Deformability Cytometry Clustering with Variational Autoencoders

Daniel D. Seith\textsuperscript{1}, Cody Combs\textsuperscript{2}, and Zuzanna S. Siwy\textsuperscript{1,2,3,*}

\textsuperscript{1}Department of Chemistry, University of California Irvine, Irvine, CA
\textsuperscript{2}Department of Physics and Astronomy, University of California Irvine, Irvine, CA
\textsuperscript{3}Department of Physics and Astronomy, University of California Irvine, Irvine, CA
* zsiwy@uci.edu

October 1, 2022

Abstract

Mechanobiology has shown great success in revealing complex cellular dynamics in various pathologies and physiologies. Most methods for assessing a cell’s mechanical properties, however, generally extract only a few physical constants such as Young’s modulus. This can limit the potential for accurate classification given the wide variety of rheological properties of cells, there are many ways for cells to differ. While it was recently shown that deep learning can classify cells more accurately than traditional approaches, it is not clear how this may be extended to unsupervised classification. In this work, we showcase the potential for a deep learning model to classify cells in an unsupervised fashion using a blend of physical properties. We introduce the combination of a variational autoencoder and a previously described clustering loss for classifying cells in an unsupervised fashion.
Introduction

The ability to characterize a cell’s mechanical properties is emerging as a new method for high throughput characterization. One of the most commonly used mechanotyping method is deformability cytometry and has been shown to characterize COVID-19 pathology, red blood cell disorders, and circulating tumor cells. While there are previous works applying deep learning to deformability cytometry, there are few, if any, instances of the application of unsupervised deep learning to deformability cytometry. Given deep learning has shown to be particularly suited to vision tasks, we wondered if it would also unlock new capabilities for deformability cytometry. In this work, deep learning models for unsupervised deformability cytometry classification are investigated. The dataset for this study was previously described and includes HL60 cells before and after treatment with Cytochalasin D (CytoD). CytoD is known to disrupt actin polarization and leads to an increased deformability of HL60 cells. Unsupervised cell classification could be investigated using two routes (Figure 1). In the first approach, scalar features, such as region deformation, relaxation rate, and peak deformation, can be extracted manually or with a feature extraction algorithm, and subsequently utilized by a clustering algorithm. In a second approach, deep learning methods can be used to better access the high dimensional latent space for clustering.

Unsupervised clustering is first investigated using traditional clustering algorithms: a Gaussian mixture model (GMM) and spectral clustering algorithm, with principle component analysis (PCA) extracted features. However, these scalar features contained a very significant class overlap and the clustering task was too challenging for the clustering algorithms to accurately separate the two populations of cells, HL60 and HL60d. These issues motivated us to look into deep learning models for clustering, namely a variational autoencoder (VAE), which has previously shown success in clustering cell data. Here, a novel unsupervised clustering algorithm using a VAE for deformability cytometry was developed, which is referred to as DeepDeform. A VAE alone was tested first as a benchmark of clustering performance. However, the VAE was limited in its ability to cluster cell populations, since it is only capable of compressing data and is not inherently a classifier. We then assessed the performance of the VAE in tandem with the previously mentioned traditional clustering algorithms, which showed poor clustering performance likely due to improperly balanced clusters. In DeepDeform, a semantic clustering by adopting nearest neighbors (SCAN) loss is integrated to abrogate the deficiencies of traditional VAEs. The implementation of SCAN enabled better access to the high dimensional latent space for clustering cell populations, since it did not require compression of the latent space. Using DeepDeform, the clustering accuracy of the two cell populations rivaled that of several supervised methods. Additionally, the accuracy approached that of supervised models when selecting confident samples.

