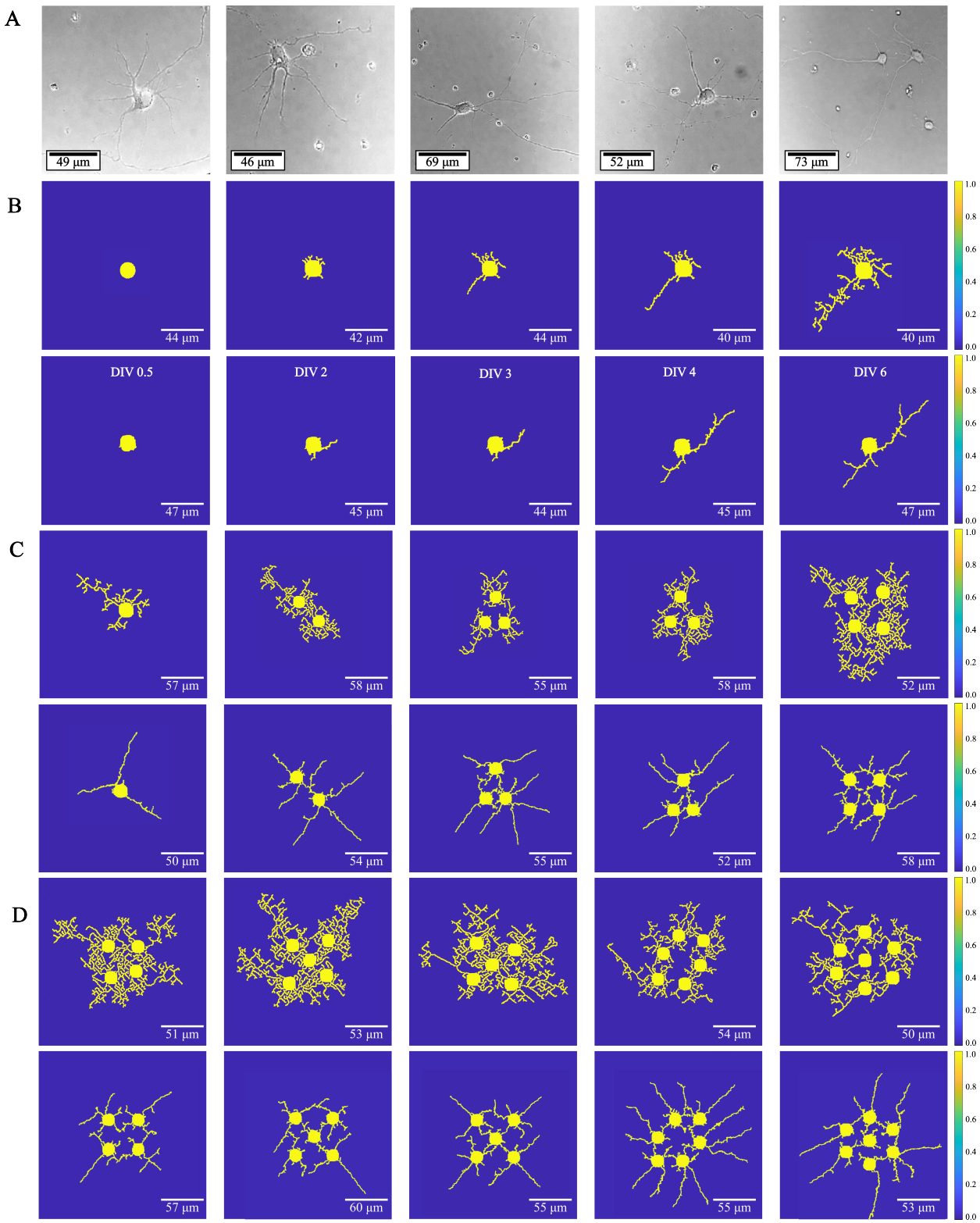

Figure 6: Neurite morphometric features-driven neuron growth results with intrinsic growth stage transition. (A) Experimental neuron growth culture images of rat hippocampal neurons. (B) Neuron growth simulation results at each DIV using our previous neuron growth model [28] (top row) and feature-driven neuron growth model (bottom row). (C-D) Comparison of both models for 1 to 7-neuron scenarios.

