# Deep convolutional and recurrent neural networks for cell motility discrimination and prediction

Jacob C. Kimmel, Andrew S. Brack, and Wallace F. Marshall

***Abstract*—Cells in culture display diverse motility behaviors that may reflect differences in cell state and function, providing motivation to discriminate between different motility behaviors. Current methods to do so rely upon manual feature engineering. However, the types of features necessary to distinguish between motility behaviors can vary greatly depending on the biological context, and it is not always clear which features may be most predictive in each setting for distinguishing particular cell types or disease states. Convolutional neural networks (CNNs) are machine learning models ideally suited to the analysis of spatial data, allowing for relevant spatial features to be learned as parameters of a model. Given that motility data is inherently spatial, we apply CNNs to classify different motility behaviors using two novel approaches. The first approach represents motility explicitly as a 3D space with markers denoting a cell's location at each time point, and the second utilizes recurrent long-term short-term memory (LSTM) units to represent the temporal dimension implicitly. Both 3D CNNs and convolutional-recurrent neural networks (RNNs) provide accurate classification of simulated motility behaviors, the experimentally measured motility behaviors of multiple cell types, and characteristic motility behaviors of muscle stem cell differentiation states. The variety of cell motility differences we can detect suggests that the algorithm is generally applicable to novel cell and sample types. 3D CNN and RNN based autoencoders were also effectively trained using the explicit 3D representations to learn motility features in an unsupervised manner. Additionally, adapted RNN models effectively predict muscle stem cell motility from past tracking data.**

***Index Terms*—convolutional neural network, recurrent neural network, cell motility, cell classification, long-term short-term memory**

## I. Introduction

CELL motility is an emergent property of living matter that spans the nanomolecular and macroscopic length scales, involving a complex regulatory network and dynamic reorganization of the cell's geometry [1], [2]. Cells can display a diverse set of motility behaviors, and these behaviors can provide a useful window for inference of a cell's functional state. Neoplastic transformation has long been appreciated to alter cell motility behaviors, increasing the migration rate of various models in culture and serving as a mechanism for metastasis [3], [4], [5], [6], [7]. The motility behaviors of cancer cells in culture can even be predictive of broader tumor progression [8].

Likewise, the migration of progenitor cells is critical in early development and tissue regeneration [9]. Skeletal muscle stem cells (MuSCs) provide an accessible platform to study stem cell motility phenotypes *in vitro* by timelapse imaging. During embryonic development, MuSC precursors must migrate from early stage developmental structures (somites) to their adult location along the edge of muscle fibers in the trunk and limbs [10], [11]. In the adult, motility continues to play a critical role, as MuSCs migrate along muscle fibers *in vivo* to sites of injury to initiate tissue repair [12], [13]. Motility behaviors are heterogeneous between MuSCs and change during stem cell activation [14], [15]. Heterogeneous fitness for regeneration within the MuSC pool is well appreciated [16], and analysis of heterogeneous motility behaviors may provide an additional lens through which to decompose different MuSC phenotypes.

Given the biological importance of motility phenotypes, classification of cells based on motility behaviors has useful applications in research and diagnostics. Similarly, exploration of heterogeneity within the motility behaviors of a cell population may provide biological insights. However, it is often difficult to determine which features of motility behavior will be predictive of a phenotype of interest, or allow for discrimination of heterogeneous behavior. Different phenotype classification tasks and cell populations may require distinct feature sets to extract valuable biological information. A method to algorithmically determine relevant features of cell motility for a given classification or discrimination task is therefore advantageous.

### A. Related Work

To date, a number of tools have been proposed that rely upon a set of handcrafted features to quantify cell motility behaviors, providing some remarkable results [17], [18], [19], [20], [21]. Neural progenitor cells were discriminated by morphology and motility behavior alone [21], and genes that affect motility have been identified solely from timelapse imaging data [17]. We have recently demonstrated that rates of cell state transitions and the ordered or random nature of these transitions may also be inferred from motility alone [15]. These dramatic results demonstrate the potential insights that may be gathered from more extensive analysis of cell motility. However, these methods rely upon engineering of a hand-crafted feature set, and have thus far focused largely on features of speed and directional persistence. It is possible that more complex features may allow for improved discrimination of cell motility phenotypes, but it is difficult to predict what these features may be in each context.

Convolutional neural networks provide an approach to learn relevant features from data, rather than handcrafting features

J. C. Kimmel and W. F. Marshall are with the Department of Biochemistry and Biophysics, University of California San Francisco, California, CA, 94158, USA e-mail: Wallace.Marshall@ucsf.edu

A. S. Brack is with the Department of Orthapedic Surgery, University of California San Francisco, California, CA, 94158 USA e-mail: Andrew.Brack@ucsf.edu

based on a "best guess" of which features are relevant. In the field of computer vision, convolutional neural networks (CNNs) have recently made rapid advancements, demonstrating state-of-the-art performance on a variety of image classification tasks [22], [23], [24], [25]. CNNs utilize a set of parametrized kernels to extract spatial features, allowing distinct feature kernels to be learned for a given classification task [26]. In this way, CNNs are able to learn a "representation" of the problem's feature space. Feature space representations may also be learned in an unsupervised manner by training CNN autoencoder architectures to encode and decode [27], [28]. This approach may be useful for learning relevant motility features where an explicit classification task is not present.

While CNNs are most commonly applied to tasks involving analysis in two-dimensional images at a single time-point, convolution is a natural analytical tool for any input information with spatial dimensions. CNNs have been successfully applied to a diverse set of non-imaging domains, including natural language processing [29], bird song segmentation [30], and EEG recordings [31]. Perhaps most clearly mirroring our challenge of motion classification, CNNs have performed well in the classification of video recordings [32], [33], [34], [35]. These successful implementations have simply extended CNNs to consider three-dimensional images as inputs, where one axis is time. If the spatial nature of cell motility data is represented explicitly as a 3D image, in the same manner used for video classification, CNNs may allow for motility phenotype classification and unsupervised feature learning, without *a priori* definition of handcrafted features.

Deep neural networks have also been extensively applied to the analysis of sequential inputs, such as natural language sentences and biological polymer sequences [29], [36], [37]. While simple 1D CNNs that consider raw sequence inputs can be effective, the introduction of recurrent units such as long-short-term memory (LSTM) units to learn temporal relationships within the input sequence can improve performance and effectively learn long-term dependencies [38]. Cell motility data can be represented as a two-channel, 1D sequence, where each channel contains position values for an axis in physical space. In this representation, 1D CNN models with recurrent units may also allow for motility phenotype analysis without handcrafted features.

We investigated whether CNNs could be effectively applied to the problem of cell motility phenotype classification utilizing either of these two representation schemes. Cell motility is inherently 3D spatial data, where one dimension is time, such that either explicit representation of the temporal dimension in an image, or learning the temporal dimension relationships with a recurrent unit, may allow for effective analysis.

Here, we present *Lanternfish*, a tool to represent motility paths explicity as 3D images or implicity as multi-channel time series, classify different motility behaviors, learn motility features in an unsupervised fashion using deep neural networks, and predict future cell motility from past behavior. *Lanternfish* represents cell motility using a set of positional markers in a 3D volume, with the depth axis representing time, or as a simple multi-channel time series, where time series values are Cartesian coordinates. We demonstrate that standard CNN architectures are sufficient to accurately distinguish experimentally observed cell motility phenotypes represented using either method. Autoencoder architectures based on these models can be trained successfully on motility representations for use as unsupervised feature extractors. Additionally, we show that our RNN model can be adapted to predict cell motility in subsequent frames.