The manuscript is organized as follows. Section 0.1 provides an overview of the methods, such as the dataset and training procedures. Section 0.2 describes initial efforts to investigate cell populations using conventional clustering and dimensionality reduction techniques, such as principal component analysis (PCA). Section 0.3 presents the development, optimization, and performance of unsupervised clustering using DeepDeform on the HL60 model system.
Figure 1: **Routes to clustering DC data** Two potential routes to clustering deformability cytometry data exist and are shown here. Each provides unique strengths and weaknesses.
0.1 Methods

Channel Design

The dataset for unsupervised clustering was obtained using the same microfluidic channel design as described previously. Briefly, the channel was composed of four zones: two narrow regions each followed by two cavities. Specifically the channel was 150 µm long with equal length sub-regions (50 µm). The narrow regions have widths of 25 µm. The first cavity has a width of 50 µm and the second cavity is the bulk. The height of the channel was 20 µm.

Dataset

The dataset for unsupervised clustering included 2239 human leukemia 60 (HL60) cells. The two classes are comprised of untreated (HL60) and cytochalasin D (CytoD) treated HL60 cells (HL60d). There is a roughly even split (51% - 49%) between the two classes: HL60 and HL60d. CytoD, a fungal toxin, disrupts actin polymerization, resulting in a decrease in cell stiffness, while leaving morphology and radius relatively unaffected. This means that any clustering algorithm needs to rely heavily on mechanical properties to separate the populations rather than more accessible information such as a cell’s morphology or radius. Since the class labels are known, an unsupervised model’s ability to extract these known labels can be evaluated without the model training on them directly. There are 4 different biological replicates of HL60 cells and 4 biological replicates of HL60d cells. Multiple technical replicates were also gathered with the same density of cells being used for each session. The data gathered consist of spatially synchronized 35 frame sequences of 96×96 binary masks of cells passing through the channel. The video data can be represented in fewer dimensions by identifying important features such as peak deformation, change in deformation in the first and second regions, as described previously.

Features

The features for the unsupervised clustering were extracted from the raw binary masks.

VAE Architecture

Provided below is the VAE architecture of DeepDeform. Raw binary mask data were used from the previous work. Adam was used as an optimizer with a learning rate of 1e-3. Gradients were clipped at a norm of 3.0. Experiments were tracked using wandb. The loss function was the sum of reconstruction loss and KL divergence. Both training and validation losses were monitored during training in addition to the average class of the five nearest neighbors (Figure 2). The batch sizes were 32. Data were shuffled each iteration. Unless specified, all dense and convolutional activation functions were leaky rectified linear activation unit.

Encoder Network:

1. Conv2D - Filters: 8 - Kernel: 3×3
2. MaxPooling2D
| Mechanotyping Features Used for Classification | Description |
|-----------------------------------------------|-------------|
| Radius                                        | Radius of cell in cavity where deformation is minimal. Average radius of circle of equivalent perimeter and circle of equivalent area. |
| rD1                                           | Maximum relative aspect ratio of fitted ellipse measured in first narrow constriction. |
| rD2                                           | Maximum aspect ratio of fitted ellipse measured in second narrow constriction. |
| R2 Slope                                      | Slope of linear fit between position of maximum deformation in first narrow constriction and position of least deformation in cavity. Represents cell relaxation. |
| R3 Slope                                      | Slope of linear fit between position of least deformation in cavity and position of maximum deformation in second narrow constriction. Represents cell compression. |
| R1-R3 AR                                      | Difference between AR1 and AR2. |
| R1 Per                                        | Average perimeter of mask in first narrow constriction. |
| R3 Per                                        | Average perimeter of mask in second narrow constriction. |
| R1 Area                                       | Average area of mask in first narrow constriction. |
| R3 Area                                       | Average area of mask in second narrow constriction. |
3. Conv2D - Filters: 16 - Kernel: 3×3
4. MaxPooling2D
5. Conv2D - Filters: 32 - Kernel: 3×3
6. MaxPooling2D
7. Flatten
8. GRU - Dimension 32

