Received: 19 July 2019 | Revised: 28 September 2019 | Accepted: 16 October 2019

DOI: 10.1002/jbio.201960050

FULL ARTICLE

JOURNAL OF BIOPHOTONICS

Classifying T cell activity in autofluorescence intensity images with convolutional neural networks

Zijie J. Wang[1,2] | **Alex J. Walsh**[2] | **Melissa C. Skala**[2,3] | **Anthony Gitter**[1,2,4*]

[1]Department of Computer Sciences, University of Wisconsin-Madison, Madison, Wisconsin

[2]Morgridge Institute for Research, Madison, Wisconsin

[3]Department of Biomedical Engineering, University of Wisconsin-Madison, Madison, Wisconsin

[4]Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison, Madison, Wisconsin

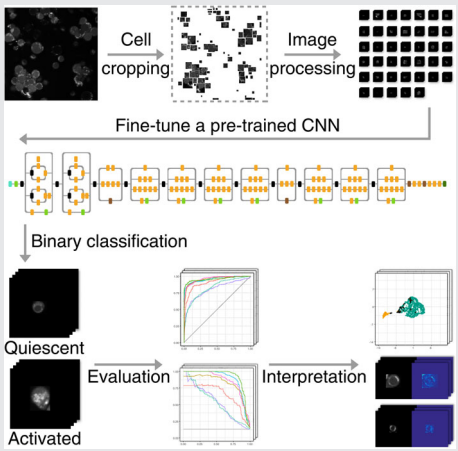
***Correspondence**
Anthony Gitter, Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison, Madison, WI.
Email: gitter@biostat.wisc.edu

Present address
Zijie J. Wang, School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, Georgia.

Alex J. Walsh, Department of Biomedical Engineering, Texas A&M University, College Station, Texas.

Funding information
Morgridge Institute for Research; National Cancer Institute, Grant/Award Numbers: P30 CA014520, R01 CA205101; University of Wisconsin-Madison

Abstract



The importance of T cells in immunotherapy has motivated developing technologies to improve therapeutic efficacy. One objective is assessing antigen-induced T cell activation because only functionally active T cells are capable of killing the desired targets. Autofluorescence imaging can distinguish T cell activity states in a non-destructive manner by detecting endogenous changes in metabolic co-enzymes such as NAD(P)H. However, recognizing robust activity patterns is computationally challenging in the absence of exogenous labels. We demonstrate machine learning methods that can accurately classify T cell activity across human donors from NAD(P)H intensity images. Using 8260 cropped single-cell images from six donors, we evaluate classifiers ranging from traditional models that use previously-extracted image features to convolutional neural networks (CNNs) pre-trained on general non-biological images. Adapting pre-trained CNNs for the T cell activity classification task provides substantially better performance than traditional models or a simple CNN trained with the autofluorescence images alone. Visualizing the images with dimension reduction provides intuition into why the CNNs achieve higher accuracy than other approaches. Our image processing and classifier training software is available at https://github.com/gitter-lab/t-cell-classification.

KEYWORDS
deep learning, label-free, NAD(P)H intensity, transfer learning

1 | INTRODUCTION

Immunotherapy is a type of cancer treatment that uses the body's own immune cells to boost natural defenses against cancer. T cells are a promising target for immunotherapies because of their antigen specificity and diverse cytotoxic and immune-modulating activities. Prior to activation by an antigen, T cells are in a resting or quiescent state. Upon activation, T cells increase in size, proliferate, and produce cytokines [1]. T cells are highly heterogeneous due to various states of activation and production of cytokines with cytotoxic or immune-modulating effects. Immunotherapies that enhance T cell cytotoxicity are currently used in clinical cancer treatments [2]. Other immunotherapies that enhance T cell regulatory activities are in development for diseases including HIV and diabetes [3, 4].

Adoptive cell therapies are the class of immunotherapies in which immune cells from a patient or donor are removed from the body, expanded in vitro, and then injected into the patient. Chimeric antigen receptor (CAR) T cell therapies are adoptive cell therapies in which a patient's T cells are genetically engineered to express a CAR that is specifically targeted to a particular cancer-expressing protein. Several CAR T cell therapies are currently used clinically for cancer treatment. However, extensive functional heterogeneity at the single-cell level has been observed in vivo for CAR T cell immunotherapy for B cell lymphoma in mice, with only approximately 20% of CAR T cell-B cell interactions leading to target killing [5]. The difference in killing efficiency is likely due to heterogeneity in cytotoxic potential among the CAR T cells [5]. Owing to this T cell heterogeneity, T cell activation and function must be assessed in a non-destructive and label-free manner at the single-cell level to allow assessment and purification of the therapeutic cells subsequently injected into the patient. However, most current T cell profiling methods, such as immunofluorescence of surface protein expression and cytokine production, rely on exogenous contrast agents. Labelling intracellular cytokine production requires cell fixation, limiting application of these methods to in vitro assessment of subsets of T cells. A label-free and non-destructive pipeline to determine T cell activation is necessary for in vitro characterization and sorting of expanded T cells to ensure optimally functional T cells are used in cellular immunotherapies and for in vivo preclinical assessment of immunotherapies [6].

