3D SINGLE-CELL SHAPE ANALYSIS OF CANCER CELLS USING GEOMETRIC DEEP LEARNING

A PREPRINT

Matt De Vries[1,2,*]
matt.devries@icr.ac.uk

Lucas Dent[1*]
lucas.dent@icr.ac.uk

Nathan Curry[2]
nathan.curry08@imperial.ac.uk

Leo Rowe-Brown[2]
l.rowe-brown19@imperial.ac.uk

Adam Tyson[3]
adam@adamltyson.com

Christopher Dunsby[2]
christopher.dunsby@imperial.ac.uk

Chris Bakal[1]
chris.bakal@icr.ac.uk

ABSTRACT

Aberrations in cell geometry are linked to cell signalling and disease. For example, metastatic melanoma cells alter their shape to invade tissues and drive disease. Despite this, there is a paucity of methods to quantify cell shape in 3D and little understanding of the shape-space cells explore. Currently, most descriptions of cell shape rely on predefined measurements of cell regions or points along a perimeter. The adoption of 3D tissue culture and imaging systems in medical research has recently created a growing need for comprehensive 3D shape descriptions of cells. We have addressed this need using unsupervised geometric deep learning to learn shape representations of cells from 3D microscopy images of metastatic melanoma cells embedded in collagen tissue-like matrices. We used a dynamic graph convolutional foldingnet autoencoder with improved deep embedded clustering to simultaneously learn lower-dimensional representations and classes of 3D cell shapes from a dataset of more than 70,000 drug-treated melanoma cells imaged by high throughput light-sheet microscopy. We propose describing cell shape using 3D quantitative morphological signatures, which represent a cell's similarity to shape modes in the dataset, and are a direct output from our model. We used the extracted features to reveal the extent of the cell shape landscape and found that the shapes learned could predict drug treatment (up to 86% accuracy) and cell microenvironment, and are explainable. In particular, we found strikingly similar deep learning shape signatures between cells treated with microtubule polymerisation inhibitors and branched actin inhibitors. Finally, we implemented our methods as a Python package for ease of use by the medical research community.

Keywords 3D, geometric, deep learning, single-cell, shape

Introduction

A central principle in biology is that the shape of living things is connected to their function. Shape is determined by the interaction of evolution with physical laws [1, 2], and aberrations in shape can be a cause or sign of disease [3]. This connection between shape and function holds at every spatial scale from molecules to cells, to organs, and organisms[4].

[1]Dynamical Cell Systems, Department of Cancer Biology, Institute of Cancer Research, London, SW3 6JB, UK
[2]Photonics Group, Department of Physics, Imperial College London, London, SW7 2A, UK
[3]Sainsbury Wellcome Centre & Gatsby Computational Neuroscience Unit, University College London, London, W1T 4JG, UK
[*]These authors contributed equally to this work

Due to the importance of shape, biologists have developed many descriptors that attempt to measure and describe it. Traditionally, we understand shape using predetermined or "classical" features. These are pre-selected, mathematically defined measures of geometry and often require a preconceived idea of which features are important for the task at hand. Geometric descriptors include region-based shape descriptors such as area, perimeter, and eccentricity [5, 6, 7, 8, 9], as well as boundary features such as the relative position of consecutive points along a 2D perimeter [10, 11]. In the context of studying living cells in 2D images, these geometric approaches have been remarkably successful at identifying major categories of cell shape, and linking cell shapes to the molecular signalling pathways that control geometry [8, 12, 3, 13], or respond to changes in geometry [14, 15, 16, 17].

While measuring shape in 2D images has been illuminating, cells exist in 3D environments and their 3D geometry is a rich source of information about the inner workings of cells and tissues. For example, recent pioneering work has demonstrated that shape motifs extracted from 3D images can be linked to lipid membrane signalling and cancer signalling pathways [14], and changes in 3D nuclear morphometry are linked to chromatin organisation [18]. At the scale of developing tissues, the discovery of new 3D geometries has explained how cells pack stably into curved organs [19]. Now, the development of high throughput light-sheet imaging [20, 21] has made it possible to image large numbers of cells in 3D with sub-cellular resolution. This has led to a growing need for accurate 3D shape representations. We have addressed this need by building upon recent advances in deep learning (DL), and the field of computer vision.

Robust representation of the shape of 3D objects is a major goal in the field of computer vision [22], and there has been rapid progress in this domain. Supervised and unsupervised methods of shape description have been built on large-scale labelled datasets of 3D models of everyday objects [22, 23, 24]. State-of-the-art methods have analysed these 3D models as either 3D binary voxelised objects, or point clouds for tasks of classification [25, 26, 27, 28] and unsupervised representation learning [29, 30, 31, 32, 33, 34, 35]. Point cloud representation is an alternative approach to representing 3D images with regular dense voxels. Point clouds are composed of unordered and irregular sets of points, $\mathbf{P} \in \mathbb{R}^{N \times 3}$. Point clouds are scattered collections of points in 3D space and are arguably the simplest shape representation. These are easily obtainable once a segmented mask has been created and are memory-efficient when compared to 3D images. 3D images grow in memory depending on the size of the object or the number of voxels, however, point clouds may be represented by a fixed set of points so that larger objects use the same amount of memory as smaller objects. This means that methods using point cloud representation are easily scalable. This data representation is the most commonly used in 3D shape analysis primarily because this is the raw data of object surfaces received from LIDAR sensors in applications of autonomous driving, robotics, and scene understanding. This has led to in-depth research into DL architectures which explore local geometric information, using convolution, graph, and attention mechanisms on point clouds for improvements in 3D shape analysis.

DL on point cloud data often involves representing point clouds as graphs and then performing geometric deep learning (GDL) on these graphs [25, 36, 27, 37, 38]. In computer science, graphs are data structures that consist of nodes and edges. DL on graphs requires adaptations as the data structure is non-Euclidean and makes regular operations such as convolution rather elusive. GDL is a branch of deep learning that consists of methods to generalise DL methods to non-Euclidean domains such as graphs [39].

Feature or representation learning are techniques that enable a machine to automatically learn representations needed for downstream tasks including clustering or classification. Autoencoders are DL algorithms that are commonly used for representation learning tasks. In general, autoencoders are machines that consist of an encoder and a decoder part where the encoder maps the input to a lower-dimensional space and the decoder attempts to reconstruct the input. Autoencoders are trained by minimising the difference between the input and the reconstructed data. Graph-based autoencoders learn lower-dimensional representations of graph-structured data and have been used to learn representations of 3D shapes[32, 31, 40].

Motivated by the need to extract information embedded in 3D shape, and by the effectiveness of GDL approaches in generating 3D shape representations, we have used artificial intelligence to reveal the geometric landscape of biological cells. In this study, we build on the progress made in computer vision and use neural networks to automatically learn high-quality shape representations for single cells in 3D microscopy images. To our knowledge, this represents the first unsupervised approach to the description of 3D cell morphology. After the generation of high-quality 3D shape representations we use our GDL features to predict drug treatments, and explore and explain the features learned. Finally, the methods have been coordinated in an open-source Python package for ease of use for the research community.

The code and links to all repositories used in this study are freely available at `https://github.com/DeVriesMatt/cellshape`.

Results

Data acquisition and processing

To create a dataset of cell and nuclei 3D shapes that exist in a human cell line, we used light-sheet microscopy to image more than 70,000 WM266.4 metastatic melanoma cells embedded in tissue-like collagen matrices. To sample a wide range of the possible shapes that cells can make we treated cells with inhibitors of the cytoskeleton (Figure 1A), as well as inhibitors of cells signalling pathways. These cells were embedded in an environment that spanned from mechanically rigid and flat (within 7 μm of the coverslip), to soft and 3D (greater than 7 μm away from the coverslip). Coverslip height was identified using a z profile of the nucleus intensity as described in [41]. Based on previous works on 3D cell shape analysis and collagen dispensed on rigid substrates, we treat these as two different environmental classes, namely proximal and distal [41].

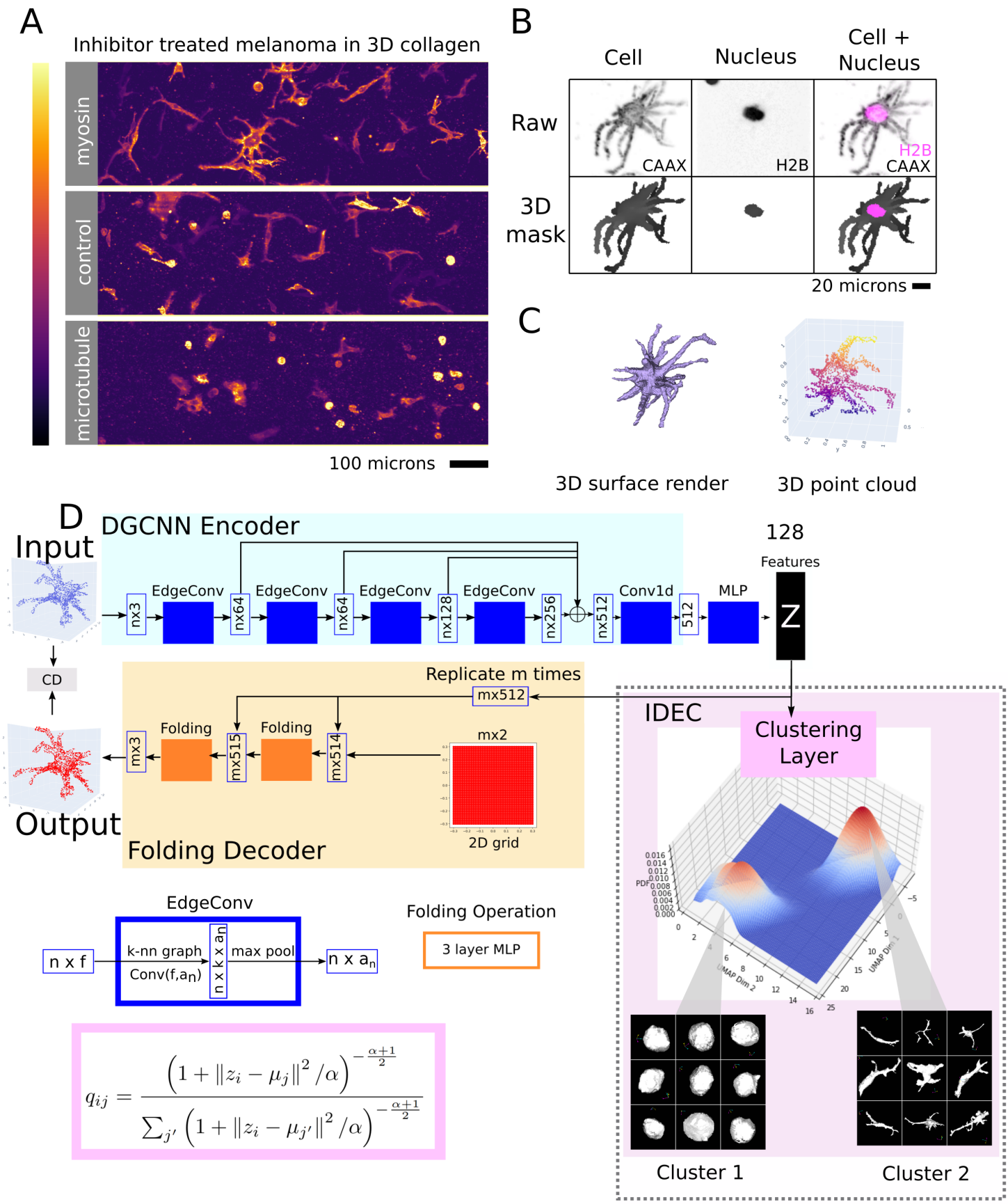
In order to focus on shape data, we created binary masks of each cell and nucleus by thresholding the whole volume by active contour segmentation [42]. Touching cells and nuclei were separated by the watershed-based methods described in [41] (Figure 1B). Our GDL method requires point cloud data. As such, 3D voxel masks were then converted to mesh objects using marching cubes and 2048 points were sampled from the surfaces of each mesh object to give a point cloud representation of each cell and nuclei (Figure 1C). We have created a Python package for point cloud generation called '`cellshape-helper`'. We have implemented a number of GDL algorithms for shape representation learning on point clouds and made these available as a python package called '`cellshape-cloud`'. Point clouds were pre-processed by mean centring and scaling to assist in training our GDL model.