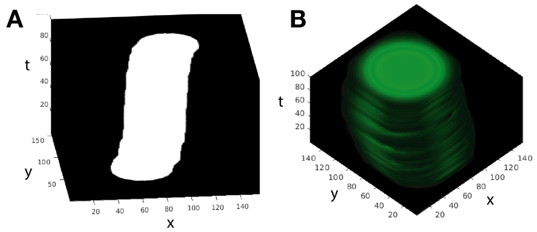

Fig. 1. Representative motility traces using different markers of location. (A) Disk structuring element markers, (B) Gaussian distribution markers.

## II. Methods

All implementations for work presented here are available on Github at https://github.com/jacobkimmel/lanternfish.

### A. Explicit Spatial Representations of Motility Paths

Motion data in a two-dimensional plane is inherently three dimensional, with two dimensions in physical space ($x$ and $y$) and a single time dimension $t$. Each of these dimensions has relevant spatial meaning, and spatial relationships are required to fully represent the motion of an object. This spatial nature makes motion an ideal candidate for the application of convolutional neural networks, which specialize in learning representations of spatial data.

One manner in which motility may be presented for analysis by CNNs is in the form of a static 3D image. Representing time as a spatial axis has allowed for successful time-series analysis by CNNs in multiple other problem domains [32], [33], [34], [35]. Cell motility is typically recorded as the position of the cell centroid at each time point. To represent this time series of positions as a 3D image, we first produce a simple 3D representation of an $(x, y)$ path by placing a 1 pixel (px) binary marker on the location of the object at each time point $t$ in a single slice of a cube with dimensions $(X, Y, T)$, leaving all other values at 0, where $x \in X$, $y \in Y$, and $t \in T$. Viewed one plane at a time along the $t$ dimension, this cube is simply a video of the 2D path representing the object's location with a 1 px marker. However, this trivial representation presents a very sparse feature space, and intuitively may not allow for efficient learning of convolutional kernels.

In expectation of this sparsity problem, we produced tools to build representations that mark an object's location in each $(X, Y, T)$ plane with a binary disk of arbitrary size or Gaussian distribution of arbitrary variance, instead of a single 1 px point. Gaussian distributions are scaled [0, 1] for each $\sigma$ value. The resulting representation resembles a "stack of dinner plates" (Fig. 1). These representations contain information about the objects location at more $(x, y)$ coordinates within a plane than the 1 px representations, so we reasoned that they may aide learning of 3D convolutional kernels.

Further information can be encoded by setting the amplitude of the disk or distribution in each $t$ plane based on some real valued measurement. For instance, instantaneous speed or object size could be encoded as amplitudes. Compression of motion paths may be necessary due to GPU memory constraints. For all experiments performed here, motion paths were compressed 4- or 6-fold in the $(x, y)$ dimensions by simple integer division of $(X, Y)$ coordinates.

### B. Multi-channel Time Series Representation of Motility Paths

Cell motility is also naturally represented as a two-channel time series, where the values of each channel are the Cartesian coordinates $X$ and $Y$. This representation is obviously extensible to the 3D motion case with the simple addition of a third channel for the $Z$ axis. CNNs may be applied with 1D convolutional filters to these multi-channel time series. Multiple problem domains have shown success in applying CNNs to multi-channel time series data in this manner [39], [40]. This approach is inherently simpler than the explicit representation described above. However, 1D representation requires a model to implicitly learn kernels to preserve the relationship between $X$ and $Y$ dimensions in higher layers. We reasoned that there may be cases where this implicit representation is inferior to an explicit representation based on this principle.

### C. 3D Classification architecture

For classification of different types of motion represented as 3D images, we apply a standard CNN architecture utilizing 3D convolutional and max pooling layers, diagrammed in (Fig. 2A). 3D Convolutional layers convolve the 3D motion cube inputs with a set of parametrized kernels, passing the convolutional outputs to the layers above. The max pooling layers perform a $\max$ operation for voxels in an 3D-window, reducing the input size, and returns the resulting output to the layer above. This architecture is similar to well known 2D classification architectures [22], [41]. All convolutional layers are paired with a rectified linear unit (ReLU) activation ($\max(0, x)$) [42], utilize unit strides $s = 1$, and convolve with $(3, 3, 3)$ kernels. Convolutional layers pad input images by 1 px by reflecting edge values to avoid reduction of input size by convolution. Max pooling layers use kernels of size $(2, 2)$ and a corresponding stride of $s = 2$.

Fully connected layers are the same as in a traditional neural network, in which each perceptron unit considers input from all units in the previous layer, and outputs to all units in the next layer [43]. Dropout layers eliminate a random proportion $p$ of fully connected units from a fully connected layer during each forward pass, reducing reliance upon individual units and preventing overfitting [44]. Two fully connected layers with dropout ($p = 0.3$, where $p$ is the proportion of neurons dropped per epoch) and ReLU activations are utilized at the bottom of the network. Final class outputs are returned by a fully connected layer with a number of neurons equal to the number of classes and a *softmax* activation (Fig. 2A).

3D CNN classification networks were trained using stochastic gradient descent with momentum ($\mu = 0.5$). Categorical crossentropy was used as a loss function. We find that training is sensitive to the learning rate $\epsilon$, and thus utilize a low initial learning rate $\epsilon_0 = 0.005$ with a rapid decay function

$$\epsilon_i = \epsilon_0 d^i$$

where $i \in [0, N]$ is the training epoch, $\epsilon_0$ is the initial learning rate, and $d$ is a decay coefficient, set to $d = 0.8$ for our experiments.

### D. 1D Recurrent Classification Architecture

Recurrent classification networks follow a similar standard architecture, utilizing 1D convolutional layers with size 3 kernels at the base followed by a max pooling layer with kernel size 2 and stride $s = 2$. The center of the network contains an LSTM layer with $n = 256$ units. Following the LSTM layer are two fully-connected layers with Dropout (as above) and a final *softmax* classification layer (Fig. 2C).

### E. 3D Autoencoder Architecture

The 3D CNN autoencoder architecture is similar to the classification network, employing stacked 3D convolutional and max pooling layers at the bottom of the network to encode the input, followed by fully-connected layers to reduce dimensionality and subsequent stacked 3D convolutions and upsampling layers to decode the input (Fig. 2B). As in the classification architecture, all convolutional layers are paired with a ReLU activation. 3D CNN autoencoder networks were trained with the *Adadelta* optimization algorithm [45], utilizing crossentropy or mean-squared error as the loss function for binary and Gaussian representations respectively. This architecture resembles others in the literature [26], [27].

### F. 1D Recurrent Autoencoder Architecture

As with the 3D CNN autoencoders, our RNN autoencoder architecture resembles the corresponding classification network. Following the fully-connected layers in the classification architecture, the RNN autoencoder appends a 1D upsampling layer and mirror 1D convolutional layers to return the input back to the original size (Fig. 2D). Mean-squared error (MSE) against the input sequence was utilized as a loss function for training. RNN autoencoders were trained with the *Adam* optimizer with a learning rate of $\epsilon = 0.001$.