Autofluorescence imaging is appealing because it only relies on endogenous contrast and is non-destructive. Endogenous fluorophores include NAD(P)H, FAD and collagen. Reduced nicotinamide adenine dinucleotide (NADH) and flavin adenine dinucleotide (FAD) are coenzymes of metabolism. NAD(P)H is an electron donor and is produced in glycolysis and consumed in oxidative phosphorylation. FAD is an electron acceptor and is produced in oxidative phosphorylation. The fluorescence lifetime is the time that the fluorophore is in the excited state, typically picoseconds to nanoseconds in duration, and is sensitive to the microenvironment of the fluorophore. Activated, functional immune cells require specific metabolic programs to support high levels of proliferation and cytokine production. Therefore, autofluorescence imaging of NAD(P)H and FAD provides endogenous endpoints of cellular metabolism reflective of immune cell function. Previous studies have used autofluorescence lifetime imaging to identify macrophages within the tumor microenvironment in vivo [7] and classified the activation state of T cells in vitro [6]. The fluorescence lifetime of NAD(P)H and FAD is highly sensitive to the microenvironment and binding of NAD(P)H and FAD. Thus, this fluorescence lifetime can be used to resolve metabolic differences between functional states of immune cells. However, fluorescence lifetime imaging requires specialized and expensive microscope components, limiting its use. Although autofluorescence intensity images lack the depth of information provided by the fluorescence lifetime, intensity images can easily and quickly be acquired on almost any commercial fluorescence microscope, allowing widespread adoption and seamless integration of a new technique into existing protocols of live cell assessment. Here, we develop a computational framework that uses autofluorescence intensity images to assess T cell activation state at the single-cell level.

Machine learning is promising for classifying cell subtypes from label-free images. For example, Pavillon et al. [8] used regularized logistic regression to predict macrophage activation state, and Yoon et al. [9] identified lymphocyte cell types with *k*-nearest neighbors. Advanced machine learning models, in particular convolutional neural networks (CNNs), are now the prevailing approach for a variety of cellular image analyses [10–12]. CNNs can classify cell phenotypes [13, 14], segment cells [15, 16], restore images [17], and predict protein localization [18, 19], cell lineage choice [20], the biological activity of small molecules [21] or ratios of activated T cells in a population [22]. They are also effective at cell type classification tasks such as predicting cell cycle state [23] and cell sorting [24].

In this study, we use transfer learning with a pretrained CNN to classify T cell activation state at the single-cell level. Transfer learning re-uses a model for one task to improve performance on another task. Instead of extracting a small set of features from images before training a cell type classifier [25], we treat the autofluorescence intensity images as the input and take

advantage of an existing CNN that has been trained on generic images. The pre-trained CNN extracts high dimensional image features. We train a simple classifier on these features or fine-tune partial layers to adapt the CNN for T cell activity classification. Repurposing a CNN pre-trained on generic images has been successful in medical imaging applications [26–28] and cellular image analyses [29] such as classifying white blood cell types [30], recognizing cell staining patterns [31], and predicting mechanism of action in compound treatments [32–34]. Compared to end-to-end CNN training, the transfer learning approach is more computationally efficient and requires fewer training samples. Because T cells differ from donor to donor in real immunotherapy applications, we use a rigorous donor-specific cross-validation scheme to train and evaluate our models. For the same reason, we hold out all images from one donor and only use them to assess the final performance of our best model.

The pre-trained CNN can accurately classify T cell activity across donors with autofluorescence intensity images as the only input. We compare the pre-trained CNN to a spectrum of simpler models to better understand when and why deep learning is needed. Adapting pre-trained CNNs is an important strategy in this domain and the most accurate approach overall, improving upon classifiers that operate on previously-extracted cell image features. In particular, fine-tuning some higher-level layers outperforms directly using pre-trained CNN-extracted features. However, it is generally not worth the additional computational expense to fine-tune all layers of the CNN. Interpretation techniques demonstrate that the pre-trained CNN learns better representations for the two types of T cell images than other featurizations. Our success in classifying T cell activity without exogenous contrast agents or fluorescence lifetime suggests that modern machine learning approaches may help compensate for imaging data with less molecular specificity.

2 | RESULTS

2.1 | Overview

Our goal is to classify individual T cells as activated (positive instances) or quiescent (negative instances) using only cropped autofluorescence intensity cell images. We explore multiple classification approaches of increasing complexity. A frequency classifier uses the frequency of positive samples in the training set as the probability of the activated label. This naive baseline model assesses how well the class skew in the training images can predict the label of new images. In addition, we test three Lasso logistic regression approaches on different featurizations of the cropped T cell images. The first uses the image pixel intensities directly as features. The second uses only two image summaries as features, the cell size and total intensity. The third uses attributes calculated with CellProfiler [35], such as the mean intensity value and cell perimeter.

We also assess multiple types of neural networks. A fully connected neural network (multilayer perceptron) generalizes the logistic regression model with pixel intensities by adding a single hidden layer. The LeNet CNN architecture [36] learns convolutional filters that take advantage of the image structure of the input data. This CNN is simple enough to train from random initialization with a limited number of images. Finally, we consider two deeper and more complex CNNs. Both use transfer learning to initialize the Inception v3 CNN architecture with a model that has been pre-trained on generic (non-biological) images. One version trains a new fully connected layer from scratch using off-the-shelf features extracted from cell images with the pre-trained CNN. An alternative fine-tunes multiple layers of the pre-trained CNN.

We select these classifiers from the same broad category. Except for the trivial frequency classifier, all models can be represented as a form of neural network with different input features and architectures. Also, we select features that are easy to extract from cell images, such as the raw pixel matrix and total intensity, as well as CellProfiler attributes, which are commonly used in cellular image classification studies [8, 37].