An automated method to profile cell shape in 3D using geometric deep learning

Towards the goal of creating an automated method to profile single-cell shapes in 3D, we used GDL models. Our GDL model, dubbed Dynamic Graph Convolutional FoldingNet (DFN) (Figure 1D) (detailed description in), is a combination of three techniques and is intended to learn shape features and classes simultaneously. DFN consists of an autoencoder to learn shape representations in an unsupervised manner with an additional clustering layer that groups cells into classes based on these features. A point cloud representation of a cell or nuclei is passed into the encoder part of the autoencoder, this encoder part is known as a dynamic graph convolutional network [25] (light blue in Figure 1D). Here, the point cloud goes through a series of operations known as edge convolution. Edge convolution is an operation which takes in a point cloud, creates a k-nearest neighbour graph on that point cloud, and then performs 2D convolution on the edges or adjacency matrices of that graph and outputs a new lower-dimensional set of points. This part is called a dynamic graph convolution as a new graph is constructed at every layer of the network. The output of this encoder is a lower-dimensional representation, $\mathbf{Z}$, of the input point cloud. This lower-dimensional representation passes through two branches, each with different purposes. In order to learn representative features, we need to be able to reconstruct from this lower-dimensional space to the original point cloud. To do this, we use a decoder in the form of a FoldingNet [31] (light orange in Figure 1D). The FoldingNet takes in the lower dimensional representation concatenates it with points sampled from a 2D plane and performs multi-layer perceptron operations on this to output a reconstructed point cloud. This part is trained on how similar the input and output point clouds are. In order to learn shape classes, a clustering layer takes in the lower dimensional feature vector, performs k-means clustering to get an initial estimate of shape classes and then gives a probability score of how likely a particular feature representation of a 3D cell is to belong to a cluster or shape class (grey in Figure 1D). In order to avoid confusion, we refer to the k in the k-nearest neighbours graph algorithm as k_g and the k in the k-means clustering algorithm as k_c. Cluster centres and encoder weights are learned in a way which places most of the probability mass where the clustering layer is confident that certain cells belong to certain classes. Initially, the DFN was trained on the ShapeNet dataset for 250 epochs and then we continued training on our point cloud representations of both cells and nuclei for another 250 epochs using Adam optimiser with e^{-6} weight decay. We used a batch size of 16 with an initial learning rate of e^{-4} and an exponential learning rate decay scheduler. The DFN model was set to extract 128 features from each point cloud (see Methods Section).

Exploring the cell shape feature landscape

Our trained DFN model without the clustering layer was first used to extract 3D shape features from all cells in our dataset. We visualised the extent of the cell shape landscape by performing UMAP [43] on the extracted features to form a two-dimensional embedding space. We sampled two-dimensional locations along this projection and plotted



$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'} \left(1 + \|z_i - \mu_{j'}\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}$$

Figure 1: **Learning shape features and classes simultaneously through geometric deep learning and deep clustering.** (A) Single-cells embedded in a collagen matrix were treated with a variety of small molecule drugs and imaged in 3D in high-throughput by stage scanning oblique plane microscopy (ssOPM). Maximum intensity projections (XY view) of cells treated with DMSO (control), Blebbistatin (myosin) and Nocodazole (microtubule) are shown. Intensity is the grayscale intensity of the CAAX-GFP membrane marker. (B) The cells were labelled with transgenes expressing Histone-2B (which labels the nucleus) and CAAX-GFP (which labels cell membranes). Nucleus and cell label intensities (Raw) and their segmentation masks (3D mask) are shown for a single cell. (C) We used the marching cubes algorithm to create a mesh object of vertices, faces, and normals and then sampled 2048 points from this mesh object to create a point cloud for each cell cytoplasm and nucleus. The 3D surface render and 3D point cloud for a cell are shown. (D) A dynamic graph convolutional foldingnet autoencoder was used to learn a lower-dimensional representation of the input cell shape. This consisted of a Dynamic Graph CNN as an encoder (light blue) and a FoldingNet decoder (light orange). In order to learn representations and shape classes simultaneously, improved deep embedded clustering adds a clustering layer to the feature representations (light purple). This maps the representations of each cell to a specific shape class and gives a morphological signature for each cell in terms of the different shape classes that exist in the dataset. We show a dotted box around this component of the model as it is not always used and is an addition to the autoencoder.

rendered images of cell masks which generated the features projected in UMAP space (Figure 2A). Furthermore, we zoom in on locations where feature representations were dense to show common shapes that exist in the dataset. The first dimension of the UMAP seems to incorporate eccentricity, with round cells having low values for this dimension and more eccentric cells having high values. After visualising the extent of the cell shape landscape, we looked at the distribution of cells within this landscape. We estimated the probability density function using kernel density estimation of the UMAP embedding of the latent space (Figure 2B). This revealed three peaks representing common cell shapes or shape modes in our dataset. These peaks represented small round cells, large round cells, and a large group of cells that ranged from elongated to protrusive. A subset of cells in our dataset had been treated with small molecule inhibitors, including myosin and microtubule inhibitors. We visually assessed the effect of these on cell shapes in 3D. The cell shape feature landscape demonstrated that cells treated with different inhibitors cause different distributions of embedding (which represent shape distributions). For example, Control cells (Figure 2C) and cells treated with Blebbistatin (myosin inhibitor) (Figure 2D) are able to take on a larger range of shapes than cells treated with Nocodazole (microtubule inhibitor) (Figure 2E). Nocodazole-treated cell shapes are confined to rounder-like cells with few protrusive cells.

The cells in our dataset are embedded in a 3D collagen matrix at a variety of depths. To understand how differences in the physical environment influence cell shape we visualised our UMAP space based on the different environments. We coloured each embedding location in UMAP space according to the cell's distance from the coverslip (Figure 2F). This showed that cells close to the rigid coverslip (proximal) and cells embedded in collagen (distal) are separated in shape space and that the geometry and physical properties of the microenvironment have a major influence on cell shape. The distribution of cells proximal to the coverslip (fewer than 7 microns) showed two peaks or shape modes with one of these being more spread out than the other (Figure 2G). A similar analysis of the distribution of the UMAP embedding for cells distal to the coverslip (greater than 7 microns) showed one major density peak with three smaller peaks (Figure 2H). Interestingly, the embedding location of the greatest peak for distal cells is almost non-existent in cells proximal to the coverslip. It is also evident that the cells' environment impacts their ability to take on different shapes.

Deep embedded clustering learns shape features and classes simultaneously

Many studies have used deep learning to cluster or organise everyday 3D objects by their shape [31, 44, 45, 46]. These tasks use datasets that have distinct semantic (or human meaningful) classes. For example, the ModelNet [22] dataset includes chairs, aeroplanes, baths, beds, lamps and cars. In contrast, in problems of phenotypic classification within a dataset of cells, we do not have labelled phenotypes for every single cell. Furthermore, we did not know the number of semantic or phenotypic classes in our dataset, or even if any distinct classes exist. An interesting solution given this lack of prior knowledge has been to automatically learn feature profiles and classify images into classes without predefined labels [47, 48]. We wanted to explore the different classes of cell shape in our dataset and propose jointly learning feature representations and shape clusters, simultaneously.

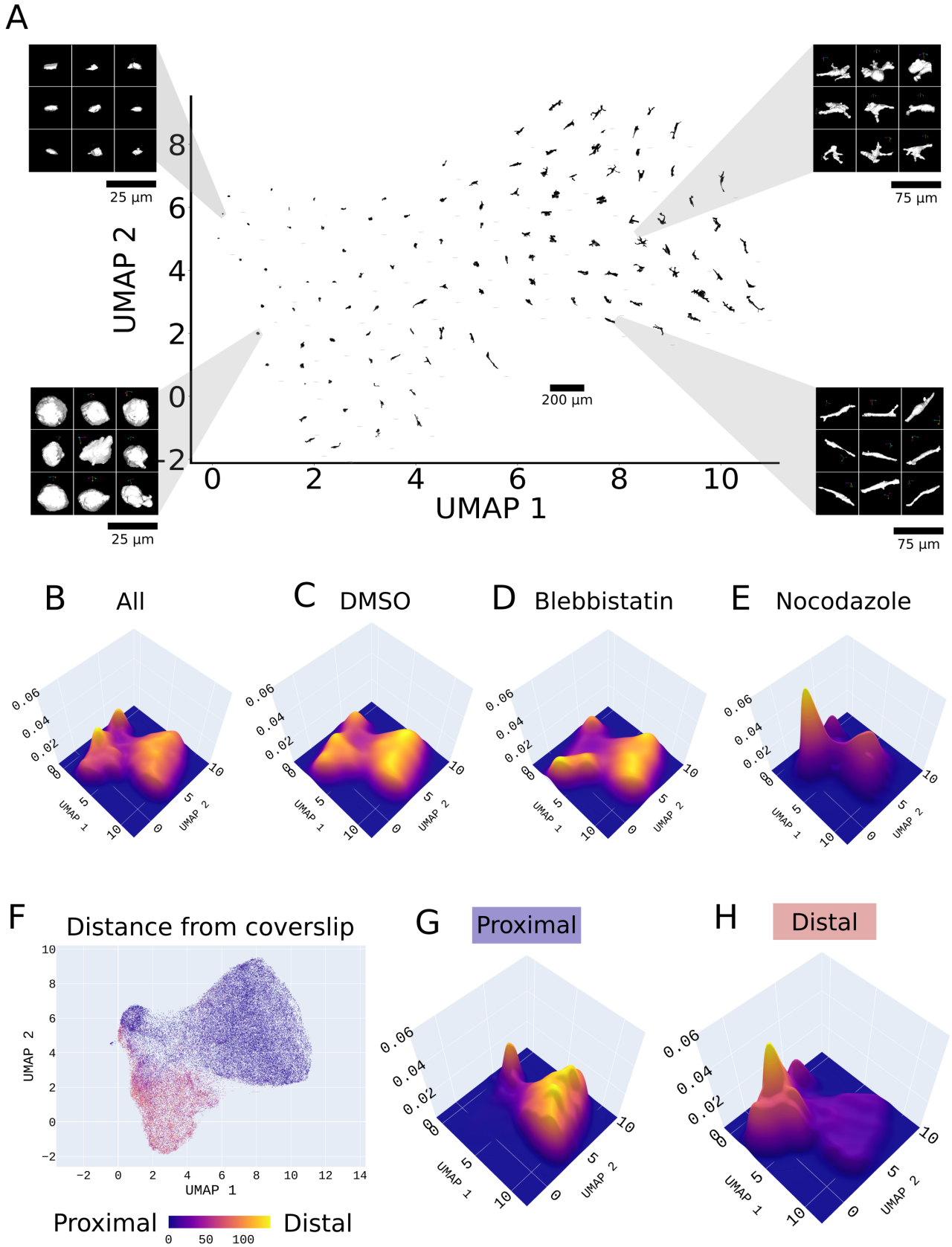

A
25 µm
75 µm
UMAP 2
UMAP 1
200 µm
25 µm
75 µm
B All
C DMSO
D Blebbistatin
E Nocodazole
F Distance from coverslip
UMAP 2
UMAP 1
Proximal
Distal
G Proximal
H Distal

Figure 2: **Dynamic graph convolutional foldingnet reveals the cell shape landscape in a cancer cell line.** We performed UMAP on the 128 features extracted using the DFN without the addition of the clustering layer. (A) Rendered sample images of cell masks are plotted in UMAP. (B-E) The probability density of the UMAP was estimated using kernel density estimation. DMSO treated cells (C) had different UMAP distributions than Blebbistatin (D) treated cells and Nocodazole (E) treated cells. (F) The points on the UMAP were coloured by their distance relative to the coverslip. The microenvironment impacts the ability of cells to take on different shapes. (G) Cells on the coverslip tend to be flatter and more spread out. (H) Cells embedded in collagen (further away from the coverslip) are more round.