### G. 1D Recurrent Motility Prediction Archiecture

We adapt our RNN autoencoder architecture to a prediction architecture by removing the max pooling, fully-connected, and dropout layers. Sequences are convolved by four 1D convolutional layers, as in the autoencoder, before being passed to a linearized LSTM and convolved by four more 1D convolutional layers. The final convolutional layer uses a linear activation function rather than a ReLU. Input sequences length $\tau_{in}$ are provided in the same multi-channel time series format as our other RNN architectures, and output sequences

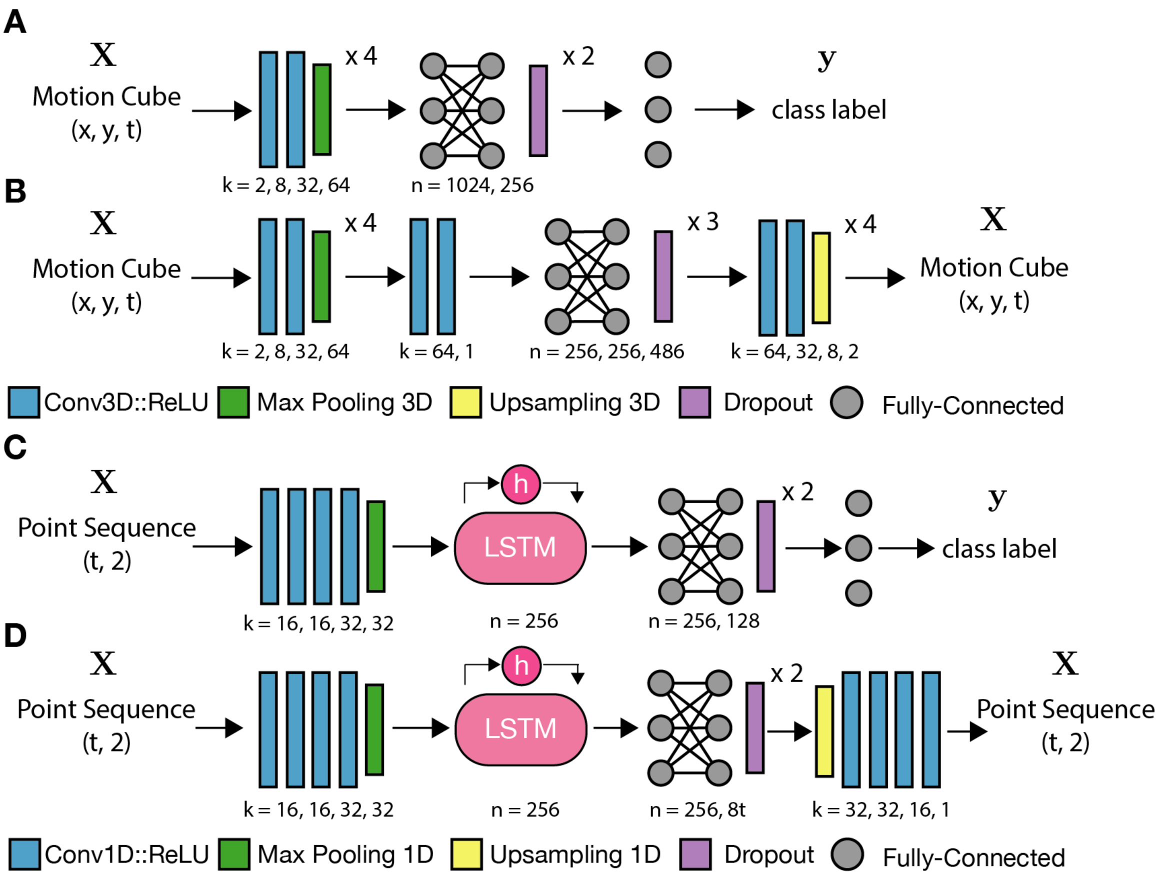

Fig. 2. Cell motility classification and autoencoder architecture overview. (A) 3D CNN classification and (B) autoencoder architectures, where $k$ is the number of parameterized kernels used by 3D convolutional layers in each block and $n$ is the number of nodes in a fully-connected layer. (C) 1D recurrent neural network classification and (D) autoencoder architectures, where $k$ is the number of parameterized kernels used by each 1D convolutional layer and $n$ is the number of nodes in a fully-connected layer or LSTM unit. Convolutional layers and paired with a rectified linear unit activation. Pooling and upsampling layers operate with isotropic kernels of size 2 and stride of 2. Zero padding is performed as needed in autoencoder models to match input size.

are multi-channel time series of length $\tau_{out}$. The number of LSTM units is adjusted to $n = 2\tau_{out}$ depending on the length of desired output sequences. Mean squared error between the predicted path and the ground truth path was used as the loss function and the *Adam* optimizer was used with learning rate $\epsilon = 0.001$ (Fig. 6C).

### H. Baseline Motility Classification

As a baseline motility classifier, a heuristic feature extractor is paired with a Random Forest (RF) classifier [46]. The feature extractor calculates four parameters of motion: (1) mean displacement, (2) displacement variance, (3) total distance traveled, and (4) net distance traveled. These four heurstics are commonly employed in the quantitative cell motility literature [21], [47], [48]. The RF classifier utilizes 10 estimators and scikit-learn default parameter settings. Code for the baseline classifier is available on Github.

### I. Baseline Kinematic Motion Prediction

A linear kinematic model is used for baseline motility predictions. The kinematic model calculates the mean velocity

$$\vec{v} = \frac{1}{\tau}\sum_{i=1}^{\tau} \vec{dv}_i$$

across the last $\tau$ time steps in the preceding track and projects the object by $\vec{v}$ for each predicted time step. The temporal window $\tau$ is optimized by parameter search.

### J. Cell Culture

Mouse embryonic fibroblasts, muscle stem cells, and myoblasts were cultured as previously described [15]. Neoplastic MEFs were generated as described and generously donated by the authors of [49].

### K. Timelapse Cell Imaging

Timelapse cell imaging, cell segmentation, and cell tracking was performed as described [15]. Briefly, cells were imaged for 10 hours in DIC at 6.5 minute intervals using a stagetop incubator at 37°C and 5% $CO_2$. Images were segmented using common heuristic techniques and tracking was performed using a modified version of uTrack [50]. Cell tracking data is available on the “Heteromotility” Github repository https://github.com/cellgeometry/heteromotility.

## III. Experimental Results

### A. Motility Simulations

To determine if CNNs could discriminate between different types of motion under ideal conditions, we trained both 3D

CNN and RNN classification networks on simulated data from 3 distinct, biologically relevant models of motion, namely random walking, Levy flights, and fractional Brownian motion. Random walking is motion with normally distributed random step sizes and directionality. Random walking is observed in freely diffusing biomolecules [51]. Levy flights similarly display random directionality, but step sizes are instead chosen from a long-tailed Levy distribution. Levy flights are observed in multiple biological systems and optimize path finding [52], [53], [54], [55]. Fractal Brownian motion models a random walk with long term dependence, similarly relevant as a representation of regulated motion in biology [56], [57]. By starting with simulated data we can optimize parameters using large sample sizes that would be difficult to obtain with living cells.

Random walks, Levy flights, and fractal Brownian motion were simulated for classification, each with a mean displacement of 5 $(x, y)$ units per time step. Simulations were carried out for 100 time steps and restricted to a $(2048, 2048)$ pixel plane, representing the field-of-view that might be expected using a standard 4 megapixel microscopy camera.

For 3D CNN classification with explicit representations, 4-fold compression and cropping were performed to meet GPU memory constraints. Compressed tracks traveling more than 156 px from the origin in any direction were removed from analysis, to prevent dilution of the representation space by a few outlier tracks. Remaining tracks were represented in $(156, 156, 101)$ vx cubes. Nvidia GTX 1080 (Pascal) and Titan Xp GPUs were used for all experiments.