The overall workflow for our pre-trained CNN with fine-tuning is described in Figure 1. The original microscopy images are segmented, cropped and padded. We filter images that do not contain a T cell and other artifacts, leaving the final image counts for each of the six donors shown in Table 1. Then we train, evaluate and interpret the machine learning models. Figure 1 shows the training procedure for the pre-trained CNN with fine-tuning as an example.

The T cell microscopy images may vary from donor to donor. A trained model must be able to generalize to new donors in order to be useful in a practical pre-clinical or clinical setting. Therefore, all of our evaluation strategies train on images from some donors and evaluate the trained models on separate images from a different donor, which is referred to as subject-wise cross-validation [38] or a leave-one-patient-out scheme [31]. We initially assess the classifiers with cross-validation across donors. In addition, we hold out all images from a randomly selected donor, donor 4, and only use them after completing the rest of our study to confirm that our model selection and hyper-parameter tuning strategies generalize to a new donor.

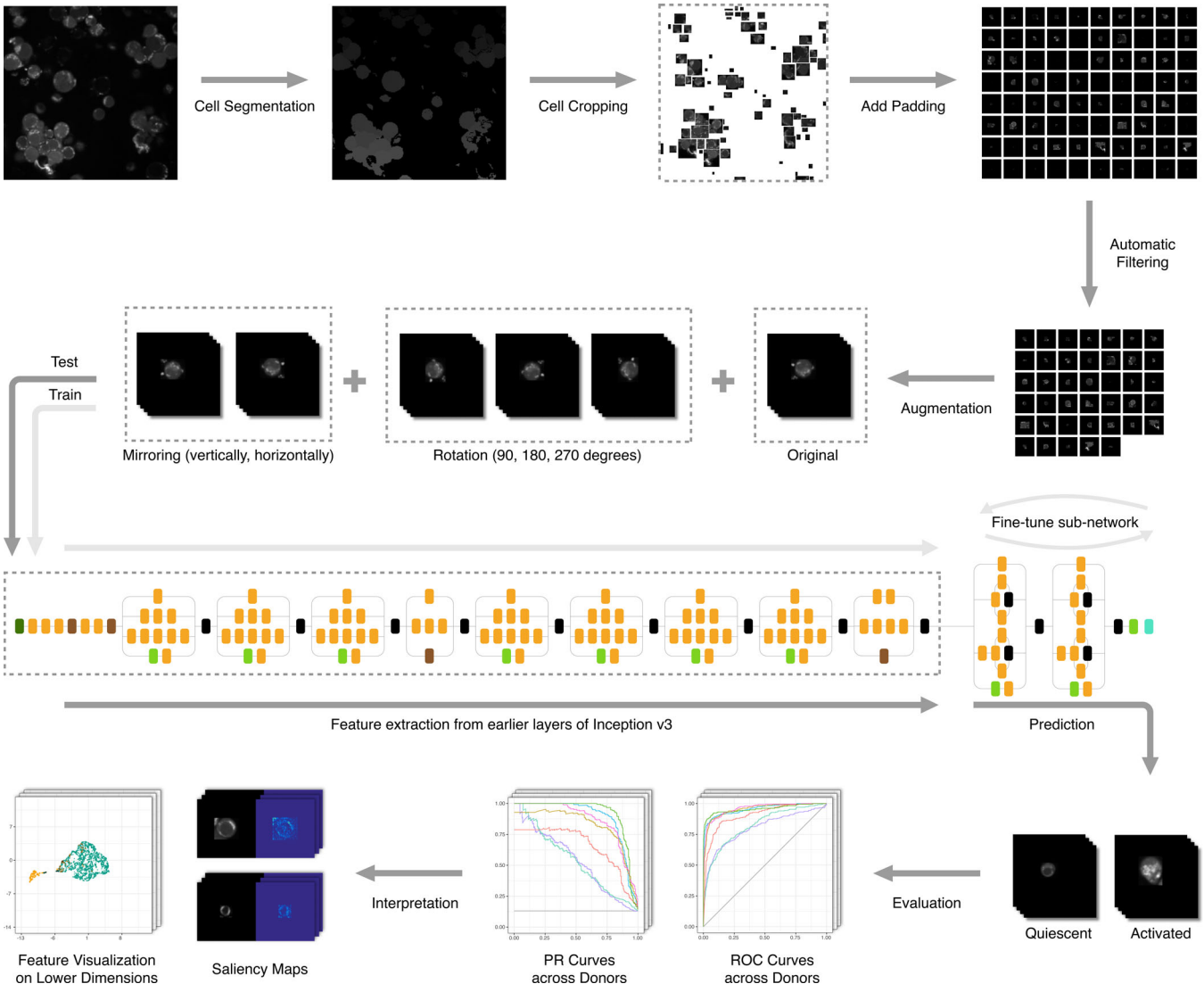


FIGURE 1 Our T cell image data processing workflow

TABLE 1 Image count per donor in different filtering stages

	Original			After lifetime filtering			After entropy and intensity filtering		
	Cell count			Cell count			Cell count		
	Activated	Quiescent	Class skew	Activated	Quiescent	Class skew	Activated	Quiescent	Class skew
Donor 1	609	2184	0.22	271	1631	0.14	235	1551	0.13
Donor 2	1139	399	0.74	656	152	0.81	647	141	0.82
Donor 3	604	2351	0.20	487	1276	0.28	446	1238	0.26
Donor 4	789	2110	0.27	494	1629	0.23	482	1569	0.24
Donor 5	791	528	0.60	694	252	0.73	683	246	0.74
Donor 6	531	1007	0.35	446	589	0.43	442	580	0.43

Note: The mean fluorescence lifetime filter is used to discard cells that are visually indistinguishable from T cells, such as red blood cells. Then, the entropy and total intensity thresholds are used to remove dim images or images containing no cells. The donor class skew is measured as the percentage of activated cells.