$\mu$ Network
1. Dense - Dimension 8

$\sigma$ Network
1. Dense - Dimension 8

Decoder Network:
1. Repeat Vector - Number of repeats: Number of input timesteps
2. GRU - Dimension 32
3. Dense - Dimension 24 × 24 × 64
4. Reshape - 24 × 24 × 64
5. Conv2D Transpose - Filters: 64 - Kernel: 3×3
6. Conv2D Transpose - Filters: 32 - Kernel: 3×3
7. Conv2D Transpose - Filters: 16 - Kernel: 3×3

Semantic Clustering by Adopting Nearest Neighbors (SCAN) Implementation

Once encoder training was completed, the model weights were frozen and a single two dimensional, softmax-activated, dense projection layer was added. The SCAN loss was implemented according to the work of Gansbeke et al. with two separate Adam optimizers for the consistency and entropy components respectively. Both optimizers had a beta$_1$ of 0.9, a beta$_2$ of 0.999 and an epsilon of 1e-7. The learning rate of the consistency optimizer was set to 1e-3 while the entropy optimizer was 5e-3.

$$
\Phi_{\eta}^{c} = \frac{1}{m} \sum_{X \in D} \Phi_{\eta}^{c}(X)
$$
entropy component = \sum_{c \in C} \Phi_{\eta}^{c} \log \Phi_{\eta}^{c}
$$
\text{consistency component} = \frac{1}{m} \sum_{X \in D} \sum_{k \in N_{c}} \log (\Phi_{\eta}(X) \Phi_{\eta}(k))
$$
SCAN loss = \lambda \times \text{entropy component} + \text{consistency component}
Figure 2: Average homogeneity of all points’ nearest 5 neighbors during VAE training.
Unsupervised Clustering Using Manually Extracted Features

In our previous work we observed that for a supervised task, machine learning was required to extract the most useful information out of cell cytometry data. Given that many problems in science are open-ended and do not have known labels, it is necessary to extend our deformability platform to unsupervised problems. It is not, however, clear what the ideal route towards creating a clustering algorithm is (Figure 1). One route, utilizing manually extracted features, could offer a simple and interpretable model for classification. Meanwhile, deep learning utilizes the raw binary masks and could offer improved clustering accuracy.

To investigate the first route, the ten manually extracted features (Table 1) previously found useful for supervised classification were first considered. These data were a good starting point for the analysis since they have a much lower dimensionality than the sequence

Figure 3: First two principal axes from PCA, ground truth labels are shown.
of raw masks (10 vs. 35×96×96). However, we condensed the data further since most clustering algorithms show best performance with two or three dimensions. The dimensionality was further reduced down to two dimensions with PCA, which is a lightweight and accessible technique for dimensionality reduction. With a two-dimensional input space, the practicality of existing clustering algorithms: Gaussian mixture model (GMM) or the spectral clustering model was assessed with the data. Both algorithms are easily accessible and have been previously used for clustering biological data. The desired task was to learn patterns inherent to the data that would allow them to correctly assign class labels (HL60/HL60d) without training with the labels directly. The compressed features are shown in Figure 3. Both GMM and spectral clustering showed poor performance with a clustering accuracy of only 55.4% and 50.0%, respectively. This poor performance was due to the large overlap between the two classes in the lower dimensional space created by PCA. Additionally, more class overlap likely existed in the hand-picked features than in the binary masks.

0.3 Unsupervised Clustering Using Deep Learning

It was hypothesized that an optimized dimensionality reduction technique would compress the input data more effectively than manually extracting features. This would enable the use of richer data, such as the raw binary masks. A variational autoencoder (VAE) was therefore applied as an algorithm to access the rich latent space created by efficient compression of binary masks. VAEs reduces the dimensionality of input data to arbitrary dimensions and has been previously used for clustering cell data.