### *B. CNNs accurately classify simulated motility behaviors using both representations*

Experiments to classify explicit 3D representations were performed using binary disks of three diameters $d \in \{1, 5, 25\}$ and broad Gaussian distributions of different variance $\sigma \in \{3, 10, 20\}$ as place markers. 12,000 samples per class were used for training, 1,500 for stopping criteria, and 1,500 for final validation. Early stopping was performed in all models after the testing loss failed to improve for 3 consecutive epochs [58]. Models were evaluated based on the prediction accuracy on the validation set.

The largest binary disk representations achieved ˜95% validation accuracy after 30 epochs, and the largest Gaussian representations of the same data yielded ˜81% validation after 30 epochs of training (Fig. 3A). Using binary distributions, accuracy increased as the marker sized increased. At all marker sizes, binary markers perform better than Gaussian markers. Both binary and Gaussian representations appear to overfit in later epochs, as evidenced by the divergence of the training performance from validation performance (Fig. 3B). A baseline random forest (RF) classifier model utilizing heuristic motility features (see Methods) reached ˜54% (5-fold cross-validation). The 3D CNN approach therefore represents a 75% improvement over the heuristic RF baseline.

These results suggest that 3D CNNs are sufficient to distinguish different classes of motion represented as 3D images, that multiple representation schemes can be effective, and that 3D CNNs can beat baseline heuristic classification methods by a wide margin. Large binary representation schemes appear to be the most effective representation scheme we tested. Therefore, we utilize large binary representations for all further 3D CNN experiments with live cell data.

Experiments to classify 1D multi-channel time series representations were performed using the same train, test, validation data split described above for 3D CNN experiments. Convolutional recurrent neural network (RNN) models train two orders of magnitude more rapidly than comparable 3D CNN models, and the reduced memory requirements allow for much larger batch sizes than in the 3D case. RNN classifiers achieve ˜99% validation accuracy, demonstrating superior performance to the 3D CNN models on simulated data. These results indicate that RNN models are sufficient to distinguish different models of motion represented as multi-channel time series, and that this classification scheme is superior to both a baseline heuristic approach and the 3D CNN approach for this task.

### *C. CNNs accurately discriminate cell types by motility behavior*

After validating that CNNs were sufficient to distinguish simulated classes of motion, we applied the same classification networks to distinguish different types of experimentally measured cell motility. Cell motility was tracked in three different cell types by timelapse imaging for 10 hours, followed by segmentation and tracking by standard methods. Mouse embryonic fibroblasts (MEFs) are commonly used for *in vitro* cell culture assays, and neoplastic transformation of these cells has been demonstrated to alter their motility behaviors [15]. We tracked both wild-type and neoplastic (*c-Myc* overexpression, *HRas-V12*) MEFs to compare their motility behaviors. Muscle stem cells (MuSCs) are the obligate stem cell of the skeletal muscle, and their motility is known to be effected by their activation state [14]. Activated MuSCs commit to become myoblasts, a transit amplifying myogenic progenitor cell. We tracked both MuSCs and myoblasts to compare motility between these states of myogenic commitment (see Methods for culture details).

To determine if 3D CNNs could distinguish cell types based on experimentally measured motility, we trained a 3D CNN to discriminate between MEF and MuSC motility, represented using large binary disks ($diameter = 25$ px) in 3D space as described above. RNN models were trained as on simulated models of motion above. Both networks were initialized with weights from the corresponding trained simulation classifier.

The 3D CNN classification network was trained for 30 epochs and RNN classifiers for 1000 epochs. Early stopping as above was performed with a 3 epoch and 100 epoch patience period for 3D CNNs and RNNs, respectively. Training was performed on MuSC motility traces and MEF motility traces ($n = 405$ per class). Testing and validation were each performed with $n = 50$ samples per class.

Each network type was trained five separate times to account for variability in stochastic optimization. Mean validation accuracy on this cell type classification task was ˜91.2% for 3D CNN models and ˜92.6% for RNN models (Fig. 4A). Mean

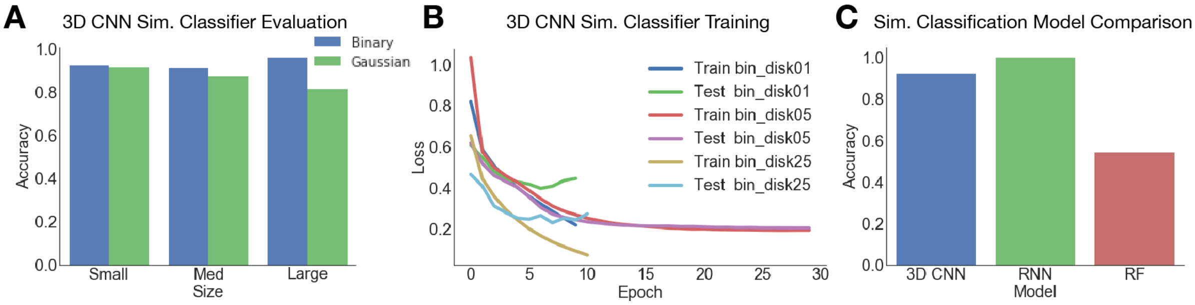

Fig. 3. 3D CNN and RNN classifiers effectively distinguish simulated models of motion. (A) 3D CNN classifier performance using different marker sizes. (B) 3D CNN classifier training progress with different binary marker sizes. (C) RNN and 3D CNN (binary marker, 25 px) classifier performance.

validation accuracy for our heuristic RF baseline model was ˜86.6%. These results indicate that even with a small data set such as this, both 3D CNNs and RNNs can be effectively trained to discriminate different types of cell motility and are superior to a baseline heuristic model (Fig. 4A).

### D. CNNs provide discriminative power between stem cell activation states

To determine if CNNs can distinguish between more nuanced differences in cell state, both 3D CNN and RNN classifiers were trained to discriminate between myogenic activation states. Training was performed on $n = 200$ MuSCs and myoblasts per class, with testing on $n = 50$ samples and validation on $n = 27$ samples. Dynamic data augmentation was utilized in the 3D CNNs due to the small available sample size. Motion cubes were horizontally and vertically flipped to increase training set diversity without perturbing the representation of motility. No augmentation was used for RNN models.

As above, five independent training experiments were performed for each network. Transfer learning was employed, taking advantage of weights learned from simulated data classification. Mean validation accuracy reached ˜90.7% for 3D CNN classifiers, ˜90.4% for RNN classifiers, and ˜88.5% for our heuristic baseline (Fig. 4A). These results demonstrate that both 3D CNN and RNN models can discriminate between stem cell activation states based on motility alone and are superior to a baseline heuristic model, even with small data sets.

Classifiers were also trained in the same manner to discriminate between wild-type and neoplastic MEFs with transfer learning from the simulated motion classifier. Training was performed on $n = 160$ samples per class, with testing and validation on $n = 30$ samples per class. Classification failed to achieve validation accuracy >64% for either 3D CNN or RNN models (Fig. 4A). However, this is still notably superior to the baseline heuristic model, which performed with accuracy near the noise floor at 54.6%. The more nuanced phenotypic difference between wild-type and neoplastic MEFs may be an inherently more challenging classification problem. The small available sample size likely compounds this difficulty and exacerbates the classifier's poor performance.

### E. Cell mimetic pretraining

Given the success of pre-training by classification of simulated models of motion, we next attempted to generate simulated data that more accurately reflected real cell motility to enhance pre-training efficacy. For a set of real cell motility data, we measure the displacements and turning behavior of each cell. Displacements are measured simply as the Euclidean distance between each set of sequential timepoints. The turning direction at a point $t_i$ is determined as the angle between the vectors that connect points $t_{i-1}$ to $t_i$ and $t_i$ to $t_{i+1}$.