2.2 | Cross-validation across donors

In order to assess our classifiers' performance on cell images from new donors, we design a nested cross-validation scheme to train, tune and test all models. Due to this cross-validation design, the same model could have different optimal hyper-parameters for different test donors. Therefore, we group the final model performance by test donors (Figure 2). We plot multiple evaluation metrics because each metric rewards different behaviors [39]. The area under the curve (AUC) and average precision are summary statistics of the receiver operating characteristic (ROC) curve (Figure 3) and precision recall (PR) curve (Figure 4), respectively. The test donors have different training sets and class skews (Table 1), and some classifiers are more robust to imbalanced data than others. Therefore, each donor has a specific pattern in the two curves, especially in the PR curves (Figure 4). However, for each curve, the relative ordering of the classifiers is generally consistent across donors. For all three evaluation metrics, the two pre-trained CNN models outperform other classifiers.

The frequency classifier's average accuracy for all test donors is 37.56% (Figure 2 and Table S1). The low accuracy of this simple method implies that the majority class in the training and test sets is likely to be different. For example, there are more activated cells from donor 2 while there are more quiescent cells from the combination of donors 1, 3, 5 and 6. This baseline establishes that classifiers that fail to use features other than the label count will perform poorly.

Three logistic regression models using different features all give better classifications than the baseline model. Logistic regression with the image pixel matrix leads to an average accuracy of 78.74% (Figure 2 and Table S2). Among those 6724 pixel features, 5822 features on average are removed by the Lasso regularization. To interpret this model, we plot the exponential of each pixel's coefficient to visualize the odds ratios. As shown in Figure S1, this model learns the shape of cells. Larger cells are more likely to be classified as activated. Logistic regression using only mask size and total intensity as features gives slightly better performance with an average accuracy of 79.93% (Figure 2 and Table S3). For all test donors, the optimal coefficient of cell mask size is negative, whereas the coefficient of total intensity is positive. In practice, we expect larger cells to be activated, but the negative coefficient indicates the model learns the wrong relationship between cell size and activity state. This can be explained by the inconsistent cell size distribution across donors (Figure S2) and the correlation of cell size and total intensity (multicollinearity). Comparing the odds ratio of one standard deviation (SD) increment of each feature, however, shows this logistic regression model is much more sensitive to total intensity than cell size. Finally, the logistic regression model with CellProfiler attributes yields 87.14% average accuracy (Figure 2 and Table S4). After computing the odds ratio adjusted to the SD of each feature, attributes that are related to image intensity and cell area have the strongest impact on the predictions.

Non-linear models with image pixels as input have accuracies comparable to the logistic regression model with CellProfiler features. We tune the learning rate, batch size and the number of hidden layer neurons of the simple neural network with one hidden layer. Even though its average accuracy of 86.48% (Figure 2 and Table S5) is slightly lower than logistic regression with CellProfiler features, it has more stable performance across the test donors. In comparison, the LeNet CNN has a more complex architecture and takes advantage of the image structure of the input data. After selecting the best learning rate and batch size, LeNet reaches an average accuracy of 89.51% (Figure 2 and Table S6).

Our most advanced models using the pre-trained CNN outperform all other methods. Both versions of the pre-trained CNN use cell images as input and require a previously trained CNN. For one version, we use the pre-trained CNN as a feature extractor and then train a new hidden layer with off-the-shelf features. Alternatively, we fine-tune multiple higher-level layers of the CNN with T cell images. We include the fine-tuned layers as a hyper-parameter. Specifically, we define n, ranging from 1 to 11, as the number of last Inception modules in the pre-trained Inception v3 CNN to fine-tune. For example, if $n = 1$, we only fine-tune the last Inception module, whereas we fine-tune all the layers of the Inception v3 CNN when $n = 11$. After tuning n along with the other hyper-parameters, we compare the CNN with fine-tuning to the CNN off-the-shelf model in order to study the effect of fine-tuning on classifier performance. In addition, we compare the test results of different n to analyze how the number of fine-tuned layers affects classification.

The average accuracy for the pre-trained CNN off-the-shelf model is 90.36% (Figure 2 and Table S7) and 93.56% for the pre-trained CNN with fine-tuning (Figure 2 and Table S8). The fine-tuning model uses 11, 10, 7, 11, and 8 layers as the optimal n for the five test donors. However, depending on the test donor and the evaluation metric, the number of fine-tuned layers does not necessarily have a strong effect on the predictive performance (Figure 5). Different n values yield similar evaluation metrics. Fine-tuning all 11 layers also greatly increases the CNN training time (Figures S3 and S4).