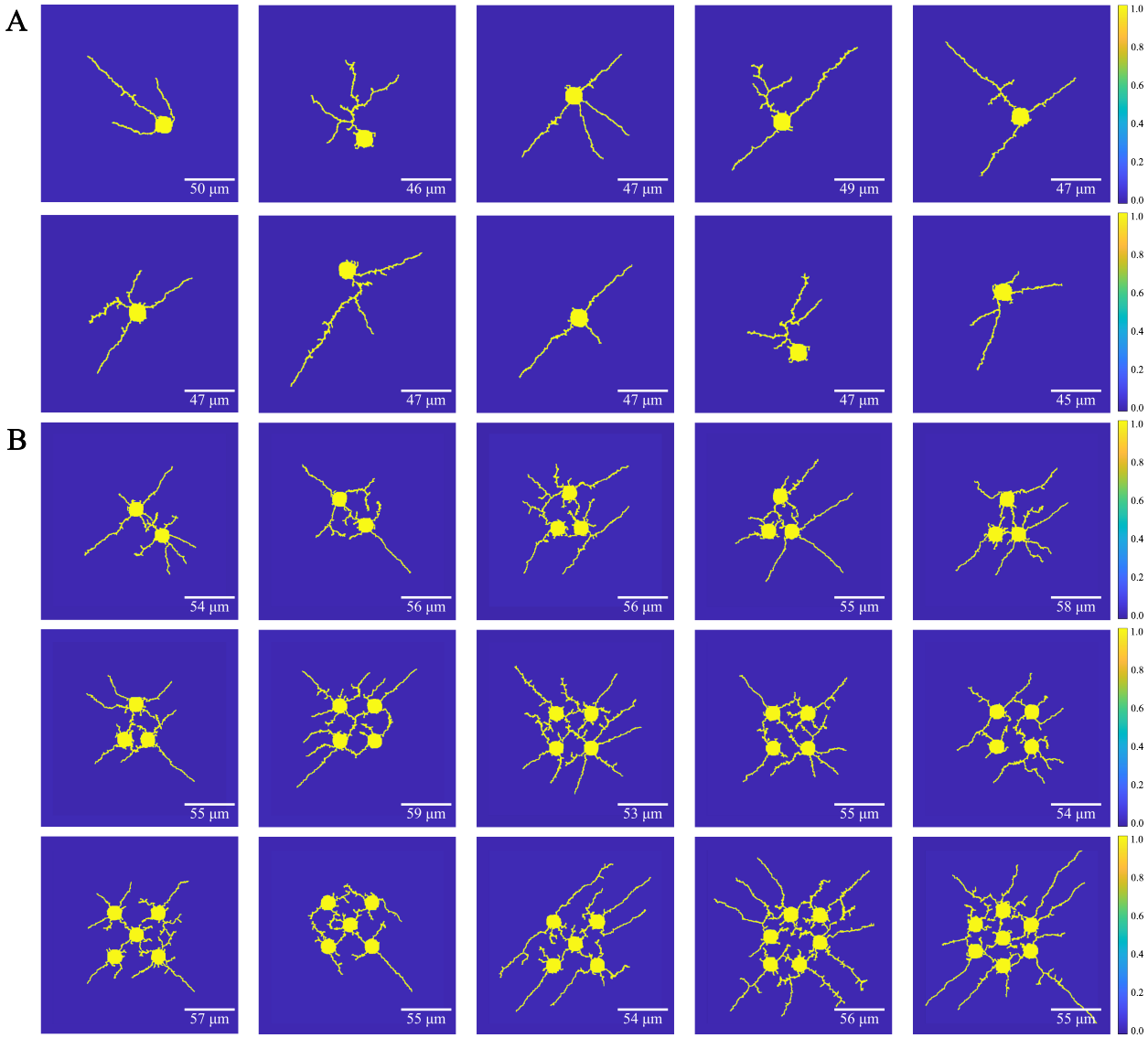

Figure 7: Feature-driven neuron growth model simulation results using a grid size of 400x400. (A) DIV 6 single neuron growth simulation results with neurite morphometric features implementations. (B) Multiple neuron scenarios (2-7 neurons) simulation results.

(bottom row) on a single neuron over the growth process from DIV 0.5 to DIV 6. It is obvious that our previous neuron growth shows unnecessary branching at a later stage not observed in experimental images (Figure 6A), while our feature-driven neuron growth model results exhibit less neurite branching with much fewer neurite endpoints. This behavior illustrates the effect of $\bar{n}_e$, which constrains the number of neurite endpoints throughout the growth. Our feature-driven model shows relatively straight neurites through each DIV. This indicates the effect of tortuosity $\bar{\tau}$, which continuously guides the neurite to grow in a relatively straight path through the external-cue-guided mechanism. To simulate multiple neurons (2-7 neurons) as shown in Figure 6C&D, we leverage MATLAB's connected component analysis to iterate through each neuron S_{neu} within ϕ and apply Modules B-E accordingly. For each S_{neu}, individual S_{tips} and M_ϕ are calculated and then combined together with other neurons at the end of the iteration to drive ϕ interface change in Module A.

In Figure 7, we present more results using the feature-driven neuron growth model. All neuron growth simulation results ranging from 1 to 7 neurons show similar morphology compared to corresponding experimental neuron cultures. Among single neuron scenarios (Figure 7A), most neurites attain straight neurite behaviors and experience

oscillating small bending behaviors due to $\bar{\tau}$ implementations. Neurite branching behaviors are also observed as neurites grow further out. Some cases develop all three generations of neurites (primary, secondary, and tertiary). Both of these features are representative of increasing $\bar{n}_e$ at later DIV. For a quantitative comparison, we apply the semi-automated neurite morphometric evaluation to single neuron cases shown in Figure 7A and compare the corresponding features to experiments in DIV 4; see Table 4. We can observe that our feature-driven neuron growth model generates similar neurite morphology compared to experimentally cultured neurite growth results.

Table 4: Neurite morphometric features statistics comparison.

Dataset	$\bar{\theta}_t(^\circ)$		$\bar{\tau}$		$\bar{n}_e$		$\bar{l}_{seg}(\mu m)$		$\bar{l}_{total}(\mu m)$
	μ	σ	Q_1	Q_3	Q_1	Q_3	Q_1	Q_3	
Experiments	21.28	4.91	1.0341	1.0623	3	7	6.65	10.86	218.74
Simulations	25.56	5.97	1.0437	1.0947	5	6	6.72	7.134	202.01

4. CNN-based prediction of the neuron growth process

In this section, we demonstrate a convolutional neural network (CNN) based surrogate model to predict neuron growth scenarios. We first introduce our CNN auto-encoder implementation with multiple layers. Then, we explain our data generation procedures. Finally, we present our model prediction results compared to the IGA-C solver.