After training the DFN autoencoder, we performed improved deep embedded clustering (IDEC) [49] by adding a clustering layer to the output of the encoder part. This clustering layer (grey in Figure 1D) takes the embedded feature space, $\mathbf{Z}$, as input and outputs a soft-label over the shape classes found according to a Student's t-distribution (purple box in Figure 1D). The soft-label is interpreted as the probability of assigning an input cell i to shape cluster j. For example, if there are k_c clusters in the dataset, each cell is assigned k_c soft-labels (one for each cluster) and each soft-label contains scores from zero to one reflecting how similar the cell is to each of the different cluster centres (with one being most similar)(Figure 3A). Since the 3D shape of cells in a fixed assay is a relatively continuous distribution with an indefinite number of distinct classes (cells may be very similar in shape, yet almost never the exact same), it is more natural to describe a shape by a signature of how similar it is to a number of different exemplar classes. For example, we may wish to describe the 3D shape of a cell to be 30% similar to shape cluster 0, 10% similar to shape cluster 1, 5% similar to shape cluster 2 for shape classes $i = 0, 1, ...k_c$, such that the sum of similarity percentages add to 100 ($\sum_{i=0}^{k_c} k_c^{(i)} = 100$, where $k_c^{(i)}$ is shape class i). This can be defined as a quantitative morphological signature for the 3D shape of a cell (3DQMS). Therefore, our k_c-dimensional vector of soft-labels can be used as a 3DQMS. The number of clusters was chosen as a hyper-parameter. However, this could also be estimated by prior knowledge, or through more formal methods such as the elbow, average silhouette or gap statistic methods when undertaking the initial k-means clustering. The clustering layer is packaged in a tool called '`cellshape-cluster`' and can easily be imported into our '`cellshape-cloud`' autoencoder models.

For our experiments, we selected five clusters and the DFN-IDEC output a soft-label for each cell of the five shape classes found. These five clusters approximated the five most different groupings of 3D cell shape features in our dataset. We sampled from the dataset for cells which had the highest values of soft-label for each of the five classes to see classes ranging from round, to elongated, to protrusive and block shapes cells (Figure 3B). Using this soft-label, we were also able to examine the 3DQMS for cells treated with different inhibitors. We can further inspect how Blebbistatin and Nocodazole-treated cells make up the different clusters by projecting the features using UMAP and fitting a kernel density on this UMAP projection (Figure 3 C), similar to Figure 2. Cells treated with Blebbistatin explore four of the five shape clusters (Figure 3C), labelled 1-4 in Figure 3A. These shape clusters are large and protrusive when compared with shape cluster 0 which is the round cluster. Nocodazole-treated cells tend to mostly fall within shape cluster 0 (Figure 3I). Next, we standardised and averaged the 3DQMS for each cell by their treatment and performed hierarchical clustering on these to find four classes of treatment effects on shape (Figure 3D). We noted that two distinct inhibitors of myosin activation (Blebbistatin and H1152) grouped together, indicating that the 3DQMS will be a powerful tool for using cell shape to identify drugs that act on similar pathways. The other classes of treatment effects on shape included Nocodazole, which inhibits the polymerisation of microtubules, CK666, which prevents actin branching and then control cells grouped with those treatments that do not have significant effects on cell shape. To visualise the variability in 3DQMS within each treatment and differences between treatments, we took the average 3DQMS for each well and embedded these in two dimensions using UMAP (Figure 3E). This again showed the two distinct myosin inhibitors grouped together.

When clustering on the shape classes (hierarchical clustering on the columns of Figure 3D), we saw that shape cluster 0, which is the first column in Figure 3D, was the main driver of variation between the treatments. Thus, we examined the soft-label distribution for shape cluster 0 in these four main classes of treatment effects on shape (Figure 3F). Cells treated with Blebbistatin tended to have more cells with low-scoring soft-labels in cluster 0 (Figure 3F), compared to the DMSO and CK666 treated cells (Grey box in Figure 3F and F, respectively). In contrast, cells treated with Nocodazole tended to have much higher soft-label scores for cluster 0 (Figure 3F). A cell's environment impacts the shape space, as seen in Figure 2 , and thus affects the 3DQMS of different treatments. For cells distal to the coverslip, the distribution of soft-label scores of cluster 0 for cells treated with Blebbistatin tends to flatten out slightly (Magenta box in Figure 3F). All distributions shift to the right (higher scores for shape cluster 0) for distal cells (Magenta box in Figure 3F*ı*), whilst this distribution shifts to the left for proximal cells across all treatments (Purple box in Figure 3F). Cells treated with Nocodazole that were distal to the coverslip mostly scored very high for shape cluster 0 (Magenta in

Figure 3F). Finally, the feature landscape is affected by the clustering layer such that each peak roughly represents a shape cluster (Figure 3A).

Learned shape features predict cell treatment

We explored the possibility to distinguish between cells with different small molecule inhibitor treatments based purely on cytoplasm and nuclei shape features. Many studies in computer vision and 3D shape representation learning test the effectiveness of their representations with transfer classification tasks. This involves using a trained model to extract features from a dataset and then using these features to train a linear support vector machine (SVM) classifier [31, 44, 45, 46]. These tasks generally involve classification between objects from distinct semantic classes or categories, for example, cars, chairs, and aeroplanes [22]. Our dataset is comprised of a single cell type and did not have such distinct semantic classes, so to test our shape representations we predicted our drug treatment labels. Following similar procedures to previous researchers [31, 44, 45, 46], we used the trained DFN to extract features from each cell. These features were then used to train a linear SVM. For all experiments, we use one-versus-one SVM classifiers with an L2 penalty (C = 1), balanced class weights, and intercept scaling during training. We tested cytoplasm features and nuclei features alone and then combined them for ablation studies (Figure 4A). We performed a 10-fold cross-validation for each experiment and report mean accuracies.

To compare the relative importance of cell and nuclear for predicting drug treatments we tested the accuracy of classifiers using only cell features, only nuclear features or the combination of both cell and nuclear feature sets. We also analysed prediction accuracies for different environments (Figures4 B and C). To do this comparison we tested one-versus-one classification accuracies between Blebbistatin and Nocodazole-treated cells. We used these treatment sets because our deep clustering of our dataset showed that these two treatments had the most different 3DQMS profiles.

When using cytoplasm shape features alone, we were able to predict the cells' treatment with an accuracy of 79% for cells proximal to the coverslip, 82% for cells distal to the coverslip and 80% for all cells regardless of the microenvironment. This is compared to the 74% accuracy when using shape features extracted from the nucleus only for proximal cells, 76% for distal cells and 74% for all cells. When we combined these two sets of shape features, accuracies increased to 86%, 82% and 83% for proximal, distal and all cells respectively. This difference in accuracy between combined and individual data sets suggests that not only are the shapes of the cell and nucleus affected by these treatments but so too are the relationship between them (Figure 4A).

Based on the combination of cell and nuclear features giving the best prediction accuracy we tested the prediction of all of our treatments in a pairwise fashion (Figures 4B and C). Due to the strong influence of environment on shape (Figure 4A), we made these predictions separately for cells in soft collagen environments (distal to the coverslip), and cells in rigid environments (proximal to the coverslip). In the proximal setting, we saw a general trend for higher accuracy of one-versus-one predictions (Figure 4B) than those in the distal setting (Figure 4C). When looking at all cells in both settings, we see accuracies between the two settings (Figure 4D). We also visualised the difference between the two settings directly as the difference between 2D and 3D prediction accuracies (Figure 4D). For distal settings cells treated with Nocodazole, Blebbistatin and H1152 all had one-versus-one classification accuracies across all other treatments higher than 62%. On average, cells treated with Nocodazole had the highest classification accuracies (Figure 4C). We noted that Blebbistatin and H1152 were both distinguishable from control (66% for H1152 vs No Treatment and 70% for Blebbistatin vs No Treatment in both environments, shown in Figure 4D) but difficult to distinguish from one another (56% accuracy).

In the distal setting, many of the remaining treatments had one-versus-one classification accuracies of less than 0.6, indicating they occupy a similar region of shape space. Unexpectedly, we did notice that Binimetinib compared to our "No Treatment" control was as equally distinguishable as the arp2/3 inhibitor, CK666 (equally 63% in proximal cells, 62% and 61% in distal cells, and equally 61% when looking at all cells). This was notable because Binimetinib is a MEK1/2 inhibitor without major connections to regulating the cytoskeleton, whereas the primary effect of CK666 is to prevent the formation of branched actin structures.