VAE with Traditional Clustering

To maximize the potential of the model, the raw sequence of binary masks are used as inputs, which contained more data than the manually extracted features. The VAE was created using a convolutional recurrent encoder and a corresponding decoder. The decoder network contained a recurrent layer and used convolutional transpose layers. This allowed us to scale up the decoder input data and create an output size equal to the raw binary masks. The full VAE architecture is provided in Figure 4. The VAE was trained on the sum of the reconstruction loss and the Kullback-Leibler divergence (Equation 1) with an encoder output size of 32 dimensions. This size was chosen through a hyperparameter sweep that optimized the loss of the VAE.

$$\text{VAE loss} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\text{KL}}(q_\phi(z|x)\|p(z))$$ (1)

The GMM and spectral clustering models were implemented in order to utilize this latent space created by the encoder. The results are provided in Figure 5. Since the latent space created by the encoder was still too high-dimensional (32-dimensional) for the clustering algorithms to be used on their own, PCA was subsequently applied to further reduce the dimensionality to two. Both clustering algorithms tested showed an improved mean accuracy but noisy performance (±3.5%). The unstable performance could be caused by an insufficient balance of class assignments.

To seek improvements, the data were carefully studied. Interestingly, we noticed that each point contained a high nearest-neighbor homogeneity, meaning that a point was likely to
be surrounded by other points belonging to the same class. We realized that such structure in the latent space could be harnessed to improve clustering accuracy. Specifically, a technique designed by Gansbeke et al.\textsuperscript{13} was shown to take advantage of such a structure, applied in the context of unsupervised image clustering. Their method, among other features, uses a loss function (Equation 1) that encourages balanced and confident class predictions. We hypothesized that this loss function could also allow the encoder to cluster our data with greater accuracy. The VAE model was then trained and the encoder weights were frozen. A projection layer was then appended to the encoder and the network was trained on the SCAN loss function (Figure 6). The balance between the two components of the SCAN loss (entropy and consistency) was then optimized.

**Initialization Sensitivity Tests**

The next step was to characterize the performance of this trained clustering algorithm. In previous reports, deep unsupervised clustering techniques have been observed to be sensitive to initialization.\textsuperscript{17,18} Therefore, initialization was the first component of the performance to be tested. We tested the model’s sensitivity with five random seeds. The sweep over the five random initializations for the total training, VAE loss and then SCAN loss, resulted in an average accuracy of 61.9±4.2%. As shown in Figure 7, the model was still quite sensitive to initialization.

We have also realized that the poor classification performance could be due to a global structure rather than a local one. Specifically, while the five nearest neighbors generally belonged the same class, the global structure may not be conducive to clustering and is not reflected in the average nearest neighbor homogeneity. We indeed observed that increasing the VAE training time from 30 to 140 epochs was very beneficial. This extended training decreased both the standard deviation in accuracy from 4.2% to 1.3% (Figure 7) and increased the mean accuracy from 61.9% to 67.2%.

To further mitigate noise and improve training stability, we probed a correlation between each of the losses (entropy, consistency loss) and clustering accuracy. If we observed that one of the losses (whose value does not require knowledge of the ground truth label) could be used as a proxy for clustering accuracy, multiple random seeds could be tested facilitating selection of a high-performing model, and an improved overall performance. To investigate a possible correlation, the encoder was trained with five different random seeds (Figure 8). Although no clear correlation between losses and accuracy was observed, it was hypothesized that clustering losses should be compared within a given encoder model, not necessarily between them (i.e. an individual model could be optimized through random seed sweeps for the second step, though not necessarily the first). Next, for each encoder model, three different random seeds were used to train on the SCAN loss. However, none of the loss functions substantially correlated with the accuracy of the clustering, which prevented the selection of a performant model (Figure 9).