Cells are decomposed into a set of $k$ clusters by $k$-means clustering on a set of parameters measured from these displacement and turn angle distributions. The number of clusters $k = 5$ was chosen empirically to capture the diversity of the cell phenotypes while still leaving non-trivial numbers of cells in each cluster. For each cluster, a bounded Johnson distribution is fit to the aggregate distribution of displacements and the aggregate distribution of turn angles. Simulated samples are generated by randomly sampling displacement magnitudes and turn angles from the fitted Johnson distributions for $T$ time steps. To represent a population of cells, the proportion of simulations generated from each cluster is equivalent to the cluster's prevalence in the original cell data. This approach may be conceptually likened to the bag-of-words model [59], in which $k$-means clustering is used to decompose features into a representative "vocabulary." By sampling from each of $k$ clusters proportionally, we aim to capture and simulate heterogeneous phenotypes within a cell population, rather than simply reproducing a single averaged phenotype that may not be representative of any true cell phenotype.

We generated "cell mimetic" simulations for MuSCs and myoblasts by the above method, with $n = 15,000$ simulated samples for each of the two activation states. 3D CNN and RNN classifiers were pretrained by classifying between the two simulated data sets, reaching ˜97% validation accuracy for the 3D CNN classifier and ˜99% for the RNN classifier. The weights from this pretrained network were used to initialize classifiers for the myogenic activation task outlined above.

Training speed appeared to increase for 3D CNNs and remain unchanged for RNN classifiers (Fig. 4B). Mean validation accuracies were effectively unchanged at ˜91.1% for 3D CNN classifiers and ˜88.8% for RNN classifiers. These results indicate that "mimetic" pretraining may aide training

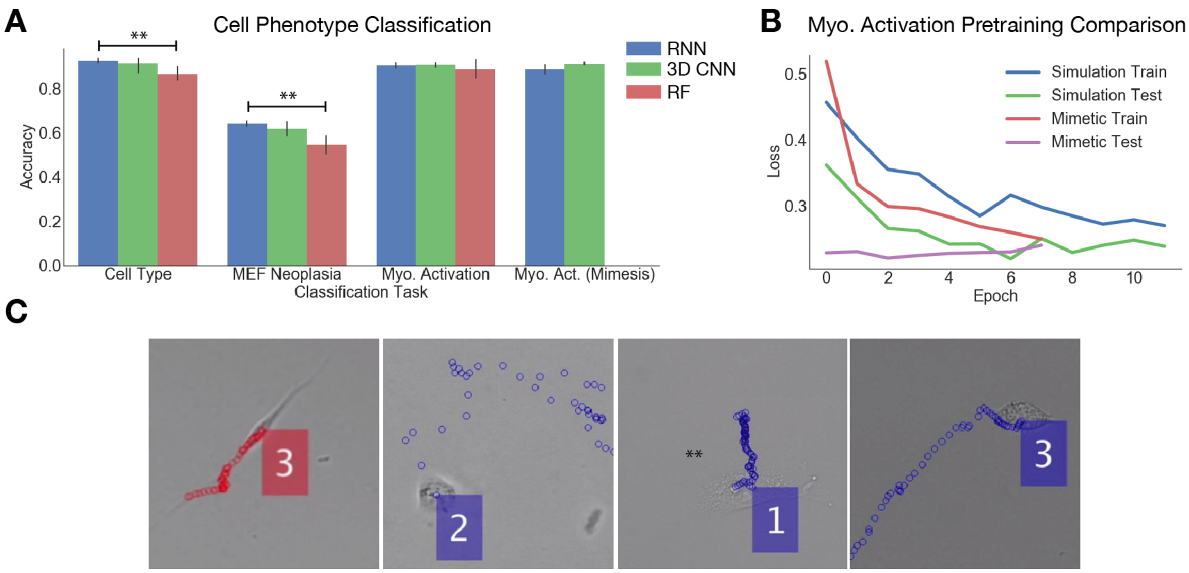

Fig. 4. CNNs can discriminate between different cell types and cell states based on motility. (A) Validation prediction accuracy for 3D CNN and RNN models on cell type classification, MEF neoplastic state classification, and myogenic activation state classification with either simulated or "mimetic" pretraining. (***t*-test $p < 0.001$) (B) Representative training progress for 3D CNN classifiers pretrained with either generic simulation classification wieghts or mimetic simulation classification weights. (C) Representative images of different cell types, from left to right: MuSC, myoblast, neoplastic MEF, and wild-type MEF. Colored markers indicate the cell's path along the substrate over time.

speed for 3D CNN motility classification networks, and has little effect on final classification accuracy.

### *F. Autoencoders allow unsupervised learning of representations in motion feature space*

Results up to this point indicate that supervised classification of different cell motility phenotypes using both 3D CNN and RNN models is effective. However, in the analysis of motility data, supervised classification data is not always available. For instance, to explore the heterogeneity of types in a given population, there is no obvious method to generate supervised classification data that may be used to learn relevant feature kernels by optimization of a standard classification loss function. This would also be an issue in the identification of heterogeneous motility behaviors in patient biopsy samples, in which the distinguishing features are not known *a priori*. Training CNNs as autoencoders in an unsupervised fashion has been used in other contexts to learn relevant feature kernels where no obvious classification problem is present [27], [28]. We next attempted to train autoencoders on our 3D representations of cell motility and multi-channel time series to learn relevant feature kernels in the absence of a supervised classification problem.

A 3D CNN autoencoder architecture was formulated using stacked convolutions, followed by fully-connected layers, upsampling, and stacked convolutional layers (Fig. 2B). Similarly, an RNN autoencoder was formulated by appending upsamping and convolutional layers following the fully-connected layers in our classification architecture (Fig. 2D).

Both autoencoders were trained on $n = 12,000$ samples of each class for three types of simulated motion (random walk, Levy flight, fractal Brownian motion), with training and validation each on $n = 1,500$ samples per class. Large binary disk representations were used for 3D CNN autoencoder experiments. Binary crossentropy was used as a loss function with 3D CNN models, while mean squared error was used to as a loss function for RNN models. All autoencoders successfully reduced loss over several training epochs. When visually inspected, 3D CNN autoencoder outputs appear to accurately reflect input motility representations (Fig. 5A, B). However, RNN autoencoder outputs consistently fail to capture the full extent of a cell's motility, though some degree of path shape is preserved (Fig. 5D).

To determine if 3D CNN or RNN autoencoders trained on motility representations could be employed as feature extractors, we utilized the output of the autoencoders' central layer (the encoded representation) as features. To quantify the amount of class information preserved by the encoded representations, we trained a Random Forest classifier to distinguish the simulation classes using either 3D CNN autoencoder features or RNN autoencoder features. Random Forests trained on 3D CNN autoencoder features achieved ˜62.3% and RNN autoencoder features achieved ˜58.2% accuracy on this 3 class problem. Both Random Forests trained on autoencoder features are notably more effective than our heurstic RF baseline, which achieved ˜54.3% accuracy on the same task. These results indicate that both 3D CNN and RNN autoencoders are able to learn meaningful motility features in an unsupervised manner. In this context, 3D CNN autoencoder features appear to be marignally more predictive than RNN autoencoder counterparts.