Consistent with the distal setting, Nocodazole, Blebbistatin and H1152 had the highest one-versus-one classification accuracies in the proximal setting (Figure 4B). Also consistent with the distal setting, we saw that the targeted therapy Binimetinib was as distinguishable from controls as CK666. Strikingly the classification accuracy of Binimetinib

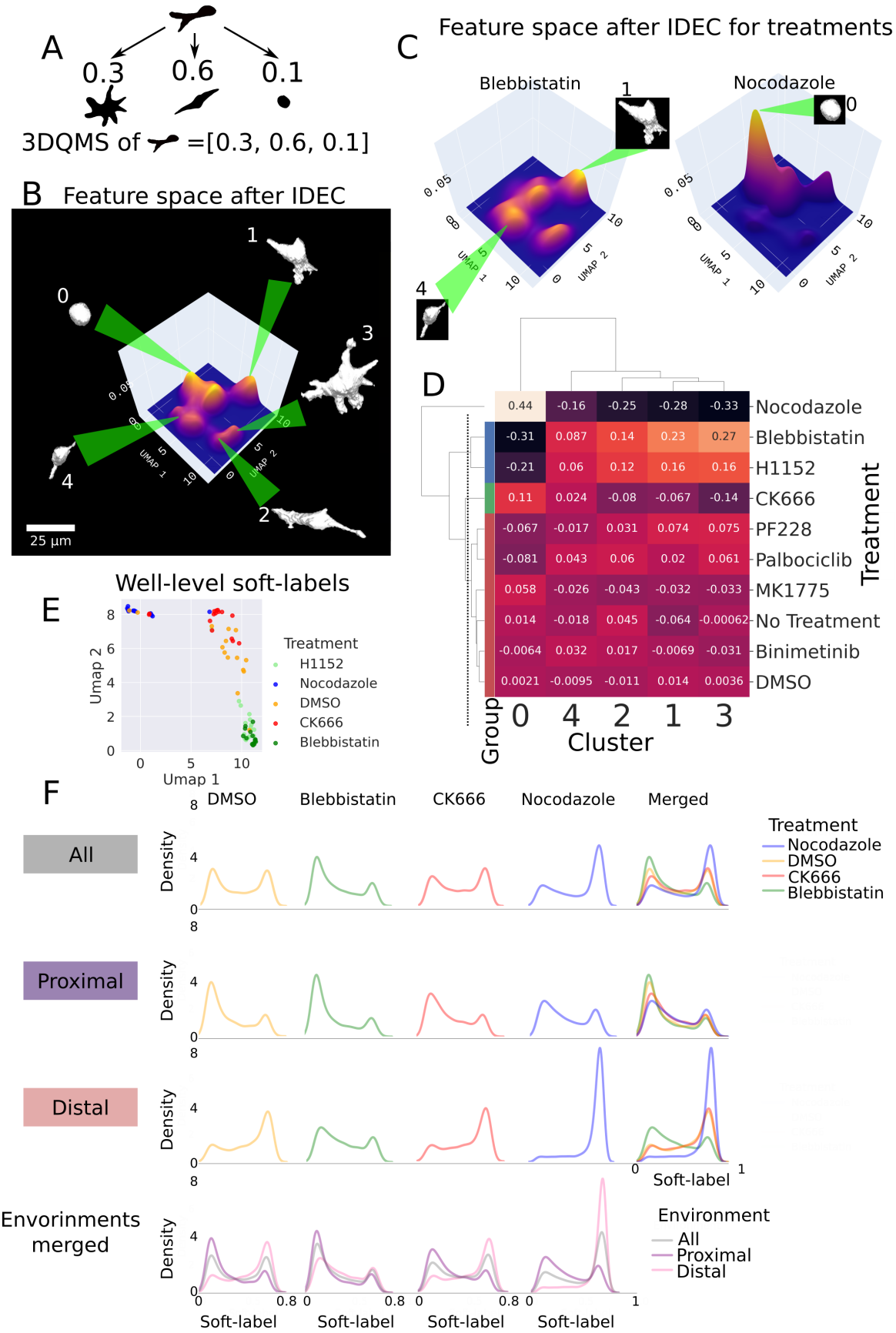

A
0.3
0.6
0.1
3DQMS of =[0.3, 0.6, 0.1]
B Feature space after IDEC
25 µm
C Feature space after IDEC for treatments
Blebbistatin
Nocodazole
D
Nocodazole
Blebbistatin
H1152
CK666
PF228
Palbociclib
MK1775
No Treatment
Binimetinib
DMSO
Treatment
Group
Cluster
E Well-level soft-labels
Umap 1
Umap 2
Treatment
H1152
Nocodazole
DMSO
CK666
Blebbistatin
F
DMSO
Blebbistatin
CK666
Nocodazole
Merged
All
Proximal
Distal
Envorinments merged
Density
Soft-label
Treatment
Nocodazole
DMSO
CK666
Blebbistatin
Environment
All
Proximal
Distal

Figure 3: **Improved deep embedded clustering learns quantitative morphological signatures for 3D shapes of cells.** We added a clustering layer to a trained DFN model to learn the shape clusters that exist in the dataset. (A) Schematic example of Improved deep embedded clustering (IDEC) soft label assignment to a cell in the case of three clusters. (B) The clustering layer outputs a distribution of soft-labels across the five classes. We show the five exemplar shape classes found and the kernel density approximation of the features extracted. (C) Shows the UMAP projection of the features extracted from the DFN-IDEC model for Blebbistatin and Nocodazole-treated cells. (D) Each cell is assigned a 3DQMS which is a five-dimensional vector that describes how similar it is to each shape cluster. We standardised and averaged the 3DQMS for each treatment and performed hierarchical clustering to reveal four classes of the treatment effects on cell shape. The dotted line shows the threshold for hierarchical clustering. (E) Shows a UMAP projection of the 3DQMS for different treatments. (F) We show the distribution of soft-label scores for cluster 0 for Blebbistatin, DMSO, CK666, and Nocodazole-treated cells respectively (All). Similarly, we show this for cells proximal to the coverslip (Proximal). (D"-H") And for cells distal to the coverslip (Distal).

was increased in the proximal setting (Figures 4B, C and E), suggesting that the shape effect of the MEK inhibitor is more pronounced in rigid environments. This does not appear to be a general effect of the cell cycle inhibitory effect of Binimetinib, for two reasons. First, cells treated with the CDK4/6 cell cycle inhibitor, Palbociclib, were not readily distinguishable from control. Second, In contrast to Binimetinib, there was no obvious interaction between shape and environment on the effect of treatment with Palbociclib (Figure 4D). Taken together, the lack of general effect of cell cycle inhibition suggests components of the cytoskeleton may be direct targets of MEK1/2 phosphorylation.

Finally, following the observation that Blebbistatin and H1152 were distinct from control but hard to discern from one another, we developed a classification "interference test". To do this, we measured the change in the correct classification of Blebbistatin-treated from control cells, in three-way comparisons between DMSO, Blebbistatin and each other treatment. We found that in one-versus-one comparisons, the SVM classified Blebbistatin correctly from a pool of Blebbistatin and control cells with an accuracy of about 65% (65% of true Blebbistatin-treated cells were predicted as Blebbistatin). This is different to the accuracies reported above as those were the accuracies for both treatments involved in the classifier. This accuracy showed little change in three-way comparisons between Blebbistatin, DMSO and iterations of most other treatments. For example, introducing MK1775 into the classification (Blebbistatin:MK1775:DMSO) had little effect (0.02 interference), but introducing H1152 (Blebbistatin:H1152:DMSO) had a large impact on classification accuracy (0.18 interference) (Figure 4E). Blebbistatin and H1152 act at separate points in the Rho-Rock-Myosin II activation cascade, indicating that our extracted shape features and classification interference test can be used to group drugs with similar mechanisms of action (Figure 4F).

Interpreting the learned features

Feature interpretation allows biologists to inspect DL models. Ultimately we want to know what the models are seeing in the data and learn from them. DL models may be learning redundancies or focusing on information that is not important to the task at hand. Alternatively, the models may learn something which is not already known, thus providing new information on possible biomarkers.

Given that the DFN is able to automatically extract features which classify myosin and microtubule inhibitors based on cell shape (Figure 4), we sought to interpret the cell shape features learned by our method. These methods can be easily extended to nuclei shape features. We interpreted our features depending on two criteria, first we discovered intuitive explanations for the DL features that were the most powerful drivers of variation in cell shape space by relating them to manually crafted, 'classical' features.

Our second approach was a task-specific approach, where we decoded the DL features that contributed the most to successful discrimination between cell treatments in different classification tasks (see Methods Section). Here, we compared features important for classifying Blebbistatin versus Nocodazole-treated cells.

Naturally, there existed a correlation between the DL features. Therefore, we performed principal component analysis on the DL features to obtain 128 principal components (PC) ordered by the magnitude of their singular values. We then correlated the principal components with our classical features of geometry (Figure 5A). Green boxes in Figure 5A show the classical features with the highest absolute correlation to each PC. PC0 encodes many of the classical features

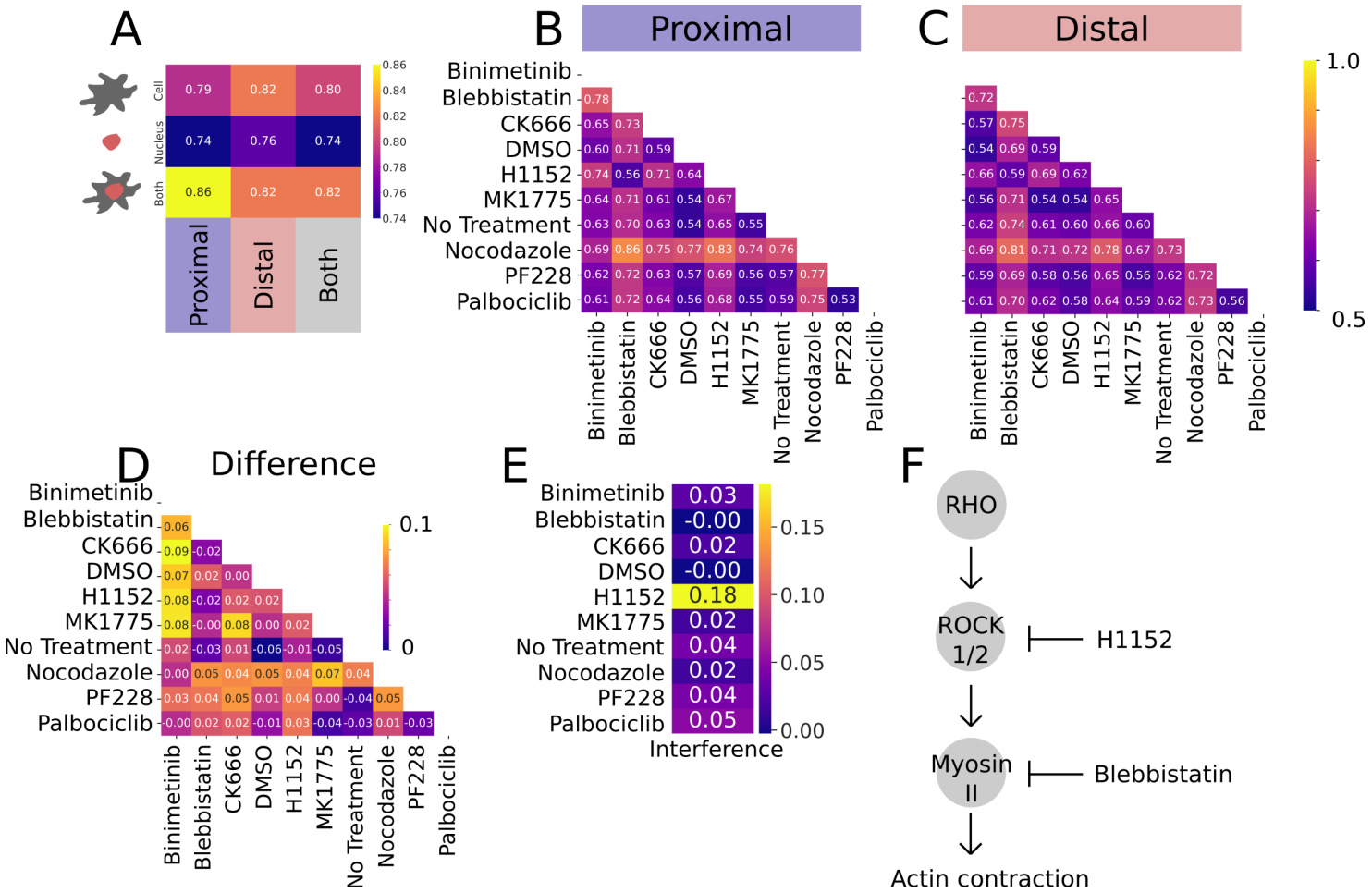


Figure 4: **Features extracted using the DFN accurately predict small molecule treatments.** (A) Features were extracted from the cytoplasm, and nuclei of cells treated with Nocodazole and Blebbistatin and used to train a linear support vector machine (SVM) to predict treatment. We tested the different feature sets using cytoplasm features only, nucleus features only and combining these two. We also tested predictions in different environments from cells proximal to the coverslip, to distal from the coverslip to all cells (both proximal and distal). (B-C) We used an SVM to predict treatment for all cells using the combined cell and nucleus feature set in (B) proximal cells, and (C) distal cells. (D) Shows the difference in accuracies between proximal and distal cells. (E) Shows the interference effect of each treatment on the accurate classification of Blebbistatin when compared to DMSO. The treatment with the strongest interference effect was H1152, which acts above Blebbistatin in the RHO-ROCK1/2-MyosinII pathway.(F) A schematic of the RHO-ROCK1/2-MyosinII pathway and the target of Blebbistatin and H1152.

with major axis, surface area and sphericity being the highest. PC2, which explains 10.85% of the variation in the data, has a maximum correlation coefficient of 0.14 with classical features (minor axis) and does not correlate highly with any classical features suggesting that the DFN may be capturing features which are not explained by our classical features. The first two PCs account for 42.47% of the variation in the data with the first 10 PCs accounting for 82.24%.