4.1. CNN auto-encoder model

We propose to use CNN with auto-encoder architecture to predict neuron growth. CNN is a subset of machine learning techniques proposed to operate on tensor data based on convolution operations. To achieve efficient and accurate prediction of the neurite growth pattern, we adopt a light convolutional autoencoder [32] as the backbone of our surrogate model, consisting of a multi-layer encoding and decoding architecture. The 3-channel input tensor consisting of phase field ϕ, orientation θ, and target iteration *iter* goes through a 3-layer encoding process and then a 4-layer decoding process with a Sigmoid activation function at the output to predict neuron growth (Figure 8).

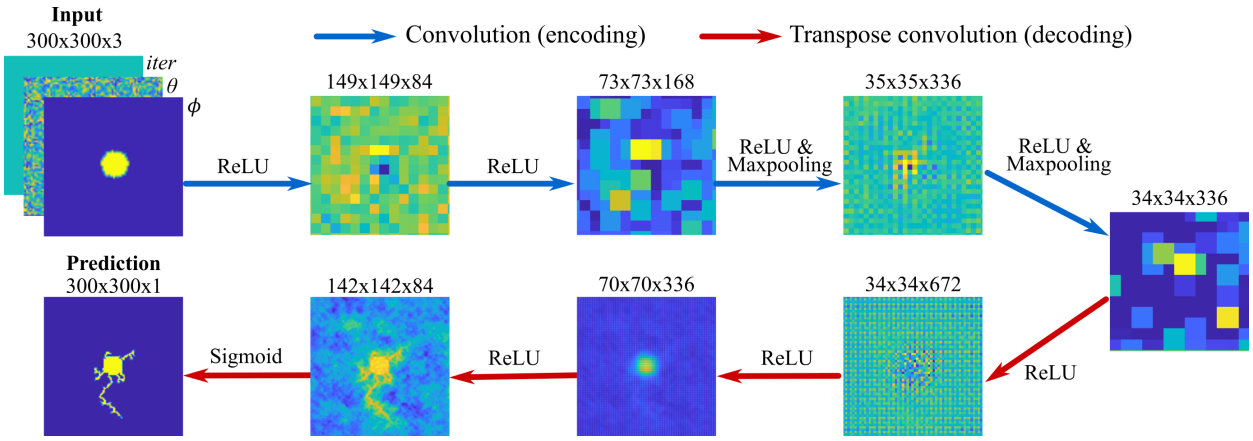

Figure 8: Convolutional neural network (CNN) with the auto-encoder architecture for neuron growth prediction.

Given a specific iteration, the model takes the neuron initializations as input and outputs the pattern of the grown neurite based on the target iteration number in the input tensors. The input includes the initial phase field ϕ, the orientation θ, and the target iteration number *iter*, where ϕ is a 300×300 matrix representing the initialized pattern of the neuron cell, and θ is randomly initialized between [0,1] for each neuron growth case. The model builds correlations among the input channels by extracting cross-channel features using convolution layers and rectified linear unit (ReLU) activation functions. The encoder reduces the spatial dimensionality of the input tensor and increases the number of channels (Figure 8). The decoder correlates and upsamples abstract features from the low-dimension representation to generate a 300×300 neuron growth prediction (ϕ). During the encoding process, we also implement

max-pooling layers to help prevent overfitting by only considering the element with the maximum value. As for the activation function, we choose ReLU because ReLU has shown similar accuracy while requiring less computational time compared to other widely used activation functions [32]. ReLU pushes negative values within the tensor to 0 while not affecting positive values. Because the phase field ϕ is binary, we add the Sigmoid activation function at the end to scale and reduce CNN model prediction to 0 and 1.

During the training process, we use binary cross-entropy (BCE) to calculate the loss gradients for back-propagation. Because the phase field ϕ is binary, we can treat the neuron growth problem as a binary classification problem in a 2D matrix for simplicity. BCE is ideally suited for this study since BCE loss computes the degree of divergence between binary predictions and the ground truth. Although the commonly used mean square error (MSE) loss function allows the model to make similar neurite growth pattern predictions, it cannot correctly generate predictions with magnitudes between 0 and 1. BCE is defined as:

$$BCE = -\frac{1}{N}\sum_{1}^{N}[y_i \times log(P(y_i)) + (1 - y_i) \times \log(P(1 - y_i))], \tag{15}$$

where y_i is the binary label, and $P(y_i)$ is the probability of the entity being y_i for all N entities. We train the model with a learning rate α of 1e-4 for 450 epochs. BCE enables us to treat the problem as a binary classification problem but does not provide a suitable error representation of the 2D domain. Thus, we use the mean magnitude of the relative error (MRE)) on the test dataset to better evaluate model performance and visualize relative errors of the 2D binary field (ϕ) [56]. The MRE is defined as:

$$MRE = \frac{\sqrt{\frac{1}{n}\sum_{1}^{n}(\phi_{gt} - \phi_{pred})^2}}{\phi_{gt}^{max} - \phi_{gt}^{min}} \times 100\%, \tag{16}$$

where n is the number of pixels in one matrix, ϕ_{gt} is the ground truth, and ϕ_{pred} is the prediction. ϕ_{gt}^{max} and ϕ_{gt}^{min} are the maximum and minimum values of the ground truth. From the generated dataset, each neuron case has three input channels: ϕ with neuron cells initialized near the center of the domain, orientation θ initialized to obtain evolving growth patterns, and target iteration *iter*s.