**Augmentations and Selecting Confident Samples**

In an effort to improve the model’s clustering accuracy and encourage the encoder to extract semantically relevant features, the strategy of augmenting data was considered.
This strategy seeks to modify the data while leaving the underlying class of the data easily discernible, forcing the model to rely on higher level features. Since the choice of augmentations is generally task-dependent\textsuperscript{19} augmentations were chosen that destroyed low-level information while preserving high-level information, such as the overall order of timesteps. To test the potential of this approach, we used several augmentations during training (Figure 10). The four augmentations provided in Figure 10 include a randomly placed box of zeros of size 30×30 (style 1), complete subtraction of a given image (style 2), zeroing out 33\% of the pixels (style 3), setting 33\% of pixels to one (style 4), and shuffling of a given image with the next two images in the sequence (style 5). Samples were chosen at random to be augmented by any one of the listed styles. If a sample was chosen for augmentation, 10\% of the timesteps were augmented with the randomly chosen augmentation style. It was observed that the augmentations actually decreased the performance significantly (Figure 0.3).

\begin{equation}
\text{confidence} = |\Phi'_{\eta}(X^{(i)}) - 0.5| \tag{2}
\end{equation}

After inspecting model performance more closely, we observed that confidently clustered samples were often accurate (prediction confidence defined in equation 2). Although one generally needs to work with the whole dataset, it can be valuable to select representative examples with a high likelihood of being a true positive. For example, in enriching a cell population for downstream processing, precision is the most important metric. Therefore, we sought to identify examples to represent each cluster by selecting only confident samples (> 0.495). The top predictions made up \sim 10\% of all predictions with an even class distribution (Figure 12). Setting a threshold for classified samples improved the accuracy to \sim 80\% (Figure 13). For comparison, the accuracy of a supervised deep learning model trained on binary masks is 91\% and the accuracy is 74.5\% for a supervised SVM trained on hand-picked features.\textsuperscript{6}

**Conclusions**

This work illustrates that deep learning is required for efficient clustering in deformability cytometry, and provides a new technique for unsupervised clustering. Although simple dimensionality reduction techniques work well in a variety of areas, the work here shows that they are insufficient for the compression of complex data obtained from deformability cytometry. In addition, this work highlights the need for extensive VAE training for the purpose of feature learning. The potential of the SCAN loss in optimizing an encoder to classify unknown populations of cells is also explored using a novel unsupervised algorithm, DeepDeform. These improvements resulted in a model with moderate unsupervised clustering accuracy. Furthermore, when the predictions are thresholded, the accuracy of this model increases to 80\%. A similar model could find applications in selecting rare, unknown groups of cells for further scrutiny, such as single cell RNA sequencing. Future work will entail the improvement of the interpretability of the DeepDeform. The features found most influential to DeepDeform can then be validated with downstream characterization, such as RNA sequencing or microscopy.
Figure 4: Application of VAE architecture for reconstructing the sequence of binary masks. The encoder network is colored orange, and the decoder network is colored blue.
Figure 5: Accuracy of GMM and spectral clustering on trained VAE latent space. Five different random seeds are tested.
Figure 6: The scheme of the training procedure. The encoder model is first trained in conjunction with a decoder using standard VAE losses. The encoder weights are then frozen and a projection layer is appended. This projection layer is then trained using the SCAN loss.
Figure 7: **Longer training improves sensitivity** The accuracy of two training sweeps over 5 random seeds are averaged and shown.
Figure 8: No relationship between SCAN losses and accuracy for a single frozen encoder. A random seed sweep was conducted using five random seeds. Classification accuracy is plotted against entropy and consistency losses.

Figure 9: No relationship between SCAN losses for differently seeded encoders. Each of the encoder models from figure 8 is used to initialize the clustering model. For a single encoder, three different random seeds were used in an effort to find an optimal clustering.
Figure 10: Five different augmentation styles. The first image on the left shows a sample image before augmentation. Style one shows a randomly placed box of zeros of size $30 \times 30$. Style two shows the complete subtraction of a given image. Style three shows 33% of the pixels zeroed out. Style four shows 33% of pixels set to one. Style five shows the shuffling of a given image with the next two images in the sequence.