We assessed how the treatments differ in terms of their PC loadings by taking the average PC loadings for cells treated with Blebbistatin, Nocodazole, H1152, CK666 and DMSO (control) (Figure 5B). For most of the first 10 PCs, we see that cells treated with Blebbistatin and cells treated with H1152 (Blebbistatin and H1152 which both act in the Rho-Rock-Myosin activation cascade) have similar average PC loadings. Interestingly, cells treated with Nocodazole and cells treated with CK666 have similar signs of average PC loadings for most PCs. These two groups (Blebbistatin and H1152 being one and Nocodazole and CK666 being another) tend to have opposite average PC loadings.

The second, task-specific approach, revealed that PC0, PC3, and PC9 were most important in classifying between Nocodazole and Blebbistatin treated cells (See Methods Section). Similar to methods in [50], the correlations shown in Figure 5A are visually confirmed by sampling cell shapes from the dataset with a range of values for different principal components (Figure 5C-E shows this for PC0, PC3, and PC9). To assess the relationship between classical features and the PC of the DL features, we fit local regression curves using weighted linear least squares regression (lowess) (Figure 5C-D). We calculated confidence intervals around the lowess fits by using bootstrapping to provide an estimate of the spread of the curve. We show the 65% confidence intervals.

The first PC, PC0, represents a combination of classical features including the major axis, surface area, and the sphericity of the cell. We see a positive linear relationship between PC0 and both surface area and major axis when fitting local regression curves between PC0 and these features and a negative relationship between PC0 and sphericity (Figures 5C). PC3 is most correlated with axial extent, distance from the coverslip and the minor axis. These relationships are more complex (Figure 5D). Similarly, PC9 has complex relationships to the volume to surface area ratio and the equivalent diameter of 3D cells (Figure 5E).

PC1, which explains 18.67% of the variation of the DL features correlates most with roll which is an orientation parameter representing the angle of rotation when looking at the origin along the x-axis (calculated using MATLAB's regionprops3 function). The orientation of cells, whilst important to geometry, may not be of interest in some biological contexts. As a step toward a rotationally invariant method to extract 3D shape features, we used a "pose correction" approach to first align the cells before extracting the features (Section). Cells were pose corrected to align them according to their principal axes and obtain canonical poses (See Methods Section). Figure 5F and G show two cells before and after alignment respectively.

We visually expressed how DL features may capture classical features by performing UMAP on the DL features and colouring each point by the corresponding classical feature value. We do this for DL features extracted from both unaligned (Figure 5H-H$''$) and aligned cells (Figure 5I-I$''$). This revealed a clear pattern of organisation of roll values suggesting that this feature is being incorporated by the DL features of the unaligned dataset (Figure 5H). However, after pose correction to align the cells by their principal axes, the effect of roll was mitigated (Figure 5I). Figure 5H$'$-H$''$ show how classical features; sphericity, eccentricity and polarity, are all incorporated by the DL features. Similarly, after aligning the cells, we still see these features being incorporated (Figure 5I$'$-I$''$).

Discussion

Deep learning has been applied in the context of 3D microscopy for tasks of segmentation [51, 52, 53], restoration and denoising [54], and classification. However, few methods explore the possibility of utilising deep learning to learn the shape feature profiles of single-cells in 3D. We have developed an automatic tool to simultaneously learn shape features and shape classes of 3D objects and applied this to a metastatic melanoma cell line. Our method assigns 3D quantitative morphological signatures to each cell. This signature measures how similar the shape of each cell is to the shape classes found in the dataset. We used these extracted features to distinguish between cells treated with inhibitors of myosin and microtubules and show that these inhibitors are environmentally dependent. We were also able to use pose correction as a practical method to remove learning of rotational variance. Finally, we provide intuitive explanations of the learned features by correlating them to classical measures of geometry and by visualising uncorrelated embeddings of these features (in the form of principal components). This showed that our model automatically learns features which capture the traditional measures of geometry, and the low correlation of our second principle component with traditional measures suggested we are capturing information that is previously missed.

Building a deep learning model requires design choices about input data, model architecture and hyperparameters. We discuss this in more detail to allow repeatability and improvements to our methods in further studies.

The first decision is the choice of input data. Since the creation of 3D object datasets such as ModelNet [22], the two broad approaches which have been used to represent the input data are to use voxels (the 3D counterpart to pixels in 2D images), or point clouds. Recently point clouds have been the dominant approach [38, 28, 55]. One reason for this is practical. In particular, point cloud data is abundant and is the type of data generated for raw object surfaces when using devices like LIDAR sensors. Additionally, point cloud data is a very simple 3D representation of 3D shape, consisting of a 2D matrix of positions of points. We had the choice of both voxel and point cloud representations and tried both approaches when representing human cells. For learning representations on 3D voxel grids, we utilised 3D ResNet

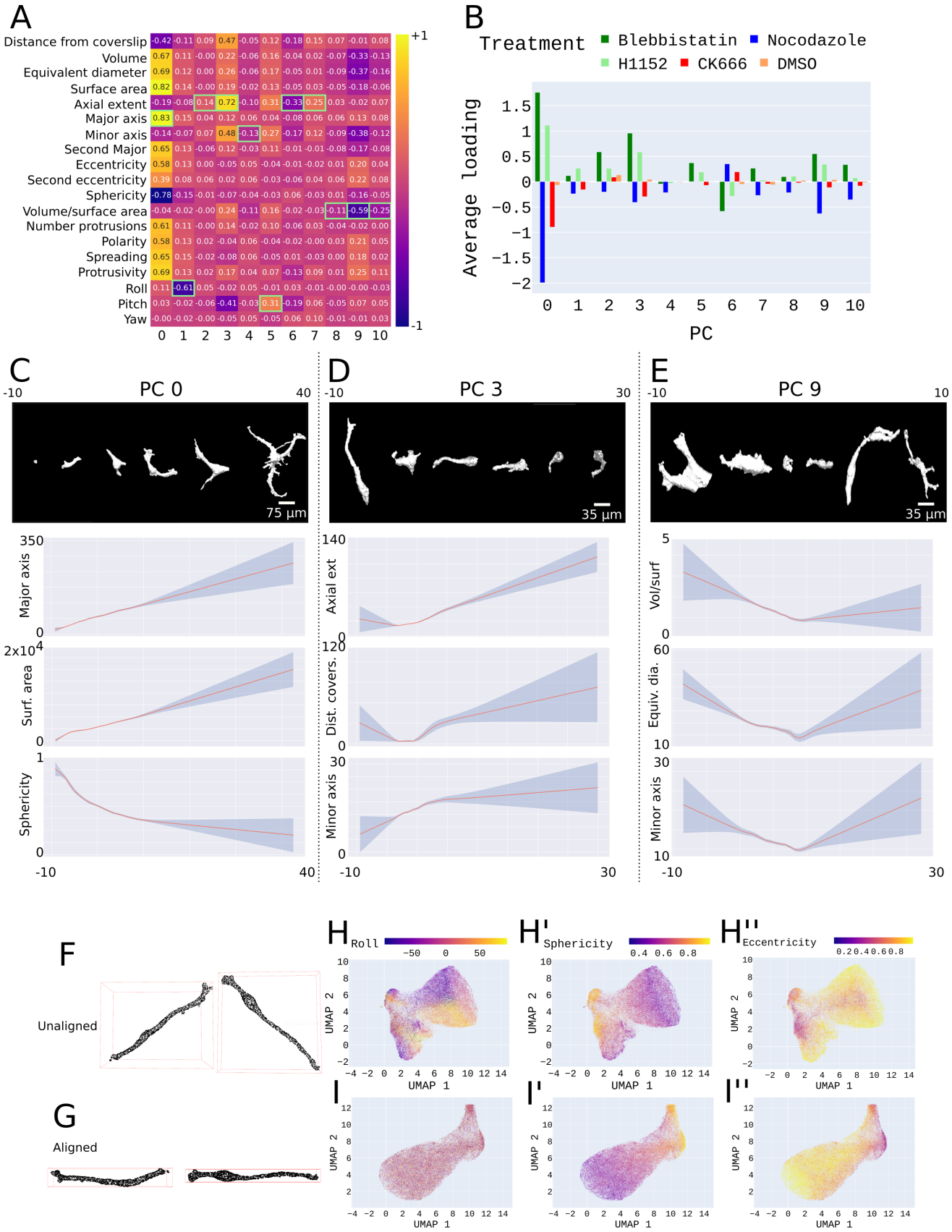

A
Distance from coverslip
Volume
Equivalent diameter
Surface area
Axial extent
Major axis
Minor axis
Second Major
Eccentricity
Second eccentricity
Sphericity
Volume/surface area
Number protrusions
Polarity
Spreading
Protrusivity
Roll
Pitch
Yaw
0 1 2 3 4 5 6 7 8 9 10
+1
-1
B
Treatment
Blebbistatin
Nocodazole
H1152
CK666
DMSO
Average loading
PC
C
PC 0
75 µm
D
PC 3
35 µm
E
PC 9
35 µm
Major axis
Surf. area
Sphericity
Axial ext
Dist. covers.
Minor axis
Vol/surf
Equiv. dia.
F
Unaligned
G
Aligned
H
Roll
H'
Sphericity
H''
Eccentricity
I
I'
I''
UMAP 1
UMAP 2

Figure 5: **Interpreting learned features and a step towards rotational invariance with pose correction.** (A) Pearson correlation coefficients between principal components of extracted cell shape features and classical features. Green boxes show the classical features with the highest absolute correlation to each principal component. (B) Average PC loadings for cells treated with Blebbistatin, Nocodazole, H1152, CK666, and DMSO. (C-E) Interpreting the cell shape features extracted from the DFN, by continuously varying PC (PC0, PC3, PC9) values. Images are sampled from binned encodings of the respective PC (PC0, PC3, PC9) value. We show a lowess fit with a 65% confidence interval between (C) PC0 and major axis, surface area (Surf. area), and sphericity. (D) Shows the sampled images for the range of PC3 values. Panels show lowess fit with a 65% confidence interval between PC3 and axial extent, distance from the coverslip (Dist. covers.), and minor axis. (E) Shows the same for PC9- the lowess fit of PC9 with volume to surface area ratio (Vol/Surf), equivalent diameter (Equiv. dia.), and Minor axis. (F) Before pose correction to align cells to their principal axis. (G) After pose correction to align cells shown in (F). (H-H′′) Scatter plot of UMAP embedding of cell shape features on unaligned data with each point coloured by their corresponding roll, eccentricity, sphericity, and polarity value. (I-I′′) shows the same for the aligned cells.

architectures and subsequently packaged our methods as a Python package called '`cellshape-voxel`' for researchers to use freely. Using performance on classification tasks as a guide, we found that autoencoders using point cloud data were superior and have reported on these results. Our choice of 2048, as the number of points, was based on previous works [31, 25]. Future work could explore methods of describing cells by either a few critical points or alternatively, a constant density of points. For example, a cell could be described by its centre of mass and points along its protrusions. While decreasing the detail by describing cells by much fewer points, computational efficiency would significantly increase.