Autoencoders are often used for unsupervised pre-training prior to a classification problem. To determine if autoencoder features could effectively aide motility classification by transfer learning, we initialized and 3D CNN and RNN models with autoencoder weights and trained these classifiers to dis-

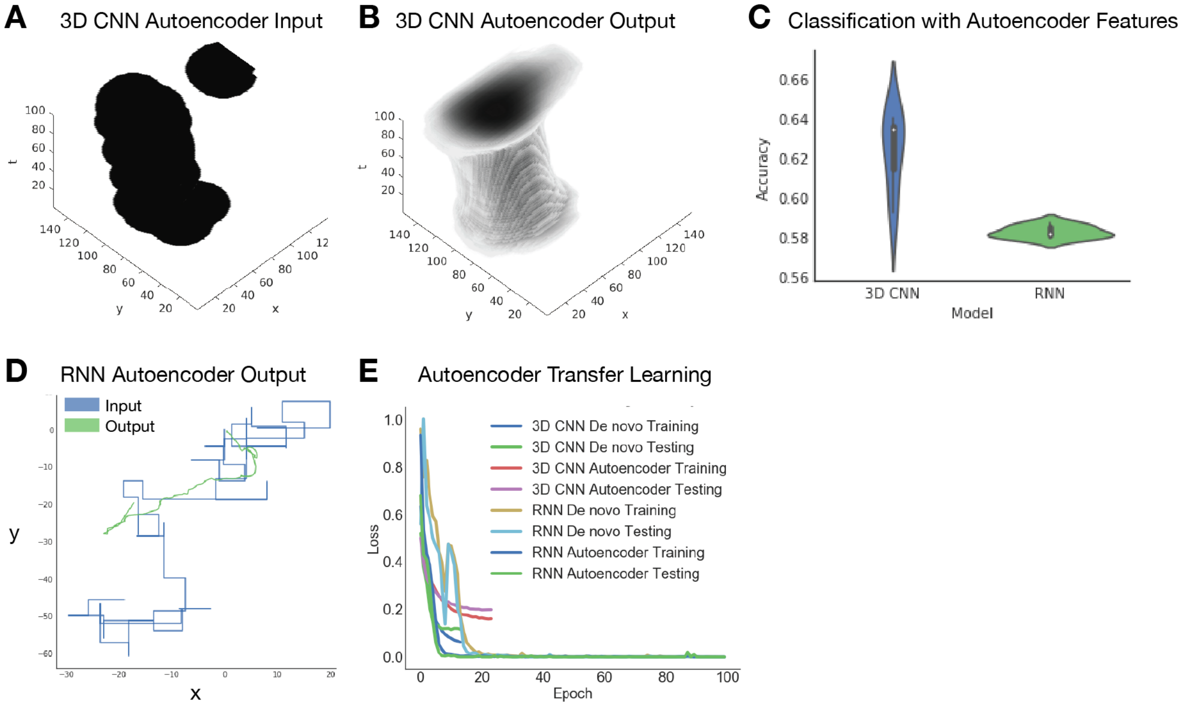

Fig. 5. CNN autoencoders can learn representations of motion feature space in an unsupervised manner. (A) Sample 3D CNN autoencoder input and (B) output. (C) Classification accuracies of Random Forest classifiers trained on 3D CNN or RNN autoencoder features. (D) RNN autoencoder sample input and output. (E) Training comparison of de novo trained simulated motility classifiers to classifiers with autoencoder transfer learning.

tinguish the simulated motion models. Transfer learning with autoencoder features appears to increase RNN training speed, but decrease 3D CNN training speed (Fig. 5E). This result indicates that autoencoding motility data may be effective pretraining for our RNN classification architectures, but not their 3D CNN counterparts.

### *G. RNNs predict muscle stem cell motility*

Tracking individual cells in timelapse microscopy experiments is a difficult multi-object tracking problem [60]. Popular tracking methods utilize a motion model to predict cell motility in advance of the next frame to improve tracking performance [61]. This motion prediction is especially useful in the event of “missed detections,” where a cell is not detected or segmented for a given set of frames but is detected later on. The most common motion models employed are based on linear kinematics, with Kalman filters serving as a popular choice [50]. However, cell motion does not adhere to kinematic assumptions in all cell types, with myogenic cells being an excellent example of such a system. A motion model specifically tailored to the cell type of interest may therefore be useful to improve tracking performance, but such specific tailoring would require a prior knowledge of the very motion features that the live cell experiment is designed to analyze. Some way to tailor prediction models on the fly could help solve this problem.

Recurrent neural networks have been effectively utilized for sequence prediction in multiple fields [62], [63], [64], [65]. We adapted the convolutional RNN autoencoder model to a sequence prediction model by removing the pooling layers and fully-connected layers and altering the number of nodes in the central LSTM layer (Fig. 6C). As a prediction task, we trained the RNN prediction model on $\tau_{in} = 20$ time steps of motion and predicted $\tau_{out} = 10$ time steps into the future. As a data set, we split MuSC motion paths into subpaths of length $\tau_{total} = 30$ for a total of $n = 8,676$ paths. A validation set of 10% of all tracks was held out, and the remaining tracks were split with 80% used for training and 20% used for testing.

As a baseline for comparison, we performed a simple kinematic prediction of MuSC paths that assumes persistence of the velocity from preceding time points. The velocity for prediction was obtained by averaging instantaneous velocity for $\tau = 15$ time points prior to the track terminus, where $\tau$ was optimized by parameter search. This baseline model leads to an average mean squared error (MSE) (30 train/test splits) of ˜220. The RNN prediction model by comparison produces a significantly lower MSE of ˜192 ($t$-test $p < 0.001$), indicating that the RNN model is a superior motion predictor in the MuSC context (Fig. 6B). Representative track endings (length $\tau_{out} = 10$) produced by the RNN prediction model are displayed alongside the preceding track beginnings (length $\tau_{in} = 20$) and the ground truth track endings (Fig. 6A). In most cases the motion prediction reasonably approximates the cell's ground truth direction, but does not closely mirror the exact path (Fig. 6A, inset **i** and **ii**). In some events, the RNN model fails to predict even the correct direction of motion (Fig. 6A, inset **iii**). We performed the same experiment with mimetic myoblast simulations using $n = 10^5$ total samples, holding $n = 5000$ samples for validation. Similar to the MuSC results, RNN motion predictors achieved a markedly lower MSE of