4.2. Data generation

Using the IGA-C-based phase field neuron growth model with incorporated neurite features, we run a batch of neuron growth cases with different neuron placements. For each case, we initialize neuron cells as solid circles inside the domain and randomly initialize an orientation θ. For multiple neuron growth cases with complex neurite network formations, we initialize multiple neuron cells in the domain for the solver to develop neurite interactions among neurons. We consider seven types of neuron scenarios in our model, from 1 to 7 neurons, for generating the training dataset. In each case, we initialize the domain as a 60×60 quadrilateral control mesh with neuron cell initialization and a corresponding random variable θ of the same size to introduce different growth behaviors into the model. The domain adaptively expands as the neurite growth approaches the boundary. We run the solver until DIV 6 is reached for each case with a time step $\triangle t = 0.05$. Simulation results at each iteration are extracted and stored based on collocation points in the domain. Because we adaptively expand the domain during the simulation, the final mesh size varies case by case. We downsample the domain into 300×300 to fit the dataset into our convolutional neural network. Then, to lower memory requirements and remove redundant information, we extract 60 data frames per case. Our CNN model takes the initial phase field ϕ, case-specific randomly initialized orientation θ, and target iteration *iter* as the input and outputs predicted ϕ at target *iter*. Thus, we extract and assemble these three variables from each data point to obtain a dataset of size $(105, 60, 300, 300, 3)$. Before training, the dataset is shuffled and split 75/25 as training and testing dataset, respectively.

4.3. Single and multiple neuron CNN predictions

In this section, we present predictions of single and multiple neuron cases using our CNN surrogate model. After setting up our model architecture using Pytorch [57], we use Tesla V100-16 on Bridge-2 Supercompter at Pittsburgh Supercomputer Center (PSC) to train the model on 75% of randomized neuron dataset for 450 epochs. The model

reaches an average MRE of 2.56% using our previous neuron growth dataset and 2.23% using the feature-driven neuron growth dataset.

Using our CNN-based model, we can predict the growth of single neuron cases and calculate the absolute error compared to corresponding IGA-C solver results (Figure 9A&B). Each neuron prediction closely resembles neurite growth patterns shown in ground truth with complex branching and axon differentiation. We also observe that the highest error occurs at the neurite tips. As the *iter* number increases, more neurite tips form due to branching. Consequently, the mean MRE of single neuron predictions goes up to 2.87% in the cases shown. We then study multi-neuron scenarios (2-7 neurons) with complex neurite networks using our CNN-based model and compare the results with IGA-C results (Figure 9C&D). In each case shown, we observe that more neurons form more complex neurite networks with the largest mean MRE of 4.99% and that these results show that our model can reproduce similar neurite growth patterns to the IGA-C solver with high accuracy.

To better visualize our model performance, we plot the MRE in a scatter plot with the average MRE as a red line (see Figure 10A&C). We observe that model prediction error climbs as *iter* increases when complex neurite patterns form for all cases. The maximum MRE appears on multi-neuron cases at later *iter*, particularly for ones with complex neurite network formations. We record a maximum MRE of 19.48% and 7.60% for our previous neuron growth model and the proposed feature-driven neuron growth model, respectively. We select four worst-case predictions based on MRE for both models (see Figure 10B&D) and observe that they exhibit similar incorrect neurite patterns in areas far away from the center, indicating a lack of information during training in these areas. This finding is within expectation because our neuron growth problem is set to grow from the center to the edge of the domain that is adaptively expanded during the simulation. Therefore, no matter how large our dataset is, there will always be a lack of information near the boundary. Consequently, our end-to-end CNN-based model may be limited when facing complex neurite networks near the domain boundary. We also observe that the computation time of our CNN prediction (approximately $0.25s$ per prediction) is vastly faster than that of the IGA-C neuron growth solver (approximately 240 hours on average). Therefore, the CNN-based model can predict complex neuron growth patterns up to 7 orders of magnitude times faster than the IGA-C neuron growth solver.

5. Conclusion and future work

This paper demonstrates a new computational pipeline to incorporate experimental neurite morphometric features into an IGA-C phase field model to simulate biomimetic intrinsic multi-stage neuron growth behaviors. Furthermore, we demonstrate that a CNN-based surrogate model can significantly reduce the associated computational cost for neuron growth predictions. Based on the results presented in this paper, we conclude that:

- We have incorporated experimentally observed neurite morphometric features across 6 DIV into the IGA-C neuron growth model to drive and constrain the neuron growth process. The model generates biomimetic multiple-neuron scenarios with neurite interactions statistically comparable to experimental results.
- The feature-driven neuron growth model is adaptable to different neuron growth behaviors by switching to a different set of neurite morphometric features. In this paper, we utilize rat hippocampal neuron data.
- The CNN surrogate model can predict the multi-stage neuron growth process. Our model can learn from an abstract representation of neuron growth data and generate end-to-end accurate growth predictions at a given iteration. Our proposed model accurately predicts all growth stages ($<$ 2.23% error) while taking 7 orders of magnitude less computational times compared with our IGA-C neuron growth solver.

The CNN model significantly broadens future research possibilities due to its fast and accurate prediction capability. Our model enables researchers to visualize neuron growth results ahead of time without going through expensive neuron growth experimental trial and error to achieve desirable results. In particular, the proposed machine learning model will supplement traditional neuron culture experiments by enabling researchers to test and predict the effects of hypothetical experimental conditions before implementing them *in vitro*. The proposed computational pipeline will allow researchers to plan experiments more efficiently and reduce experimental costs significantly by generating large amounts of predictions based on given initial conditions.