2.3 | Confirming generalization with a new donor

In order to evaluate our ability to generalize to T cell images from a new individual, we completely hold out

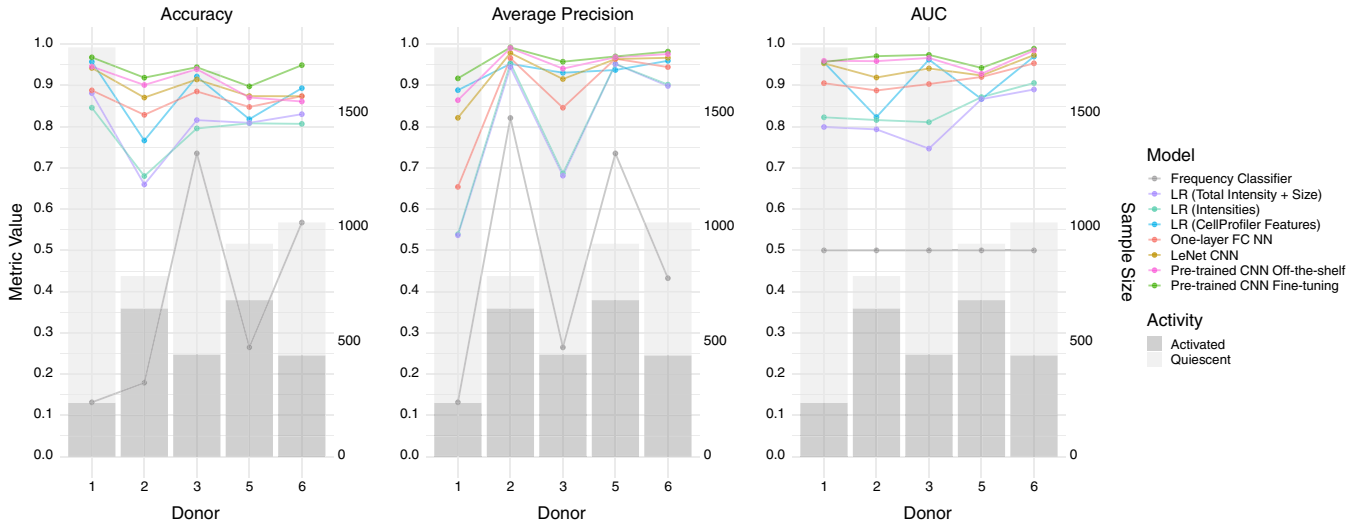


FIGURE 2 Model summary per donor for three different evaluation metrics. The line graphs display the classifiers' performance across donors. The bar graphs display the number of activated and quiescent images for each donor, which affects the baseline accuracy and average precision of a random classifier