A second key design decision is the selection of an autoencoder. As previously discussed, geometric deep learning is the branch of deep learning that deals with graph or manifold (unstructured) data such as graphs created on point clouds. Geometric deep learning has predominantly been used on point cloud data to produce state-of-the-art results for classification and representation learning tasks [38, 40]. Our model incorporated edge convolution [25] as the primary operator in our encoder part of our autoencoder as this has proved successful in representation learning tasks [55]. FoldingNet is a novel folding-based decoder designed to assist in representation learning on point clouds [31]. This decoder backbone has been used extensively in the literature primarily as it is simple, and has shown promising results across several tasks [32, 40]. Thus, we incorporate a folding-based decoder in our model. Looking beyond the shapes of cells, future research could utilise our models to explore the shapes or networks of collections of cells in 3D cancer models such as tumour spheroids or organoids. Each cell in these populations of cells could represent a point with the entire population being represented as a point cloud. We can then use our methods to extract features from these point cloud representations of cells in populations, describing 3D cancer models as networks.

A third important decision is a method to classify unlabelled data into classes. This is a challenging task with methods proposed for common 2D benchmark datasets [56, 48]. To solve this problem for biological cells, we took the approach of deep embedded clustering. Deep embedded clustering is a way to simultaneously learn feature representations and cluster assignments or classes. This method involves clustering with Kullback–Leibler divergence on feature representations from an autoencoder [57]. This method can easily be incorporated into any autoencoder architecture and offers an interpretable output in the form of a soft-label which describes the probability of assigning data input to each cluster class. This, in turn, allows the assignment of data input to be continuous across classes rather than discrete. Cell shape morphology is a continuous variable with no two cells being identical in shape, and rather similar to each other or similar to exemplar shapes [58]. Describing cell shapes by a continuous similarity score to exemplar shapes offers a more natural solution than discrete binning into certain shape classes [8]. Thus, we have opted for deep embedded clustering to learn the 3D quantitative morphological signature (3DQMS) of cell shapes. Using these 3DQMS, we found four groups of treatment effects on the shape which grouped two distinct inhibitors of myosin activation (Blebbistatin and H11522). This showed that our 3DQMS will be a powerful tool for using cell shape to identify drugs that act on similar pathways.

Finally, the choices of hyperparameters are critical decisions in any deep learning pipeline. These include training configurations such as batch size, and learning rate, as well as optimisation algorithms and model architectural hyperparameters. These include the number of clusters in clustering algorithms, the number of neighbours to use when constructing k-nearest neighbour graphs, and the number of features to extract. Hyperparameters are commonly chosen through trial and error, or grid searches across a range of combinations with the best combination chosen according to

performance on the task at hand. We chose our hyperparameters by trial and error on transfer learning classification on the ModelNet40 dataset, with guidance from previous work on which our model is based [31, 25].

A major theme in recent research into 3D shape representation is how to deal with differences in the rotation of objects with an otherwise similar geometric form [32, 59]. Orientation, composed of measures of roll, pitch and yaw, is generally considered part of the formal geometric description of an object. In some settings, a focus on orientation is useful, but in others it is undesirable. For example, earlier attempts at distinguishing everyday objects found that accurate classification of cars can drop from 70% to 20% depending on the orientation [59]. In these settings where the task is the classification of 3D objects based on shape, we generally do not expect an object to change its identity depending on its orientation. This means that for classification tasks, sensitivity to orientation is likely to be undesirable. However, there are settings where orientation is helpful. For example, in biological contexts like embryonic and postnatal development, the establishment of planar cell polarity results in the alignment of stereocilia. Orientation is a meaningful feature for the study of the shape of stereocilia in development. Similarly, orientation can reveal important information about a cell's microenvironment, for example, alignment in the orientation of cancer cells migrating can indicate polarisation of the extracellular matrix. Because orientation can be important and useful, our feature extraction captures orientation by default (Figure 2 and Figure 5). However, we note there are also many applications in biology where orientation is not of interest. For example, orientation may not be of interest when studying the shapes of human cells in suspension, or free-living marine organisms. In this study, we addressed this using pose correction techniques based on works in [60] (Figure 5). It is worth noting that pose correction caused a decrease in the extracted features' ability to distinguish between treatments. Pose correction has been used in 2D shape analysis in biology [11], and has been a standard approach for circumventing the confounding effect of rotation in 3D shape datasets [22]. Recently, a number of groups have also developed rotationally invariant methods, which are directly incorporated into their models, to learn 3D shape representations [32, 59] that could also be applied to cells.

Explainability is an important feature to consider when building deep learning models in any application, especially in medical research. We have considered ways of interpreting learned features, however, other methods may prove beneficial. When dealing with 2D pixel data of images, applications exist that overlay heatmaps to visually express where the model is looking when making certain decisions or classifications [61]. These have recently been extended to deal with 3D voxelised data and are commonly used in medical image analysis tasks [62]. These methods have been explored for use on point cloud data [63] showing that the methods may be reliable on point cloud data. Future work could incorporate class activation maps on point cloud data of cells to visually represent why certain cells are being grouped together.

While our problem is fundamentally focused on an unsupervised feature representation of 3D cell shape, we have applied it in the context of cells treated with different small molecule inhibitors. We have taken advantage of this by using features automatically extracted using our DFN model, to train a linear support vector machine (SVM) that could distinguish between cells treated with myosin inhibitors from those treated with microtubule inhibitors with accuracies up to 86%. Furthermore, we showed how both cell and nuclei shape information is important in combination to predicting treatment, suggesting that these drugs not only affect cell shape and nuclei shape individually but also the relationship between the shapes of these components. Finally, we found that predicting treatment from cell shape was dependent on the environment with 86% accuracy between Blebbistatin treated cells in a proximal setting and 82% in distal settings. Future work could explore supervised techniques to classify the treatment of a cell directly. Finally, treatments may have a significant effect on shape, but the temporal aspect of the treatment effect is not captured in a dataset of fixed cells. Understanding the temporal shape signatures or patterns in cell shape dynamics could allow a more accurate understanding of the effect of treatment on cell shape. Extending these methods to incorporate time-lapse sequences of cells could be done by utilising recurrent neural networks on sequential shape features extracted.

Finally, a major component of our work is its accessibility to the wider biological and medical community. Packing our methods in open source Python packages with adaptability and ease of use allows further research on connecting 3D shapes to function across a variety of domains. Our Python package offers adaptability and growth with the plan to implement new methods continuously.

Methods

Melanoma cell preparation

Cells used in this study were WM266.4 harbouring CAAX-EGFP (donated from the Marshall lab), with the addition of an ERK-KTR-Ruby construct (addgene #90231) and a Histone2B-iRFP670 construct (addgene #90237).

Collagen preparation

Collagen hydrogels were prepared to a final concentration of 2mg/mL. Briefly, hydrogel solutions of dH_2O, 5xDMEM, HEPES (7.5 pH), and Rat Tail Collagen IV (Corning) were prepared on ice to a final collagen concentration of $2mg/mL$. Cells were re-suspended in hydrogel solution at a concentration of 4 E4 cells per 100 μl, and 100 μl of this solution dispensed into each experimental well on a 96 well plate. After dispensing cells, the plates were incubated at 37 degrees Celsius for 1 hour, and 100 μl of DMEM was added to each well.

Treatments and cell fixation

Treatments were added to cells 24 hours after seeding in collagen hydrogels. After 6 hours of treatment, cells were fixed in 4 percent paraformaldehyde for 30 minutes at room temperature. Final concentrations for treatments were: Binimetinib (2 μM), Palbociclib (2 μM), MK1775 (1 μM), Blebbistatin (10 μM), H1152 (10 μM), PF228 (2 μM), CK666 (100 μM), Nocodazole (1 μM) and DMSO 1 in 1000. Concentrations were calculated including the 100 μl volume of the collagen hydrogel.

Microscopy setup and image acquisition

OPM imaging was performed on a modified version of the OPM system described in [21, 41]. The primary microscope objective was a 60X/1.2NA water immersion objective, the secondary objective was a 50X/0.95NA air objective and the tertiary objective was a 40x/0.6NA air objective. The OPM angle was 35 degrees.

A single sCMOS camera (pco Edge) was used in Global Reset acquisition mode with 1280 $\times$ 1000 pixels. A motorised filter wheel (FW103H/M, Thorlabs) was used to switch between filters for multichannel imaging. An OPM volume was acquired for iRFP (642 nm excitation and 731/137 emission filter (Semrock Brightline)) before the stage returned to the start position prior to the acquisition of the EGFP volume (488 nm excitation and 550/49 emission filter (Semrock Brightline)). Finally, a collagen scattered light volume (488 nm illumination and no emission filter) was acquired from the same start position. The laser illumination and camera exposure time were both 4 ms. The stage velocity was 0.16 $\mu m\ ms^{-1}$ and image acquisition was triggered every 1.4 microns of stage travel. For each field of view, the x-y stage covered 4000 μm and three regions were imaged for each well. Prior to analysis, raw frames were compressed using jetraw compression software (jetraw, Dotphoton). Volumes were then de-skewed into standard xyz coordinates [21] and binned such that the final voxel size was 1x1x1 μm^3. Image reslicing was performed using bi-linear resampling similar to the methods described in [41].

Segmentation

Segmentation of cells and nuclei was by active contour segmentation. Segmentation was performed with a threshold set at the mean volume grey value plus the standard deviation of the volume grey values and finding connected components (minimum connected component sizes were 50 μm^3 and 512 μm^3 for nuclei and cells respectively). These connected components defined sub-regions which were fully segmented.

Outlier removal

The three repeated experiments produced a segmented dataset of more the $7{\times}10^4$ single cells. Similar to [41], we removed cells and nuclei that expressed low CAAX-GFP transgene, as this caused inaccurate segmentation. We automatically removed cells with a maximum intensity less than the mean of all cell maximum intensities minus the standard deviation of all cell maximum intensities. We also removed cells based on their size. That is, if the cell volume $< 512\ \mu m^3$ or the nuclei volume $< 50\ \mu m^3$, the cell was removed from the dataset.

Due to cells being densely packed within each well, there were cases where the segmentation methods grouped two cells as one. To avoid this, we removed cells from the dataset that contained two nuclei within the segmented cell mask.

After the above outlier removal and quality control, there was an additional round of quality control during the data analysis stage. Cells with a mean CAAX-GFP intensity of fewer than 70 units were removed from the study, and cells with a mean nuclear Histone-2B of fewer than 5 units were also removed from the study.

Classical shape measurements

Similar to previous our previous works [41], cell and nuclei classical shape measurements were calculated by the regionprops3 function in MATLAB. Further measurements were derived from these outputs as described in [41].