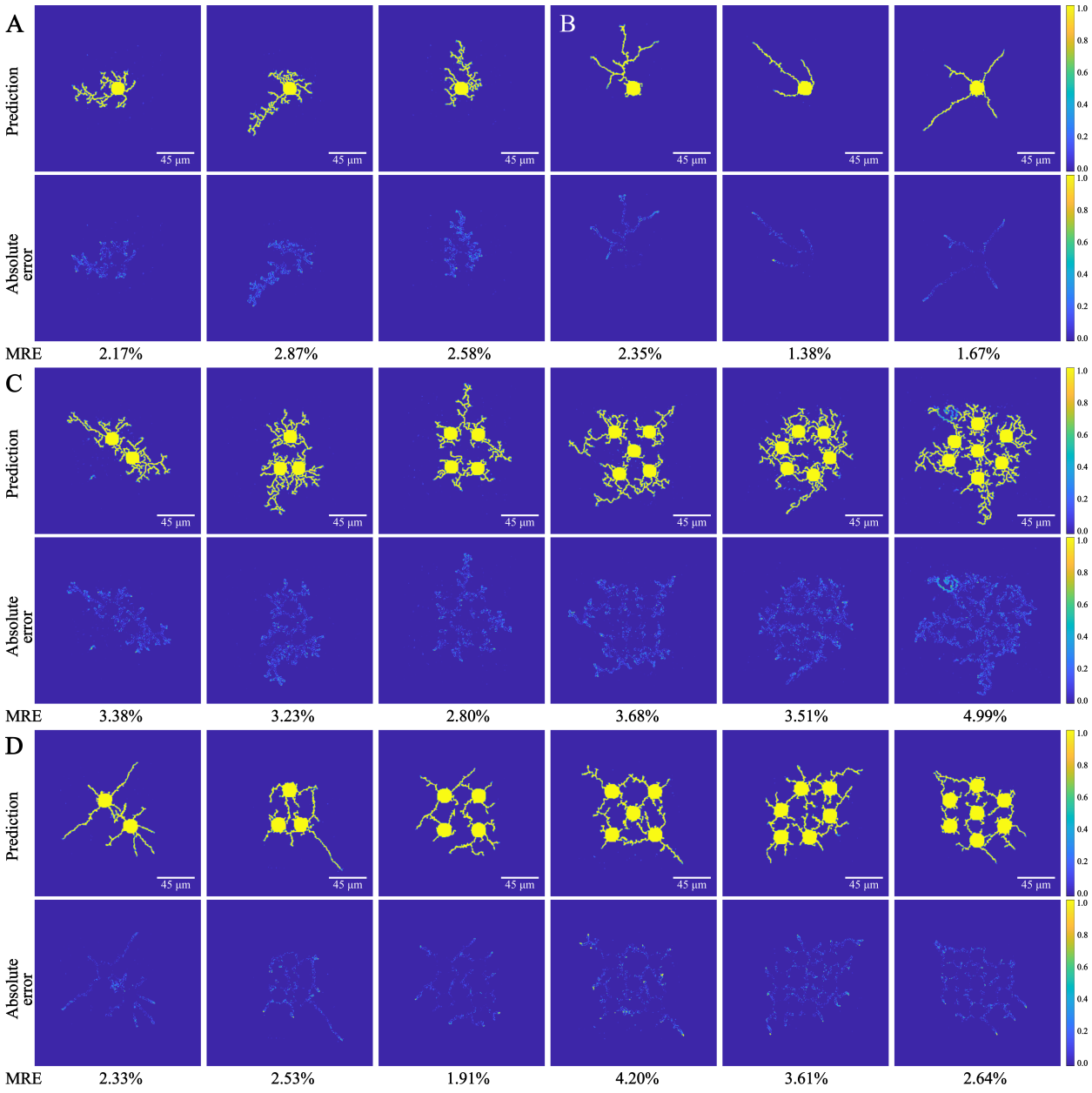

Figure 9: CNN-based model prediction results of single neuron cases using two neuron growth models. (A, C) Single and multiple neuron growth prediction using our previous neuron growth model [28]. (B, D) Single and multiple neuron growth prediction using feature-driven neuron growth model. The top row is the CNN prediction, and the bottom row is the corresponding absolute error. These results show that our CNN model can reproduce similar neurite growth patterns to the IGA-C solver with high accuracy (MRE < 5%).