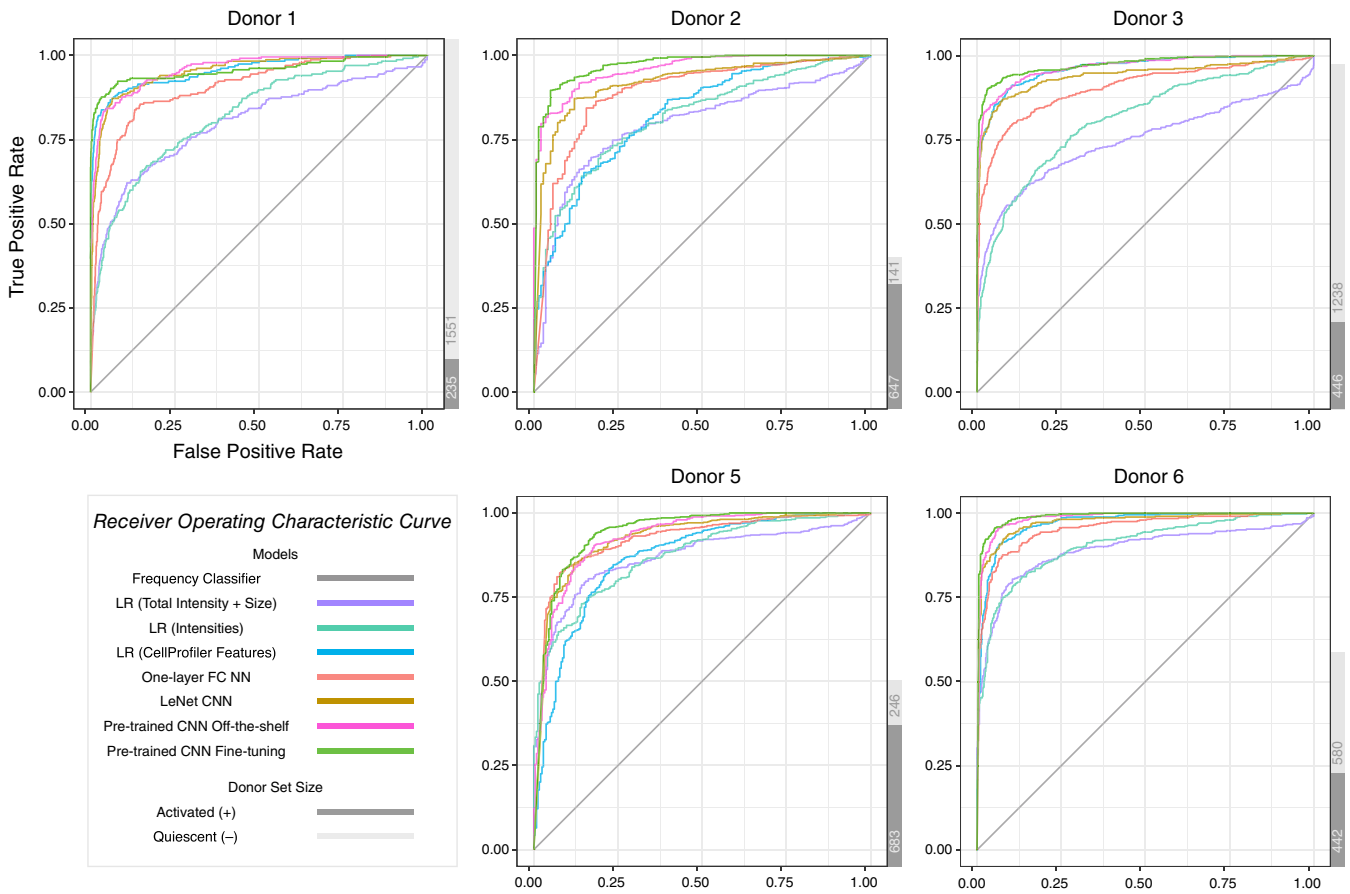


FIGURE 3 ROC curves for each type of classifier and donor. The gray bars to the right display the number of activated and quiescent images for each donor [Correction added on 17 February 2020, after first online publication: This figure has been corrected and replaced]

images from donor 4 during the study design, model implementation and cross-validation above. We apply the same nested cross-validation scheme to train, tune and test the pre-trained CNN with fine-tuning, the most accurate model in the previous cross-validation, on images from donor 4. It gives an accuracy of 98.83% (Table 2).

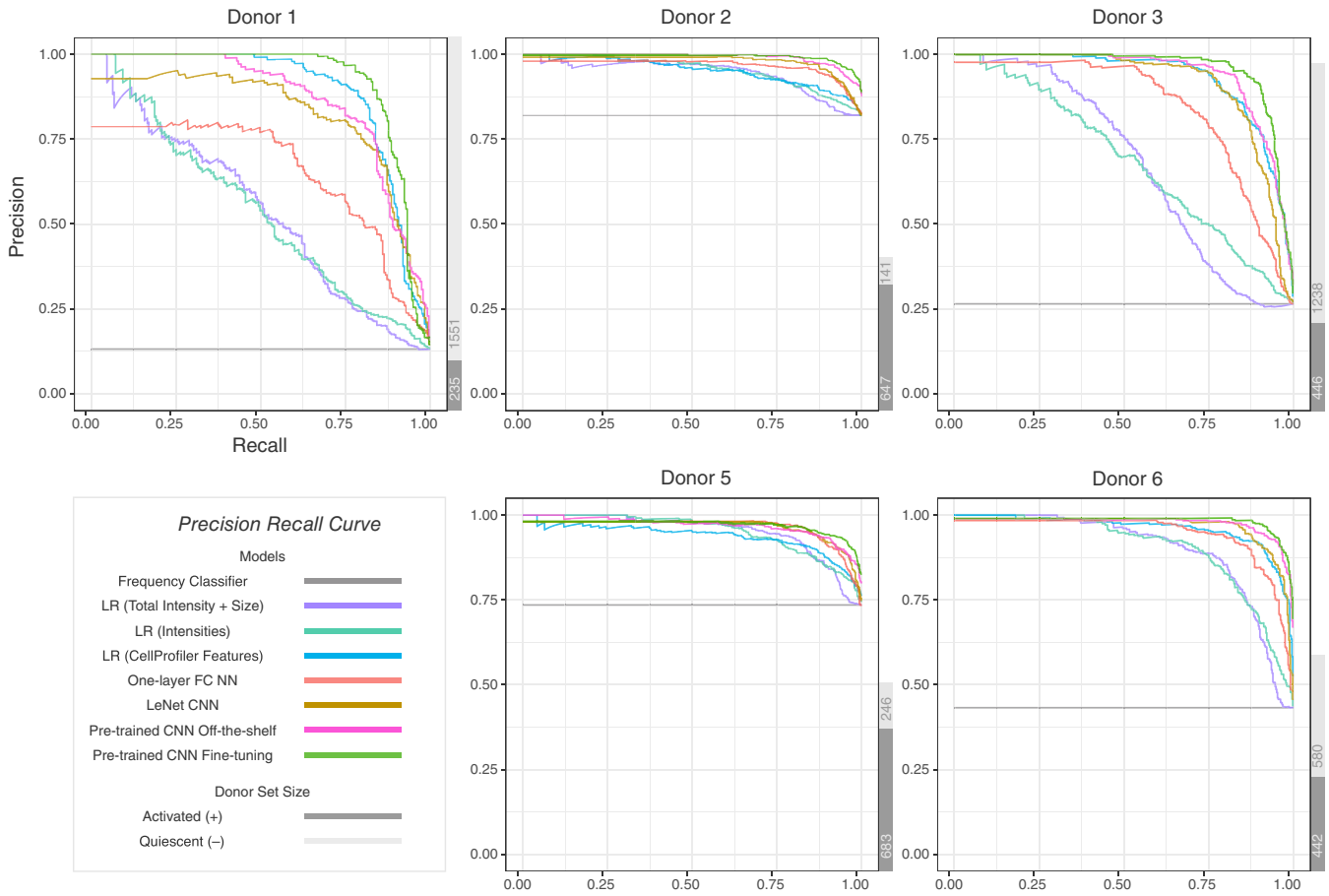


FIGURE 4 PR curves for each type of classifier and donor. The gray bars to the right display the number of activated and quiescent images for each donor

Out of 2051 predictions, there are only four false positives and 20 false negatives. The performance metrics in Table 2 are substantially higher than their counterparts in Table S8. Having training data from five donors instead of four likely contributes to the improved performance.