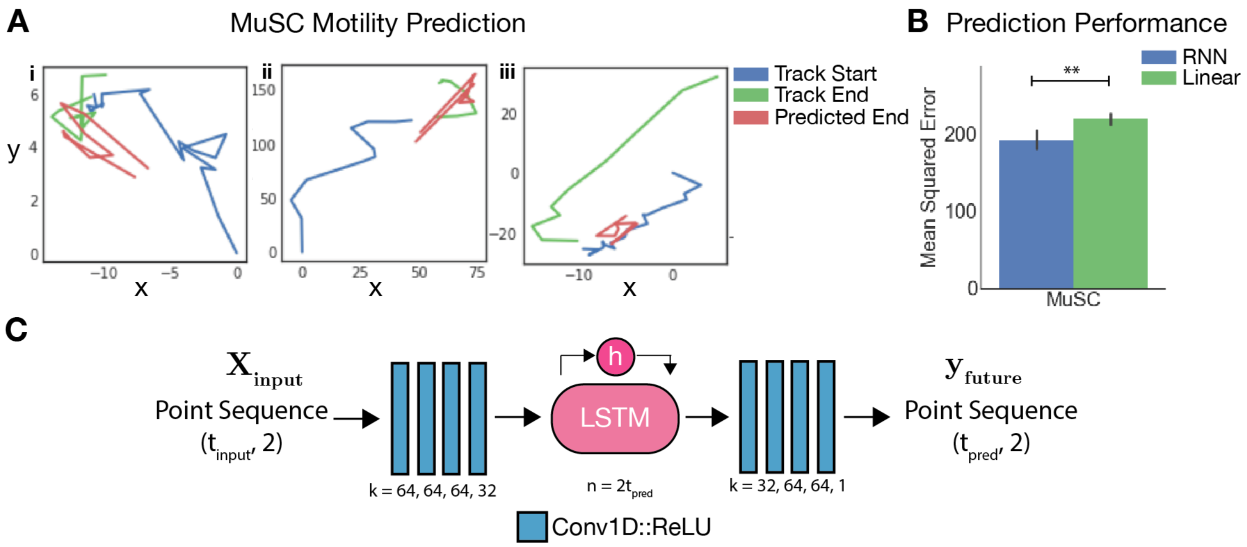

Fig. 6. RNN models predict MuSC motility more effectively than linear kinematics. (A) Representative samples of MuSC tracks used for prediction with predicted track endings and true track endings. (B) Performance of the RNN motion prediction model relative to a linear kinematic baseline model, determined as the mean squared error between ground truth and predicted track endings. (***t*-test $p < 0.001$) (C) RNN motility prediction architecture, where $k$ is the number of kernels in each 1D convolutional layer and $n$ is the number of units in the LSTM. All convolutional layers except the final layer are paired with a ReLU activation.

˜1195, relative to the baseline kinematic model MSE of ˜9797 (*t*-test $p < 0.001$).

These results indicate that convolutional RNN models can be effective cell motility prediction models and are superior to simple linear kinematic approaches in some real world circumstances. RNN motility prediction models may therefore offer a scalable way to fit a uniquely tailored motion model to specific cell biology contexts. These cell-context specific RNN motility predictors may be useful to improve multi-cell tracking performance, as outlined above.

## IV. Conclusion

Convolutional neural networks enable representation learning, or learning of features relevant for the description of a feature space. By representing cell motility as a 3D image, we show that 3D CNNs may be applied as an effective analytical tool. Using RNNs, motility may also be analyzed in the native multi-channel time series representation. Our results demonstrate that both approaches are capable of discriminating between simulated models of motion and multiple types of experimentally measured cell motility behaviors and are superior to a baseline heuristic model. In our experimentally measured cell motility data, we find that both CNN models effectively discriminate between different cell types, and different states of myogenic progenitor activation. We also find that CNN autoencoders can be trained effectively on either motion representation representations in an unsupervised fashion. Adapting the convolutional RNN autoencoder for motility prediction, we find that the RNN model is more effective at predicting MuSC motility than a kinematic model. Such prediction models may be useful for cell tracking. While we apply the methods described here to cell biology, there is no conceptual limitation that prevents application to other fields where discrimination based on motion recordings is desired. In the field of cell biology, analysis of motility with CNNs may allow for useful insights to be gathered in contexts where relevant features are non-obvious or laborious to construct.

## Acknowledgment

This work was funded by NSF grant MCB-1515456 to W.F.M., NIH grants AR060868 and AR061002 to A.S.B., and a PhRMA Foundation fellowship to J.C.K. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1650113 to J.C.K. J.C.K and W.F.M. are members of the NSF Center for Cellular Construction, NSF Grant No. 1548297. We gratefully acknowledge the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for this research. We would like to thank Morgan Truitt and Davide Ruggero for providing neoplastic MEF cells.

## References

[1] T. J. Mitchison and L. P. Cramer, "Actin-based cell motility and cell locomotion." *Cell*, vol. 84, no. 3, pp. 371–379, Feb. 1996.

[2] D. A. Lauffenburger and A. F. Horwitz, "Cell migration: A physically integrated molecular process," *Cell*, vol. 84, no. 3, pp. 359–369, 1996.

[3] H. T. Enterline and D. R. Coman, "The ameboid motility of human and animal neoplastic cells." *Cancer*, vol. 3, no. 6, pp. 1033–1038, Nov. 1950.

[4] D. R. Coman, "Mechanism of the Invasiveness of Cancer," *Science*, vol. 105, no. 2727, pp. 347–348, Apr. 1947.

[5] E. Sahai, M. F. Olson, and C. J. Marshall, "Cross-talk between Ras and Rho signalling pathways in transformation favours proliferation and increased motility." *The EMBO Journal*, vol. 20, no. 4, pp. 755–766, Feb. 2001.

[6] C. B. Pollock, S. Shirasawa, T. Sasazuki, W. Kolch, and A. S. Dhillon, "Oncogenic K-RAS is required to maintain changes in cytoskeletal organization, adhesion, and motility in colon cancer cells," *Cancer Res*, vol. 65, no. 4, pp. 1244–1250, 2005.

[7] M. A. Nieto, "Epithelial plasticity: a common theme in embryonic and cancer cells." *Science (New York, N.Y.)*, vol. 342, no. 6159, pp. 1 234 850–1 234 850, Nov. 2013.

[8] C. L. Smith, O. Kilic, P. Schiapparelli, H. Guerrero-Cazares, D.-H. Kim, N. I. Sedora-Roman, S. Gupta, T. O'Donnell, K. L. Chaichana, F. J. Rodriguez, S. Abbadi, J. Park, A. Quiñones-Hinojosa, and A. Levchenko, "Migration Phenotype of Brain-Cancer Cells Predicts Patient Outcomes," *Cell Reports*, vol. 15, no. 12, pp. 2616–2624, Jun. 2016.

[9] D. J. Laird, U. H. von Andrian, and A. J. Wagers, "Stem Cell Trafficking in Tissue Development, Growth, and Disease," *Cell*, vol. 132, no. 4, pp. 612–630, Feb. 2008.

[10] M. Buckingham, L. Bajard, T. Chang, P. Daubas, J. Hadchouel, S. Meilhac, D. Montarras, D. Rocancourt, and F. Relaix, "The formation of skeletal muscle: from somite to limb." *Journal of Anatomy*, vol. 202, no. 1, pp. 59–68, Jan. 2003.

[11] S. M. Hughes and H. M. Blau, "Migration of myoblasts across basal lamina during skeletal muscle development." *Nature*, vol. 345, no. 6273, pp. 350–353, May 1990.

[12] C. F. Bentzinger, J. von Maltzahn, N. A. Dumont, D. A. Stark, Y. X. Wang, K. Nhan, J. Frenette, D. D. W. Cornelison, and M. A. Rudnicki, "Wnt7a stimulates myogenic stem cell motility and engraftment resulting in improved muscle strength." *J Cell Biol*, vol. 205, no. 1, pp. 97–111, Apr. 2014.

[13] L. A. S. Alfaro, S. A. Dick, A. L. Siegel, A. S. Anonuevo, K. M. McNagny, L. A. Megeney, D. D. W. Cornelison, and F. M. V. Rossi, "CD34 Promotes Satellite Cell Motility and Entry into Proliferation to Facilitate Efficient Skeletal Muscle Regeneration," *Stem Cells*, vol. 29, no. 12, pp. 2030–2041, Nov. 2011.

[14] A. L. Siegel, K. Atchison, K. E. Fisher, G. E. Davis, and D. D. W. Cornelison, "3D Timelapse Analysis of Muscle Satellite Cell Motility," *Stem Cells*, vol. 27, no. 10, pp. 2527–2538, Oct. 2009.

[15] J. C. Kimmel, A. Y. Chang, A. S. Brack, and W. F. Marshall, "Inferring cell state by quantitative motility analysis reveals a dynamic state system and broken detailed balance," *PLoS computational biology*, vol. 14, no. 1, pp. e1 005 927–29, Jan. 2018.

[16] M. T. Tierney and A. Sacco, "Satellite Cell Heterogeneity in Skeletal Muscle Homeostasis." *Trends Cell Biol*, vol. 26, no. 6, pp. 434–444, Jun. 2016.