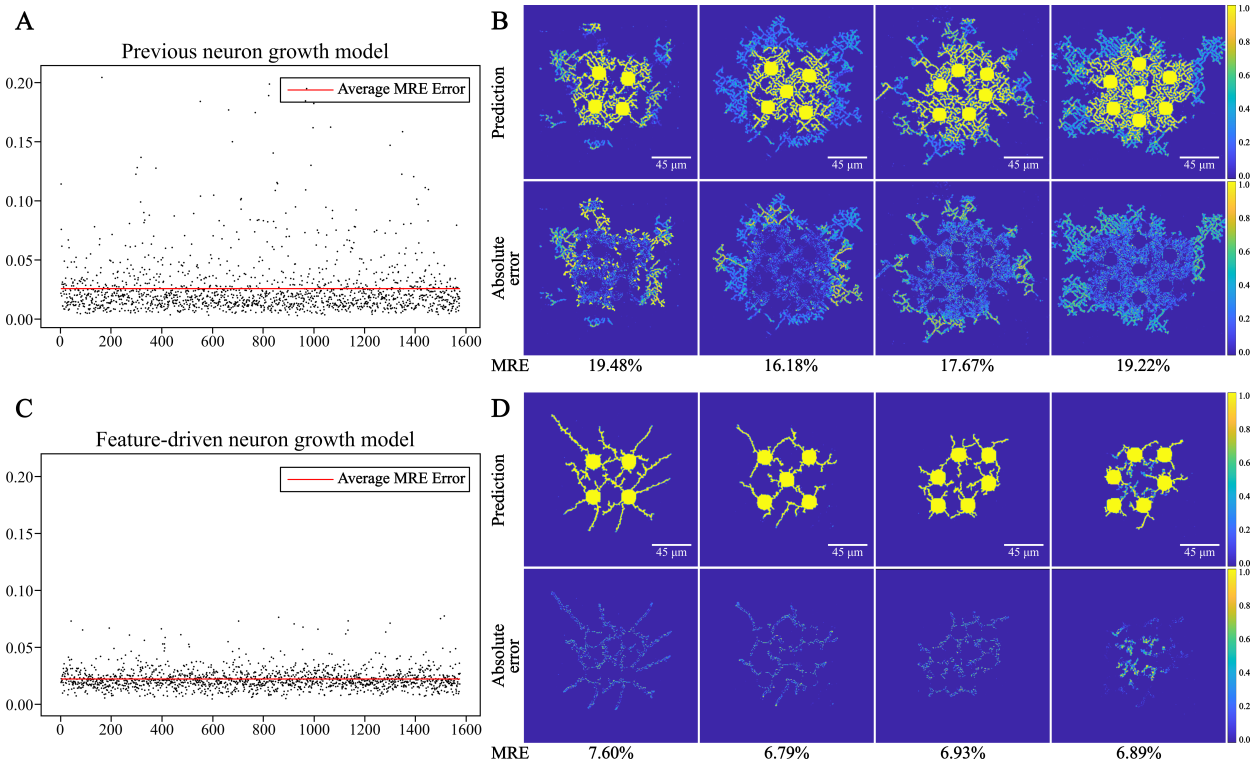

Figure 10: CNN-based model prediction error. (A, C) CNN-based model accuracy statistics of our previous and feature-driven neuron growth models using the test dataset. MRE of all test datasets are plotted as scatter points, and the average MRE is plotted as the red line. (B, D) Four worst CNN predictions of both models.

Whereas the model presented in this work substantially improves neuron growth modeling accuracy and speed, some limitations point to interesting future research directions. Our current features-driven IGA-C neuron growth model does not consider experimentally-measured growth rates of axons and dendrites [10]. Our growth model leverages the fixed mobility term difference to differentiate the growth rates of dendrites and axons. In the future, this can be improved by incorporating experimentally-measured growth rates. Another limitation is the lack of informative neuron growth factors, such as extracellular and intracellular concentrations, crucial for understanding the biophysics process behind the scenes. This information in future models may help reveal critical and previously unknown relationships between neuron growth and neurodegenerative diseases. In the future, we can combine our model with material transport in neurons to explore complex biophysics-coupled neurite morphologies [58, 59]. As shown in CNN predictions, our model lacks the accuracy of modeling neurite growth far away from the center of the soma, where information degrades. To better understand potential neurophysiological implications, we will carry out an uncertainty quantification study of our growth results. Finally, our feature-driven neuron growth model is limited to 2D using hierarchical B-splines. We plan to extend our neuron growth model to 3D and implement truncated hierarchical B-splines (THB-spline) [60, 61]. Note that our current neuron growth code is implemented in MATLAB, we plan to convert it into C/C++ to improve computational speed. Furthermore, the surrogate CNN model is trained based solely on simulation data without any information from the phase field governing equations. The residuals of the governing equations could be considered during the model training process using physics-informed loss functions [31, 33]. A generalized surrogate model can be trained and adjusted to different types of neuron growth using the transfer learning approach [62]. Along the deep learning direction, we are also exploring more efficient and effective model architectures such as recurrent neural networks (RNN) and transformers [63] to better understand temporal changes during the neuron growth process [64].

Code and data availability

The code and datasets generated and analyzed in this paper are accessible in the "FDNGCNN" GitHub repository. `https://github.com/CMU-CBML/FDNGCNN` (DOI:10.5281/zenodo.7853871). Correspondence and requests for code and data should be addressed to K.Q. or Y.J.Z.

Declaration of competing interest

The authors declare no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

K. Qian and Y. J. Zhang were supported in part by the NSF Grant CMMI-1953323. All of the authors were supported in part by the PITA (Pennsylvania Infrastructure Technology Alliance) and PAMIP (Pennsylvania Manufacturing Innovation Program) grants. K. Qian was also supported by Bradford and Diane Smith Graduate Fellowship. In addition, A. S. Liao and V. A. Webster-Wood were supported in part by an NSF CAREER award ECCS-2044785. A. S. Liao was also supported by NSF Graduate Research Fellowship Grant DGE-1745016 and Carnegie Mellon University Jean-Francois and Catherine Heitz Scholarship. This work used RM-node and GPU-node on Bridges-2 Supercomputer at Pittsburgh Supercomputer Center [54, 55] through allocation ID eng170006p from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