2.4 | Pre-trained CNN with fine-tuning errors

We inspect the T cell images that the pre-trained CNN with fine-tuning classifies incorrectly in order to better understand its failures and accuracy. We visualize the misclassified images for all test donors in Figures S5-S10 along with the predicted label, the Softmax score of the network output layer and the temperature-scaled confidence calibration [40]. The majority of misclassified cell images are badly cropped, with no cells or multiple cells included in the frame. Therefore, using a more progressive dim image filter or adding a multiple-cell detector in the image processing pipeline could further improve the model performance. However, for other images with a clear single cell in the frame, the pre-trained CNN tends to give high confidence in its misclassification. These scores suggest that these errors cannot be easily fixed without a more powerful classifier or more diverse training dataset. Temperature scaling could either soften the original Softmax score toward 50% or increase the confidence toward 100%. For the misclassified images in our study, temperature scaling always drops the Softmax probability. This observation matches Guo et al.'s finding that neural networks with higher model capacity are more likely to be overconfident in their predictions [40].

2.5 | Pre-trained CNN with fine-tuning interpretation

Visualizing the T cell dataset in two dimensions (2D) helps us understand why some classifiers perform better than others. We use Uniform Manifold Approximation and Projection (UMAP) [41] to project the images into 2D such that similar images in the original feature space are nearby in the 2D space. Coloring the images with their activity labels shows how different input

representations or learned representations separate the activated and quiescent cells. For example, in Figure 6, each dot corresponds to one image based on its representation in the last layer of the pre-trained CNNs with fine-tuning. UMAP projects the 2048 learned features in the last layer of the CNN into 2D. In general, the activated and quiescent cells are well-separated in the 2D space, suggesting that the CNN has successfully learned distinct representations for the two types of T cells. Using t-Distributed Stochastic Neighbor Embedding (t-SNE) [42] instead of UMAP for dimension reduction provides qualitatively similar results (Figure S11).

Generating similar UMAP plots for three alternative image representations shows that the two image classes are not as well separated (Figures S12-S14). When using the raw pixel features (Figure S12), the two types of T cells are spread throughout the 2D space. This contributes to the lower performance of the logistic regression and fully connected neural network models that operate directly on pixel intensity. Similarly, there is only moderate spatial separation when using the CellProfiler features (Figure S13) or the last layer of the CNN before fine-tuning it to predict T cell activity (Figure S14). These comparisons demonstrate the strong effect of fine-tuning the pre-trained CNN and also help explain the superior performance of pre-trained CNNs over the logistic regression model with CellProfiler features. In addition, by annotating the images that are misclassified by the pre-trained CNN with fine-tuning as outlined dots in each of the 2D representations, we see where this classifier makes errors. In Figure 6, the incorrect predictions are predominantly distributed in the boundary between the two clusters.

In addition to visualizing the feature representation in the pre-trained CNNs with fine-tuning, we use saliency maps [43] to interpret how these models make decisions. We generate saliency maps by computing the gradient of the CNN class score with respect to a few randomly chosen donor 1 images from both the activated and quiescent classes (Figure 7). We use two methods to calculate gradients: standard backpropagation and guided backpropagation [44]. In these heat maps, larger values (green or yellow) highlight the image regions that cause the most change in the T cell activity prediction. Smaller values (dark blue or purple) indicate pixels that have less influence. The uniformly dark blue background in both types of saliency maps indicates that the pre-trained CNNs with fine-tuning have learned to focus on the original cell image instead of the black padding. The larger values in the saliency maps with guided backpropagation often align with the high-intensity regions of the cell images, which correspond to mitochondria and depict metabolic activity [45]. Although the influential regions of the guided backpropagation-based saliency maps are biologically plausible, this type of saliency map is insensitive to random changes of either the input data or model parameters [46]. The saliency maps generated with standard backpropagation are properly affected by these randomized controls but do not concentrate on the high-intensity regions of the input images.

2.6 | Running the analysis pipeline

Our GitHub repository https://github.com/gitter-lab/t-cell-classification demonstrates and documents all steps of our analysis pipeline, from pre-processing the cropped images to fine-tuning and interpreting the Inception v3 CNN. Our Python code is presented in Jupyter notebooks