3D rendering

3D renders of intensity images are generated as a 3D projection with trilinear interpolation by using the volume viewer 2.01 Fiji plugin [64]. 3D renders of cell masks are presented as 3D surface representations using isosurfaces in napari [65].

Dynamic graph convolutional foldingnet

The DFN follows the design of the FoldingNet [31] with a dynamic graph convolutional neural netowork (DGCNN) as the encoder [25]. This encoder takes a 3D point cloud as input, constructs a local neighbourhood (k-nearest neighbour) graph on these points and applies convolution-like operations on the edges of connecting neighbouring points. They call these operations edge convolution (EdgeConv). [25] shows translation-invariant properties of EdgeConv operations. For our experiments, we have chosen $k_g = 20$ for our graph construction. We remove the final linear layer from the original architecture and replace it with one that outputs a feature vector of length 128. The decoder takes the feature vector, $\mathbf{Z}$, as input and concatenates it with points sampled from a plane in 2D space. This is then passed through a series of two folding operations (defined in [31]) to output a reconstructed point cloud. The number of reconstructed points, m, does not need to be the same as the input number of points, n. The Chamfer distance (CD) [66] is commonly used to compare two PCs. We used the extended CD presented in [31] as our reconstruction error between input point cloud S and reconstructed point cloud $\widehat{S}$. The CD is defined in Equation 1.

$$d_{CH}(S, \widehat{S}) = \max \left\{ \frac{1}{|S|} \sum_{\mathbf{x} \in S} \min_{\widehat{\mathbf{x}} \in \widehat{S}} \|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \; \frac{1}{|\widehat{S}|} \sum_{\widehat{\mathbf{x}} \in \widehat{S}} \min_{\mathbf{x} \in S} \|\widehat{\mathbf{x}} - \mathbf{x}\|_2 \right\} \tag{1}$$

Initially, the DFN was trained on the ShapeNet dataset for 250 epochs and then we continued training on our point cloud representations of both cells and nuclei for another 250 epochs using Adam optimiser with e^{-6} weight decay. We used a batch size of 16 with an initial learning rate of $0.001/16$ and an exponential learning rate decay scheduler. The DFN model was set to extract 128 features from each point cloud. All algorithms were implemented in PyTorch.

Deep embedded clustering

Deep embedded clustering (DEC) [57] is a specialised clustering technique that simultaneously learns feature representations and cluster assignments using autoencoders. Following the algorithm described in DEC, we propose learning autoencoder parameters θ which map the 3D shapes into embedded feature space $\mathbf{Z}$ as well as k cluster centres $\{\mu_j \in Z\}_{j=1}^{k}$ of the embedded feature space $\mathbf{Z}$. This is done in two phases:

1. parameter initialisation with an autoencoder (through either 'cellshape-voxel' or cellshape-cloud tools) and
2. parameter optimisation through simultaneous autoencoder reconstruction and minimisation of the Kullback-Leibler (KL) divergence between a target distribution and a distribution of soft-labels.

The second step is done through the addition of a clustering layer (CL) on the features to refine them by learning features that are optimised to represent the 3D shape as best as possible as well as grouping similar and separating dissimilar objects. This part of the model works by initialising cluster centres using the k-means clustering algorithm on the embedded feature space outputted from a pre-trained autoencoder. These cluster centres are kept as trainable parameters. The clustering layer then assigns soft-labels to each input by mapping features to clusters (q) based on the

Student's t-distribution as a kernel that represents the similarity between a feature vector ($\mathbf{Z}$) and a cluster centre (μ):

$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'} \left(1 + \|z_i - \mu_{j'}\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}} \tag{2}$$

This can be interpreted as the probability of assigning input i to cluster j, hence why this is a soft assignment. Soft-assignments with high probabilities are considered trustworthy and thus DEC designs a target distribution which raises this to the second power to place more emphasis on these confident assignments. Following DEC, we define the target distribution as:

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}} \tag{3}$$

We can then define the clustering loss as the Kullback–Leibler divergence between p and q:

$$L = \mathrm{KL}(P \| Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4}$$

The features at cluster centres become representations of 'example' or 'template' shapes of each shape class. [57] proposed training an autoencoder in the first phase (parameter initialisation) and then abandoning the decoder in the second phase to only fine-tune the encoder through the clustering loss alone. Variants of DEC since then have shown that this kind of fine-tuning may distort the embedded space and weaken its representativeness of the input [67, 68]. Thus, we follow the procedure in [49] and have added the decoder back to the second phase of training and optimised both the reconstruction loss and the clustering loss together with a final loss defined as:

$$L = L_r + \gamma L_c \tag{5}$$

where $\gamma \geq 0$ defines the magnitude that the clustering loss adds to the final loss.

Predicting small molecule treatments

Following similar procedures to [31, 69, 70, 71, 55], we used the trained DFN to extract features from each cell. These features were then used to train a linear support vector machine (SVM). For all experiments, we use one-versus-one SVM classifiers with an L2 regularisation parameter (C = 1), balanced class weights, and intercept scaling during training. We tested cytoplasm features and nuclei features alone and then combined them for ablation studies (Figure 4A). We performed a 10-fold cross-validation for each experiment and report the mean accuracy. We used scikit-learn for these methods [72, 73].

Feature importance

Extreme gradient boosting (XGBoost) is a machine learning algorithm which uses gradient boosted decision trees for classification and regression tasks [74]. A benefit of using gradient boosting is that it is relatively simple to explore which features are important for the task at hand. Importance provides a score that indicates how valuable each feature was in the construction of the boosted trees in the model. The more a feature is utilised to make a classification, the higher its importance. This is based on the number of times a feature is used in a tree. We use this for interpreting which features are important for our classification tasks. We trained an XGBoost classification model to classify Blebbistatin and Nocodazole-treated cells based on the principal components of the cell shape features extracted using the DFN without the addition of the clustering layer. For methods involving XGBoost, we used the open-source Python package 'xgboost' [74].

Aligning 3D shapes to a common axis

As a step towards rotational invariance, we propose converting the point cloud representations of 3D cells to their PCA-based canonical poses. PCA calculates 3 orthogonal bases or principal axis of point cloud data. This enables us to align original point clouds to the world Cartesian plane [60]. We briefly discuss how the canonical pose is calculated by following directly from [60]. We perform PCA on a given point cloud, $\mathbf{P} \in \mathbb{R}^{n \times 3}$, by:

$$\frac{\sum \left(\mathbf{P}_i - \overline{\mathbf{P}}\right) \left(\mathbf{P}_i - \overline{\mathbf{P}}\right)^T}{n} = \mathbf{E} \Lambda \mathbf{E}^T, \tag{6}$$

where $\mathbf{P}_i \in \mathbb{R}^3$ is the ith point of $\mathbf{P}, \overline{\mathbf{P}} \in \mathbb{R}^3$ is the mean of $\mathbf{P}, \mathbf{E}$ is the eigenvector matrix composed of eigenvectors $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ (principal axes), and $\Lambda = diag\,(\lambda_1, \lambda_2, \lambda_3)$ are the corresponding eigenvalues. By aligning the principal axes to the three axes of the world coordinate, we obtain the canonical pose as $\mathbf{P}_{can} = \mathbf{PE}$.

[60] goes on to prove the rotational invariant property of $\mathbf{P}_{can}$. We calculate the canonical pose for each point cloud representation of a cell and use this as input to our DFN.

Software availability

The described software is available as a Python package which can be installed through pip and is available at `https://github.com/DeVriesMatt/cellshape`. The main package is made up of four independent sub-packages available at `https://github.com/Sentinal4D/cellshape-cloud`, `https://github.com/Sentinal4D/cellshape-voxel`, `https://github.com/Sentinal4D/cellshape-cluster`, and `https://github.com/Sentinal4D/cellshape-helper`.

Funding

This work was funded by a Cancer Research UK Accelerator Award, a Cancer Research UK Multidisciplinary Project Award (C53737/A24342), and by the Terry Fox Foundation. This work was also supported by a Cancer Research UK and Stand Up to Cancer UK Programme Foundation Award to C.B. (C37275/1A20146). L.R.B is funded by an EPSRC PhD studentship.

Disclosures

C.D has a licensed granted patent on OPM.

References

[1] D. W. Thompson. *On growth and form, 2nd ed.* On growth and form, 2nd ed. Macmillan, Cambridge University Pre, Oxford, England, 1942. Pages: ii, 1116.

[2] Sean Porazinski, Huijia Wang, Yoichi Asaoka, Martin Behrndt, Tatsuo Miyamoto, Hitoshi Morita, Shoji Hata, Takashi Sasaki, S. F. Gabriel Krens, Yumi Osada, Satoshi Asaka, Akihiro Momoi, Sarah Linton, Joel B. Miesfeld, Brian A. Link, Takeshi Senga, Atahualpa Castillo-Morales, Araxi O. Urrutia, Nobuyoshi Shimizu, Hideaki Nagase, Shinya Matsuura, Stefan Bagby, Hisato Kondoh, Hiroshi Nishina, Carl-Philipp Heisenberg, and Makoto Furutani-Seiki. YAP is essential for tissue tension to ensure vertebrate 3D body shape. *Nature*, 521(7551):217–221, May 2015. Number: 7551 Publisher: Nature Publishing Group.

[3] Pei-Hsun Wu, Daniele M. Gilkes, Jude M. Phillip, Akshay Narkar, Thomas Wen-Tao Cheng, Jorge Marchand, Meng-Horng Lee, Rong Li, and Denis Wirtz. Single-cell morphology encodes metastatic potential. *Science Advances*, 6(4):eaaw6938, January 2020. Publisher: American Association for the Advancement of Science.

[4] Samuel H. Church, Seth Donoughe, Bruno A. S. de Medeiros, and Cassandra G. Extavour. Insect egg size and shape evolve with ecology but not developmental rate. *Nature*, 571(7763):58–62, July 2019. Number: 7763 Publisher: Nature Publishing Group.

[5] Veronika Graml, Xenia Studera, Jonathan L.D. Lawson, Anatole Chessel, Marco Geymonat, Miriam Bortfeld-Miller, Thomas Walter, Laura Wagstaff, Eugenia Piddini, and Rafael E. Carazo-Salas. A genomic multiprocess survey of machineries that control and link cell shape, microtubule organization, and cell-cycle progression. *Developmental Cell*, 31(2):227–239, 2014.

[6] Saurav Basu, Soheil Kolouri, and Gustavo K. Rohde. Detecting and visualizing cell phenotype differences from microscopy images using transport-based morphometry. *Proceedings of the National Academy of Sciences*, 111(9):3448–3453, 2014.

[7] Thouis R. Jones, Anne E. Carpenter, Michael R. Lamprecht, Jason Moffat, Serena J. Silver, Jennifer K. Grenier, Adam B. Castoreno, Ulrike S. Eggert, David E. Root, Polina Golland, and David M. Sabatini. Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. *Proceedings of the National Academy of Sciences*, 106(6):1826–1831, 2009.

[8] Chris Bakal, John Aach, George Church, and Norbert Perrimon. Quantitative morphological signatures define local signaling networks regulating cell morphology. *Science (New York, N.Y.)*, 316(5832):1753–1756, June 2007. Number: 5832.

[9] Zheng Yin, Amine Sadok, Heba Sailem, Afshan McCarthy, Xiaofeng Xia, Fuhai Li, Mar Arias Garcia, Louise Evans, Alexis R. Barr, Norbert Perrimon, Christopher J. Marshall, Stephen T. C. Wong, and Chris Bakal. A screen for morphological complexity identifies regulators of switch-like transitions between discrete cell shapes. *Nature Cell Biology*, 15(7):860–871, July 2013.