References

[1] B. N. Dugger, D. W. Dickson, Pathology of neurodegenerative diseases, Cold Spring Harbor Perspectives in Biology 9 (2016) a028035.
[2] R. C. Brown, A. H. Lockwood, B. R. Sonawane, Neurodegenerative diseases: an overview of environmental risk factors, Environmental Health Perspectives 113 (9) (2005) 1250–1256.
[3] H. Checkoway, J. I. Lundin, S. N. Kelada, Neurodegenerative diseases., IARC Scientific Publications (163) (2011) 407–419.
[4] D. K. Berg, New neuronal growth factors, Annual Review of Neuroscience 7 (1984) 149–170.
[5] B. Connor, M. Dragunow, The role of neuronal growth factors in neurodegenerative disorders of the human brain, Brain Research Reviews 27 (1) (1998) 1–39.
[6] M. M. Reimer, I. Sörensen, V. Kuscha, R. E. Frank, C. Liu, C. G. Becker, T. Becker, Motor neuron regeneration in adult zebrafish, Journal of Neuroscience 28 (34) (2008) 8510–8516.
[7] M. M. Steward, A. Sridhar, J. S. Meyer, Neural regeneration, New Perspectives in Regeneration (2013) 163–191.
[8] J. L. Elliott, W. D. Snider, Motor neuron growth factors, Neurology 47 (4 Suppl 2) (1996) 47S–53S.
[9] A. van Ooyen, Modeling Neural Development, MIT Press, 2003.
[10] C. G. Dotti, C. A. Sullivan, G. A. Banker, The establishment of polarity by hippocampal neurons in culture, Journal of Neuroscience 8 (4) (1988) 1454–1468.
[11] H. G. E. Hentschel, D. Samuels, A. Fine, Instabilities during the dendritic and axonal development of neuronal form, Physica A: Statistical Mechanics and Its Applications 254 (1-2) (1998) 46–61.
[12] A. Liao, W. Cui, Y. J. Zhang, V. Webster-Wood, Quantitative evaluation of neuron developmental morphology *in vitro* using the change-point test, Summer Biomechanics, Bioengineering and Biotransport Conference (2021).
[13] H. G. E. Hentschel, A. Fine, Instabilities in cellular dendritic morphogenesis, Physical Review Letters 73 (1994) 3592–3595.
[14] J. K. Krottje, A. Van Ooyen, A mathematical framework for modeling axon guidance, Bulletin of Mathematical Biology 69 (2007) 3–31.
[15] Y. E. Pearson, E. Castronovo, T. A. Lindsley, D. A. Drew, Mathematical modeling of axonal formation Part I: geometry, Bulletin of Mathematical Biology 73 (2011) 2837–2864.
[16] M. Aeschlimann, L. Tettoni, Biophysical model of axonal pathfinding, Neurocomputing 38-40 (2001) 87–92.
[17] J. P. Eberhard, A. Wanner, G. Wittum, NeuGen: a tool for the generation of realistic morphology of cortical neurons and neural networks in 3D, Neurocomputing 70 (1-3) (2006) 327–342.
[18] A. van Ooyen, A. Carnell, S. de Ridder, B. Tarigan, H. D. Mansvelder, F. Bijma, M. de Gunst, J. van Pelt, Independently outgrowing neurons and geometry-based synapse formation produce networks with realistic synaptic connectivity, PloS ONE 9 (1) (2014) e85858.
[19] G. J. Goodhill, M. Gu, J. S. Urbach, Predicting axonal response to molecular gradients with a computational model of filopodial dynamics, Neural Computation 16 (11) (2004) 2221–2243.
[20] S. M. Maskery, H. M. Buettner, T. Shinbrot, Growth cone pathfinding: a competition between deterministic and stochastic events, BMC Neuroscience 5 (22) (2004).
[21] R. A. Koene, B. Tijms, P. Van Hees, F. Postma, A. De Ridder, G. J. Ramakers, J. Van Pelt, A. Van Ooyen, NETMORPH: a framework for the stochastic generation of large scale neuronal networks with realistic neuron morphologies, Neuroinformatics 7 (2009) 195–210.