Figure 11: The fraction of augmented samples versus the classification accuracy. If a given sample is chosen to be augmented, on average 30% of the time steps will be augmented. Each of the five augmentations is selected at random for each timestep.
Figure 12: Averaging class balance over five random seeds. A model output of zero indicates a confident prediction of HL60 while a prediction of one indicates a confident prediction of HL60d. A prediction of 0.5 indicates complete uncertainty, which is illustrated with a dotted black line.
Figure 13: Selection of predictions with a confidence over 0.495 for both the GMM and DeepDeform. The dark green bar and red bars reflect the selection of confident samples for the GMM and DeepDeform, respectively.
Bibliography

(1) Kubánková, M.; Hohberger, B.; Hoffmanns, J.; Fürst, J.; Hermann, M.; Guck, J.; Krater, M. *bioRxiv* 2021.

(2) Caruso, C.; Guruprasad, P.; Mannino, R. G.; Zhang, H.; Josephson, C. D.; Roback, J. D.; Lam, W. A. *Blood* 2018, 132, 1033.

(3) Liu, Z.; Chen, R.; Li, Y.; Liu, J.; Wang, P.; Xia, X.; Qin, L. *Advanced Biosystems* 2018, 2, 1800200.

(4) Nawaz, A. A.; Urbanska, M.; Herbig, M.; Nötzel, M.; Kräter, M.; Rosendahl, P.; Herold, C.; Toepfner, N.; Kubankova, M.; Goswami, R. *and others bioRxiv* 2020, 862227.

(5) Kräter, M.; Abuhattum, S.; Soteriou, D.; Jacobi, A.; Krüger, T.; Guck, J.; Herbig, M. *Adv. Sci.* 2021, 8, 2003743.

(6) Combs, C.; Seith, D. D.; Bovyn, M. J.; Gross, S. P.; Xie, X.; Siwy, Z. S. *Biomicrofluidics* 2022, 16, 014104.

(7) Golfier, S.; Rosendahl, P.; Mietke, A.; Herbig, M.; Guck, J.; Otto, O. *Cytoskeleton* 2017, 74, 283–296.

(8) Kopf, A.; Fortuin, V.; Somnath, V. R.; Claassen, M. *PLoS Comput. Biol.* 2021, 17, e1009086.

(9) Ternes, L.; Dane, M.; Gross, S.; Labrie, M.; Mills, G.; Gray, J.; Heiser, L.; Chang, Y. H. *Commun. Biol.* 2022, 5, 1–10.

(10) Caron, M.; Bojanowski, P.; Joulin, A.; Douze, M. *in ECCV* 2018, pages 132–149.

(11) Liu, Y.; Wang, K.; Sun, X.; Chen, D.; Wang, J.; Chen, J. *Microfluid. Nanofluid.* 2020, 24, 1–11.

(12) Biewald, L. Experiment Tracking with Weights and Biases, Software available from wandb.com, 2020.

(13) Van Gansbeke, W.; Vandenhende, S.; Georgoulis, S.; Proesmans, M.; Van Gool, L. *in ECCV* 2020, pages 268–285.

(14) Liu, Z.; Song, Y.-q.; Xie, C.-h.; Tang, Z. *Signal Image Video Process.* 2016, 10, 359–368.

(15) Zare, H.; Shooshtari, P.; Gupta, A.; Brinkman, R. R. *BMC Bioinform.* 2010, 11, 1–16.

(16) Kingma, D. P.; Welling, M. *arXiv preprint arXiv:1312.6114* 2013.

(17) Chang, J.; Wang, L.; Meng, G.; Xiang, S.; Pan, C. *in ICCV* 2017.

(18) Xie, J.; Girshick, R.; Farhadi, A. *in ICML* 2016, pages 478–487.

(19) Tian, Y.; Sun, C.; Poole, B.; Krishnan, D.; Schmid, C.; Isola, P. *Adv. Neural Inf. Process. Syst.* 2020, 33, 6827–6839.