[17] A. S. Sebag, S. Plancade, C. Raulet-Tomkiewicz, R. Barouki, J.-P. Vert, and T. Walter, "A generic methodological framework for studying single cell motility in high-throughput time-lapse data," *Bioinformatics*, vol. 31, no. 12, pp. 320–328, 2015.

[18] A. Sacan, H. Ferhatosmanoglu, and H. Coskun, "CellTrack: an open-source software for cell tracking and motility analysis." *Bioinformatics*, vol. 24, no. 14, pp. 1647–1649, Jul. 2008.

[19] L. Martens, G. Monsieur, C. Ampe, K. Gevaert, and J. Vandekerckhove, "BMC Bioinformatics," *BMC bioinformatics*, vol. 7, no. 1, pp. 289–6, 2006.

[20] M.-A. Bray and A. E. Carpenter, "CellProfiler Tracer: exploring and validating high-throughput, time-lapse microscopy image data." *BMC bioinformatics*, vol. 16, no. 1, p. 368, 2015.

[21] A. R. Cohen, F. L. A. F. Gomes, B. Roysam, and M. Cayouette, "Computational prediction of neural progenitor cell fates." *Nature methods*, vol. 7, no. 3, pp. 213–218, Mar. 2010.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems 25*, pp. 1097–1105, 2012.

[23] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," *Molecular Systems Biology*, vol. 12, no. 7, p. 878, 2016.

[24] K. He, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Microsoft Research, Redmond, United States, Jan. 2016, pp. 770–778.

[25] D. A. Van Valen, D. A. Van Valen, T. Kudo, T. Kudo, K. M. Lane, K. M. Lane, D. N. Macklin, D. N. Macklin, N. T. Quach, M. M. DeFelice, I. Maayan, Y. Tanouchi, E. A. Ashley, and M. W. Covert, "Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments," *PLoS computational biology*, vol. 12, no. 11, p. e1005177, Nov. 2016.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[27] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *ICLR*, Dec. 2013.

[28] Y. Bengio, P. Lamblin, and D. Popovici, "Greedy layer-wise training of deep networks," *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems*, 2007.

[29] A. Conneau and H. Schwenk, "Very deep convolutional networks for text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2016, pp. 1107–1116.

[30] L. Neal, F. Briggs, and R. Raich, "Time-frequency segmentation of bird song in noisy acoustic environments," *Acoustics*, pp. 2012–2015, 2011.

[31] P. Bashivan, I. Rish, M. Yeasin, and N. Codella, "Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks," in *ICLR*, Nov. 2015.

[32] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand Gesture Recognition With 3D Convolutional Neural Networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–7.

[33] D. Wu, L. Pigou, P.-J. Kindermans, N. D.-H. Le, L. Shao, J. Dambre, and J.-M. Odobez, "Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition." *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 8, pp. 1583–1597, Aug. 2016.

[34] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4694–4702, 2015.

[35] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale Video Classification with Convolutional Neural Networks," *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1725–1732, 2014.

[36] D. Quang and X. Xie, "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences," *Nucleic Acids Research*, vol. 44, no. 11, pp. e107–e107, Jun. 2016.

[37] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnology*, vol. 33, no. 8, pp. 831–838, Jul. 2015.

[38] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A Critical Review of Recurrent Neural Networks for Sequence Learning," *arXiv*, May 2015.

[39] Z. Cui, W. Chen, and Y. Chen, "Multi-Scale Convolutional Neural Networks for Time Series Classification," *arXiv*, Mar. 2016.

[40] Z. Wang, W. Yan, and T. Oates, "Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline," *arXiv*, Nov. 2016.

[41] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, Sep. 2014, pp. 730–734.

[42] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 807–814.

[43] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, Jan. 1989.

[44] N. Srivastava, G. E. Hinton, and A. Krizhevsky, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[45] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *arXiv*, Dec. 2012.

[46] T. K. Ho, "Random decision forests," in *3rd International Conference on Document Analysis and Recognition*. IEEE Comput. Soc. Press, Aug. 1995, pp. 278–282.

[47] R. Gorelik and A. Gautreau, "Quantitative and unbiased analysis of directional persistence in cell migration," *Nature Protocols*, vol. 9, no. 8, pp. 1931–1943, Jul. 2014.

[48] J. G. Lock, M. J. Mamaghani, H. Shafqat-Abbasi, X. Gong, J. Tyrcha, and S. Strömblad, "Plasticity in the macromolecular-scale causal networks of cell migration." *PLoS One*, vol. 9, no. 2, p. e90593, 2014.

[49] M. L. Truitt, C. S. Conn, Z. Shi, X. Pang, T. Tokuyasu, A. M. Coady, Y. Seo, M. Barna, and D. Ruggero, "Differential Requirements for eIF4E Dose in Normal Development and Cancer," *Cell*, vol. 162, no. 1, pp. 59–71, Jul. 2015.

[50] K. Jaqaman, D. Loerke, M. Mettlen, H. Kuwata, S. Grinstein, S. L. Schmid, and G. Danuser, "Robust single-particle tracking in live-cell time-lapse sequences," *Nature methods*, vol. 5, no. 8, pp. 695–702, Jul. 2008.

[51] H. C. Berg, *Random Walks in Biology*. Princeton University Press, 1993.

[52] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, E. J. Murphy, P. A. Prince, and H. E. Stanley, "Levy flight search patterns of wandering albatrosses," *Nature*, vol. 381, no. 6581, pp. 413–415, 1996.

[53] G. M. Viswanathan, E. P. Raposo, and M. G. E. da Luz, "Levy flights and superdiffusion in the context of biological encounters and random searches," *Physics of Life Reviews*, vol. 5, no. 3, pp. 133–150, Sep. 2008.

[54] M. A. Lomholt, K. Tal, R. Metzler, and K. Joseph, "Lévy strategies in intermittent search processes are advantageous," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 32, pp. 11 055–11 059, Aug. 2008.

[55] G. M. Viswanathan, S. V. Buldyrev, S. Havlin, M. G. da Luz, E. P. Raposo, and H. E. Stanley, "Optimizing the success of random searches." *Nature*, vol. 401, no. 6756, pp. 911–914, Oct. 1999.

[56] B. B. Mandelbrot and J. W. Van Ness, "Fractional Brownian motions, fractional noises and applications," *SIAM review*, vol. 10, no. 4, pp. 422–437, 1968.

[57] B. B. Mandelbrot, "A Fast Fractional Gaussian Noise Generator," *Water Resources Research*, vol. 7, no. 3, pp. 543–553, Jun. 1971.

[58] L. Prechelt, "Early Stopping — But When?" in *Neural Networks: Tricks of the Trade: Second Edition*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 53–67.

[59] J. Sivic and A. Zisserman, "Efficient visual search of videos cast as text retrieval." *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 4, pp. 591–606, Apr. 2009.

[60] S. Skylaki, O. Hilsenbeck, and T. Schroeder, "Challenges in long-term imaging and quantification of single-cell dynamics," *Nature Biotechnology*, vol. 34, no. 11, pp. 1137–1144, Nov. 2016.

[61] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim, "Multiple Object Tracking: A Literature Review," *arXiv*, Sep. 2014.

[62] J. Bütepage, M. Black, D. Kragic, and H. Kjellström, "Deep representation learning for human motion prediction and classification," *arXiv*, Feb. 2017.

[63] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," *arXiv*, May 2017.

[64] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C*, vol. 54, no. C, pp. 187–197, May 2015.

[65] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," *arXiv*, Jun. 2015.