[22] H. Cuntz, F. Forstner, A. Borst, M. Häusser, One rule to grow them all: a general theory of neuronal branching and its practical application, PLoS Computational Biology 6 (8) (2010) e1000877.
[23] D. E. Donohue, G. A. Ascoli, A comparative computer simulation of dendritic morphology, PLoS Computational Biology 4 (6) (2008) e1000089.
[24] B. Torben-Nielsen, E. De Schutter, Context-aware modeling of neuronal morphologies, Frontiers in Neuroanatomy 8 (2014) 92.
[25] M. O'Toole, P. Lamoureux, K. E. Miller, A physical model of axonal elongation: force, viscosity, and adhesions govern the mode of outgrowth, Biophysical Journal 94 (7) (2008) 2610–2620.
[26] B. P. Graham, A. Van Ooyen, Mathematical modelling and numerical simulation of the morphological development of neurons, BMC Neuroscience 7 (2006) S9.
[27] T. Takaki, K. Nakagawa, Y. Morita, E. Nakamachi, Phase-field modeling for axonal extension of nerve cells, Mechanical Engineering Journal 2 (3) (2015) 15–00063.
[28] K. Qian, A. Pawar, A. Liao, C. Anitescu, V. Webster-Wood, A. W. Feinberg, T. Rabczuk, Y. J. Zhang, Modeling neuron growth using isogeometric collocation based phase field method, Scientific Reports 12 (2022) 8120.
[29] A. S. Liao, W. Cui, Y. J. Zhang, V. A. Webster-Wood, Semi-automated quantitative evaluation of neuron developmental morphology *in vitro* using the change-point test, Neuroinformatics 21 (2022) 163–176.
[30] J. Han, A. Jentzen, W. E, Solving high-dimensional partial differential equations using deep learning, Proceedings of the National Academy of Sciences 115 (34) (2018) 8505–8510.
[31] M. Raissi, P. Perdikaris, 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 (2019) 686–707.
[32] A. Li, R. Chen, A. B. Farimani, Y. J. Zhang, Reaction diffusion system prediction based on convolutional neural network, Scientific Reports 10 (2020) 3894.
[33] A. Li, A. Barati Farimani, Y. J. Zhang, Deep learning of material transport in complex neurite networks, Scientific Reports 11 (2021) 11280.
[34] A. Li, Y. J. Zhang, Isogeometric analysis-based physics-informed graph neural network for studying traffic jam in neurons, Computer Methods in Applied Mechanics and Engineering 403 (2023) 115757.
[35] T. J. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Computer Methods in Applied Mechanics and Engineering 194 (39-41) (2005) 4135–4195.
[36] T. Takaki, Phase-field modeling and simulations of dendrite growth, ISIJ International 54 (2) (2014) 437–444.
[37] R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, Physica D: Nonlinear Phenomena 63 (3-4) (1993) 410–423.
[38] H. Song, G. Ming, M. Poo, cAMP-induced switching in turning direction of nerve growth cones, Nature 388 (6639) (1997) 275–279.
[39] A. van Ooyen, B. P. Graham, G. J. A. Ramakers, Competition for tubulin between growing neurites during development, Neurocomputing 38-40 (2001) 73–78.
[40] B. Ren, J. Huang, M. C. Lin, S.-M. Hu, Controllable dendritic crystal simulation using orientation field, Computer Graphics Forum 37 (2) (2018) 485–495.
[41] H. Gomez, A. Reali, G. Sangalli, Accurate, efficient, and (iso) geometrically flexible collocation methods for phase-field models, Journal of Computational Physics 262 (2014) 153–171.
[42] D. Schillinger, M. J. Borden, H. K. Stolarski, Isogeometric collocation for phase-field fracture models, Computer Methods in Applied Mechanics and Engineering 284 (2015) 583–610.
[43] Y. Zhang, Challenges and advances in image-based geometric modeling and mesh generation, in: Image-Based Geometric Modeling and Mesh Generation, Springer, 2013.
[44] Y. J. Zhang, Geometric Modeling and Mesh Generation from Scanned Images, Chapman and Hall/CRC, 2016.
[45] F. Auricchio, L. B. Da Veiga, T. Hughes, A. Reali, G. Sangalli, Isogeometric collocation methods, Mathematical Models and Methods in Applied Sciences 20 (11) (2010) 2075–2107.
[46] H. Casquero, L. Liu, Y. J. Zhang, A. Reali, H. Gomez, Isogeometric collocation using analysis-suitable T-splines of arbitrary degree, Computer Methods in Applied Mechanics and Engineering 301 (2016) 164–186.
[47] L. Piegl, W. Tiller, The NURBS Book, Springer Science & Business Media, 1996.
[48] X. Wei, Y. J. Zhang, L. Liu, T. J. Hughes, Truncated T-splines: Fundamentals and Methods, Computer Methods in Applied Mechanics and Engineering 316 (2017) 349–372.
[49] Y. Jia, C. Anitescu, Y. J. Zhang, T. Rabczuk, An adaptive isogeometric analysis collocation method with a recovery-based error estimator, Computer Methods in Applied Mechanics and Engineering 345 (2019) 52–74.
[50] C. Anitescu, Y. Jia, Y. J. Zhang, T. Rabczuk, An isogeometric collocation method using superconvergent points, Computer Methods in Applied Mechanics and Engineering 284 (2015) 1073–1097.
[51] H. Casquero, L. Liu, C. Bona-Casas, Y. J. Zhang, H. Gomez, A hybrid variational-collocation immersed method for fluid-structure interaction using unstructured T-splines, International Journal for Numerical Methods in Engineering 105 (11) (2016) 855–880.
[52] R. W. Johnson, Higher order B-spline collocation at the Greville abscissae, Applied Numerical Mathematics 52 (1) (2005) 63–75.
[53] Thermo Fisher Scientific, B-27 Plus Neuronal Culture System, `https://assets.thermofisher.com/TFS-Assets/LSG/manuals/MAN0017319_B27_PlusNeuronalCultureSystem_UG.pdf` (2018).
[54] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, N. Wilkins-Diehr, XSEDE: accelerating scientific discovery, Computing in Science & Engineering 16 (5) (2014) 62–74.
[55] N. Wilkins-Diehr, S. Sanielevici, J. Alameda, J. Cazes, L. Crosby, M. Pierce, R. Roskies, An overview of the XSEDE extended collaborative support program, in: High Performance Computer Applications - 6th International Conference, ISUM 2015, Vol. 595 of Communications in Computer and Information Science, 2016, pp. 3–13.
[56] M. Jørgensen, T. Halkjelsvik, K. Liestøl, When should we (not) use the mean magnitude of relative error (MMRE) as an error measure in software development effort estimation?, Information and Software Technology 143 (2022) 106784.
[57] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-

performance deep learning library, in: Advances in Neural Information Processing Systems 32, 2019, pp. 8024–8035.
[58] A. Li, Y. J. Zhang, Modeling material transport regulation and traffic jam in neurons using PDE-constrained optimization, Scientific Reports 12 (2022) 3902.
[59] A. Li, Y. J. Zhang, Modeling intracellular transport and traffic jam in 3D neurons using PDE-constrained optimization, Journal of Mechanics 38 (2022) 44–59.
[60] A. Pawar, Y. J. Zhang, C. Anitescu, Y. Jia, T. Rabczuk, DTHB3D_Reg: Dynamic truncated hierarchical B-spline based 3D nonrigid image registration, Communications in Computational Physics 23 (3) (2018) 877–898.
[61] X. Wei, Y. Zhang, T. J. Hughes, M. A. Scott, Truncated hierarchical Catmull–Clark subdivision with local refinement, Computer Methods in Applied Mechanics and Engineering 291 (2015) 1–20.
[62] S. Goswami, C. Anitescu, S. Chakraborty, T. Rabczuk, Transfer learning enhanced physics informed neural network for phase-field modeling of fracture, Theoretical and Applied Fracture Mechanics 106 (2020) 102447.
[63] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. U. Kaiser, I. Polosukhin, Attention is all you need, Advances in Neural Information Processing Systems 30 (2017).
[64] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-C. Woo, Convolutional LSTM network: a machine learning approach for precipitation nowcasting, Advances in Neural Information Processing Systems 28 (2015).