[10] Pei-Hsun Wu, Jude M. Phillip, Shyam B. Khatau, Wei-Chiang Chen, Jeffrey Stirman, Sophie Rosseel, Katherine Tschudi, Joshua Van Patten, Michael Wong, Sonal Gupta, Alexander S. Baras, Jeffrey T. Leek, Anirban Maitra, and Denis Wirtz. Evolution of cellular morpho-phenotypes in cancer metastasis. *Scientific Reports*, 5(1):18437, 2015.

[11] Jude M. Phillip, Kyu-Sang Han, Wei-Chiang Chen, Denis Wirtz, and Pei-Hsun Wu. A robust unsupervised machine-learning method to quantify the morphological heterogeneity of cells and nuclei. *Nature Protocols*, 16(2):754–774, Feb 2021.

[12] Sam Cooper, Amine Sadok, Vicky Bousgouni, and Chris Bakal. Apolar and polar transitions drive the conversion between amoeboid and mesenchymal shapes in melanoma cells. *Molecular biology of the cell*, 26(22):4163–4170, 11 2015.

[13] Erik Sahai and Christopher J. Marshall. Differing modes of tumour cell invasion have distinct requirements for Rho/ROCK signalling and extracellular proteolysis. *Nature Cell Biology*, 5(8):711–719, August 2003.

[14] Meghan K. Driscoll, Erik S. Welf, Andrew R. Jamieson, Kevin M. Dean, Tadamoto Isogai, Reto Fiolka, and Gaudenz Danuser. Robust and automated detection of subcellular morphological motifs in 3D microscopy images. *Nature Methods*, 16(10):1037–1044, October 2019.

[15] Julia E. Sero, Heba Zuhair Sailem, Rico Chandra Ardy, Hannah Almuttaqi, Tongli Zhang, and Chris Bakal. Cell shape and the microenvironment regulate nuclear translocation of NF-B in breast epithelial and tumor cells. *Molecular Systems Biology*, 11(3):790, March 2015.

[16] Julia E. Sero and Chris Bakal. Multiparametric Analysis of Cell Shape Demonstrates that -PIX Directly Couples YAP Activation to Extracellular Matrix Adhesion. *Cell Systems*, 4(1):84–96.e6, January 2017. Number: 1.

[17] Patricia Pascual-Vargas, Samuel Cooper, Julia Sero, Vicky Bousgouni, Mar Arias-Garcia, and Chris Bakal. RNAi screens for Rho GTPase regulators of cell shape and YAP/TAZ localisation in triple negative breast cancer. *Scientific Data*, 4:170018, March 2017.

[18] Alexandr A. Kalinin, Xinhai Hou, Alex S. Ade, Gordon-Victor Fon, Walter Meixner, Gerald A. Higgins, Jonathan Z. Sexton, Xiang Wan, Ivo D. Dinov, Matthew J. O'Meara, and Brian D. Athey. Valproic acid-induced changes of 4D nuclear morphology in astrocyte cells. *Molecular Biology of the Cell*, 32(18):1624–1633, August 2021. Publisher: American Society for Cell Biology (mboc).

[19] Pedro Gómez-Gálvez, Pablo Vicente-Munuera, Antonio Tagua, Cristina Forja, Ana M. Castro, Marta Letrán, Andrea Valencia-Expósito, Clara Grima, Marina Bermúdez-Gallardo, Óscar Serrano-Pérez-Higueras, Florencia Cavodeassi, Sol Sotillos, María D. Martín-Bermudo, Alberto Márquez, Javier Buceta, and Luis M. Escudero. Scutoids are a geometrical solution to three-dimensional packing of epithelia. *Nature Communications*, 9(1):1–14, July 2018. Number: 1 Publisher: Nature Publishing Group.

[20] C. Dunsby. Optically sectioned imaging by oblique plane microscopy. *Optics Express*, 16(25):20306–20316, December 2008.

[21] Vincent Maioli, George Chennell, Hugh Sparks, Tobia Lana, Sunil Kumar, David Carling, Alessandro Sardini, and Chris Dunsby. Time-lapse 3-d measurements of a glucose biosensor in multicellular spheroids by light sheet fluorescence microscopy in commercial 96-well plates. *Scientific Reports*, 6(1):37777, Nov 2016.

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes, 2014.

[23] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[24] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019.

[25] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019.

[26] Cheng Zhang, Haocheng Wan, Shengqiang Liu, Xinyi Shen, and Zizhao Wu. Pvt: Point-voxel transformer for 3d deep learning. *arXiv preprint arXiv:2108.06076*, 2021.

[27] Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 915–924, October 2021.

[28] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual MLP framework. In *International Conference on Learning Representations*, 2022.

[29] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.

[30] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017.

[31] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation, 2018.

[32] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[33] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[34] Peng-Shuai Wang, Yu-Qi Yang, Qian-Fang Zou, Zhirong Wu, Yang Liu, and Xin Tong. Unsupervised 3D learning for shape analysis via multiresolution instance discrimination. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[35] Yi Shi, Mengchen Xu, Shuaihang Yuan, and Yi Fang. Unsupervised deep shape descriptor with point distribution learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[36] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[37] Haoran Zhou, Yidan Feng, Mingsheng Fang, Mingqiang Wei, Jing Qin, and Tong Lu. Adaptive graph convolution for point cloud analysis, 2021.

[38] Seyed Saber Mohammadi, Yiming Wang, and Alessio Del Bue. Pointview-gcn: 3d shape classification with multi-view point clouds. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3103–3107. IEEE, 2021.

[39] Zhenghua Chen, Min Wu, and Xiaoli Li. Generalization with deep learning, 2021.

[40] Jiahao Pang, Duanshun Li, and Dong Tian. Tearingnet: Point cloud autoencoder to learn topology-friendly representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[41] L. G. Dent, N. Curry, H. Sparks, V. Bousgouni, V. Maioli, S. Kumar, I. Munro, C. Dunsby, and C. Bakal. Environmentally dependent and independent control of cell shape determination by rho gtpase regulators in melanoma. *bioRxiv*, 2021.

[42] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[43] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018.

[44] Ling Zhang and Zhigang Zhu. Unsupervised feature learning for point cloud by contrasting and clustering with graph convolutional neural network, 2019.

[45] Xiang Gao, Wei Hu, and Guo-Jun Qi. GraphTER: Unsupervised learning of graph transformation equivariant representations via auto-encoding node-wise transformations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[46] Juyoung Yang, Pyunghwan Ahn, Doyeon Kim, Haeil Lee, and Junmo Kim. Progressive seed generation auto-encoder for unsupervised point cloud learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6413–6422, October 2021.

[47] Christoph Sommer, Rudolf Hoefler, Matthias Samwer, and Daniel W Gerlich. A deep learning and novelty detection framework for rapid phenotyping in high-content screening. *Molecular biology of the cell*, 28(23):3428–3436, 11 2017.

[48] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels, 2020.

[49] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1753–1759, 2017.

[50] Christopher J. Soelistyo, Giulia Vallardi, Guillaume Charras, and Alan R. Lowe. Learning the rules of cell competition without prior scientific knowledge. *bioRxiv*, 2022.

[51] Uwe Schmidt, Martin Weigert, Coleman Broaddus, and Gene Myers. Cell detection with star-convex polygons. In *Medical Image Computing and Computer Assisted Intervention - MICCAI 2018 - 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II*, pages 265–273, 2018.

[52] Martin Weigert, Uwe Schmidt, Robert Haase, Ko Sugawara, and Gene Myers. Star-convex polyhedra for 3d object detection and segmentation in microscopy. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[53] Carsen Stringer and Marius Pachitariu. Cellpose 2.0: how to train your own model. *bioRxiv*, 2022.

[54] Martin Weigert, Uwe Schmidt, Tobias Boothe, Andreas Müller, Alexandr Dibrov, Akanksha Jain, Benjamin Wilhelm, Deborah Schmidt, Coleman Broaddus, Siân Culley, Mauricio Rocha-Martins, Fabián Segovia-Miranda, Caren Norden, Ricardo Henriques, Marino Zerial, Michele Solimena, Jochen Rink, Pavel Tomancak, Loic Royer, Florian Jug, and Eugene W. Myers. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nature Methods*, 15(12):1090–1097, Dec 2018.

[55] Siming Yan, Zhenpei Yang, Haoxiang Li, Li Guan, Hao Kang, Gang Hua, and Qixing Huang. Implicit autoencoder for point cloud self-supervised representation learning. *arXiv preprint arXiv:2201.00785*, 2022.

[56] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

[57] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 478–487, New York, New York, USA, 20–22 Jun 2016. PMLR.

[58] Matt De Vries and Chris Bakal. What do machines see? Utilizing artificial intelligence to explore cell biology. *The Biochemist*, 43(5):48–52, 09 2021.

[59] Xianzhi Li, Ruihui Li, Guangyong Chen, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. A rotation-invariant framework for deep point cloud analysis. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2021.

[60] Feiran Li, Kent Fujiwara, Fumio Okura, and Yasuyuki Matsushita. A closer look at rotation-invariant deep point cloud analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16218–16227, October 2021.

[61] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[62] Karol Gotkowski, Camila Gonzalez, Andreas Bucher, and Anirban Mukhopadhyay. M3d-cam: A pytorch library to generate 3d data attention maps for medical deep learning, 2020.

[63] Bufan Zhao, Xianghong Hua, Kegen Yu, Wuyong Tao, Xiaoxing He, Shaoquan Feng, and Pengju Tian. Evaluation of convolution operation based on the interpretation of deep learning on 3-d point cloud. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:5088–5101, 2020.

[64] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9(7):676–682, Jul 2012.

[65] Nicholas Sofroniew, Talley Lambert, Kira Evans, Juan Nunez-Iglesias, Grzegorz Bokota, Matthias Bussonnier, Gonzalo Peña-Castellanos, Philip Winston, Kevin Yamauchi, Draga Doncila Pop, Pam, Ziyang Liu, Ahmet Can Solak, alisterburt, Genevieve Buckley, Andy Sweet, Lorenzo Gaifas, Gregory Lee, Jaime Rodríguez-Guerra,

Nathan Clack, Jordão Bragantini, Lukasz Migas, Volker Hilsenstein, Melissa Weber Mendonça, Robert Haase, Hector, Jeremy Freeman, Peter Boone, Alan R Lowe, and Christoph Gohlke. napari/napari: 0.4.13rc0, January 2022.

[66] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2463–2471, 2017.

[67] Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin. Deep clustering with convolutional autoencoders. In Derong Liu, Shengli Xie, Yuanqing Li, Dongbin Zhao, and El-Sayed M. El-Alfy, editors, *Neural Information Processing*, pages 373–382, Cham, 2017. Springer International Publishing.

[68] Xifeng Guo, En Zhu, Xinwang Liu, and Jianping Yin. Deep embedded clustering with data augmentation, 14–16 Nov 2018.

[69] Omid Poursaeed, Tianxing Jiang, Han Qiao, Nayun Xu, and Vladimir G. Kim. Self-supervised learning of point clouds via orientation estimation, 2020.

[70] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matthew J. Kusner. Unsupervised point cloud pre-training via occlusion completion. In *International Conference on Computer Vision, ICCV*, 2021.

[71] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. Spatio-temporal self-supervised representation learning for 3d point clouds, 2021.

[72] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[73] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[74] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.