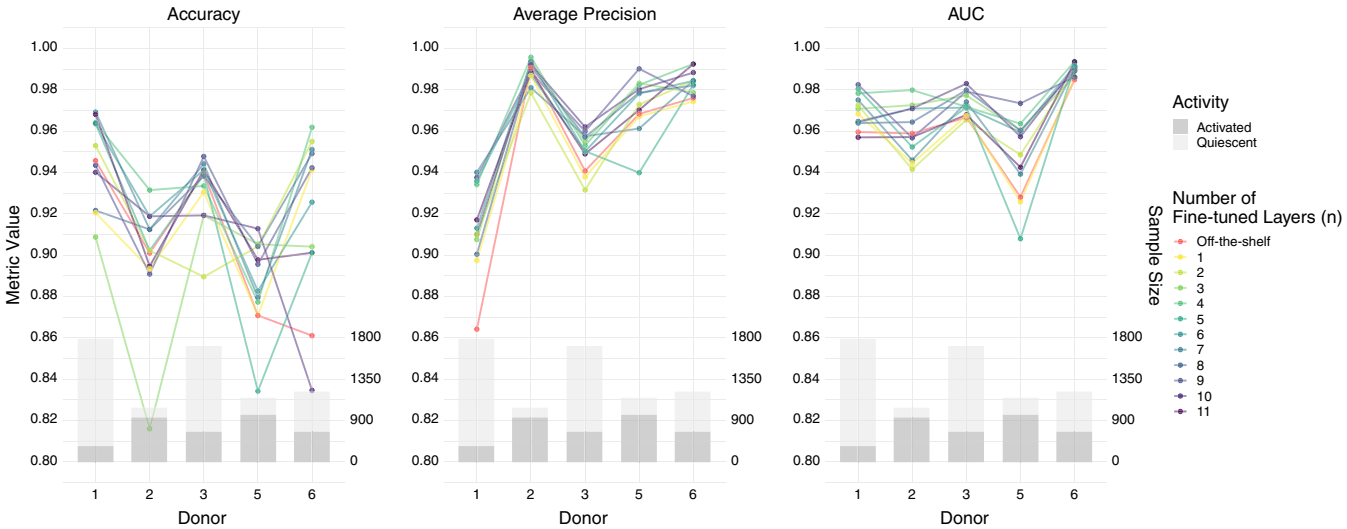


FIGURE 5 Performance comparison of fine-tuning a different number of layers and the pre-trained CNN off-the-shelf model

TABLE 2 Performance of the pre-trained CNN with fine-tuning on held out donor 4

Donor	Accuracy	Precision	Recall	Average precision	AUC	Activated count	Quiescent count
4	98.83%	99.14%	95.85%	99.79%	99.93%	482	1569

Abbreviation: CNN, convolutional neural network.

[47], which integrate our code with explanations of its functionality and visualizations of its outputs. These notebooks can serve as a tutorial of best practices in machine learning on autofluorescence microscopy images. We use Binder [48] to enable readers to execute our Jupyter notebooks in a web browser without having to download any software locally or configure a Python environment. We ensure the Jupyter notebooks continue to run as expected by automatically testing them in Linux, macOS and Windows with the Travis CI and AppVeyor continuous integration services. Our software re-runs our analyses with a randomly selected 10% of the images. This allows users to quickly train the machine learning models and understand our pipeline.

3 | DISCUSSION

Our study demonstrates that machine learning models trained on autofluorescence intensity images can accurately classify activated and quiescent T cells across donors. Because autofluorescence images are easier to acquire with standard commercial microscopes compared to fluorescence lifetime images, this workflow has the potential to become a widely applicable approach for live T cell profiling. Fine-tuning a pre-trained CNN is the most powerful classification approach, outperforming alternative machine learning models that are commonly used for microscopy image classification over multiple evaluation metrics. In particular, this CNN applied directly to cropped images has better performance than logistic regression with domain-relevant features extracted by CellProfiler.

We thoroughly explored the effect of fine-tuning more layers of the pre-trained CNN and compared it with the off-the-shelf CNN model. The common transfer learning approach fixes the CNN parameters of the initial network layers, which extract learned features from the images, and trains a simple classifier from scratch that predicts the domain-specific image labels. Our results indicate that fine-tuning pre-trained CNN layers yields better performance than directly using off-the-shelf features. In addition, although fine-tuning more layers tends to give better predictive performance (Figure 5), it is generally not worth the additional computational time and expense to fine-tune all 11 layers (Figures S3 and S4). Possible reasons include the limited sample size and relatively homogeneous cell image representations. Given the extra computational costs and implementation challenges, we recommend fine-tuning only the last few layers of a pre-trained CNN for similar autofluorescence microscopy applications. In settings that do require fine-tuning additional layers because the images are more heterogeneous, we suggest taking a larger step size in the layer number hyper-parameter optimization.

The machine learning models recognize image attributes that recapitulate biological domain knowledge. Activated T cells are larger in size [6, 49]. In addition, there are metabolic differences between quiescent and activated T cells [1], which are evident in the NAD(P)H images. The high intensity regions in the images likely correspond to mitochondria, where the majority of metabolism occurs. It is straightforward to inspect the trained logistic regression model that takes total image intensity and mask size as inputs and observe that it correctly recognizes the relationship between NAD(P)H intensity and activation state.

The parameters of the pre-trained CNN with fine-tuning are not as directly interpretable as the logistic regression model. An additional challenge is that different interpretation techniques provide distinct views of the fine-tuned CNN. Nevertheless, there are some indications in the saliency maps that this CNN also reflects T cell biology. Saliency maps help locate which regions of the input image influence the classification the most. With guided backpropagation, the high-intensity regions of the T cell images tend to be the focal points in the saliency maps. This suggests that the CNN may be sensitive to metabolic differences between quiescent and activated cells and not only changes in cell size. However, guided backpropagation and other more advanced saliency maps were found to be independent of the data, model and model parameters [46]. The standard backpropagation gradient map is sensitive to these controls, but it focuses more on general cell morphology than the metabolic activity within cells.

Each model in our study is only tuned and evaluated once, which limits our ability to assess the statistical significance of the performance differences across models. Substantial computing time and costs are required for nested cross-validation, especially when fine-tuning

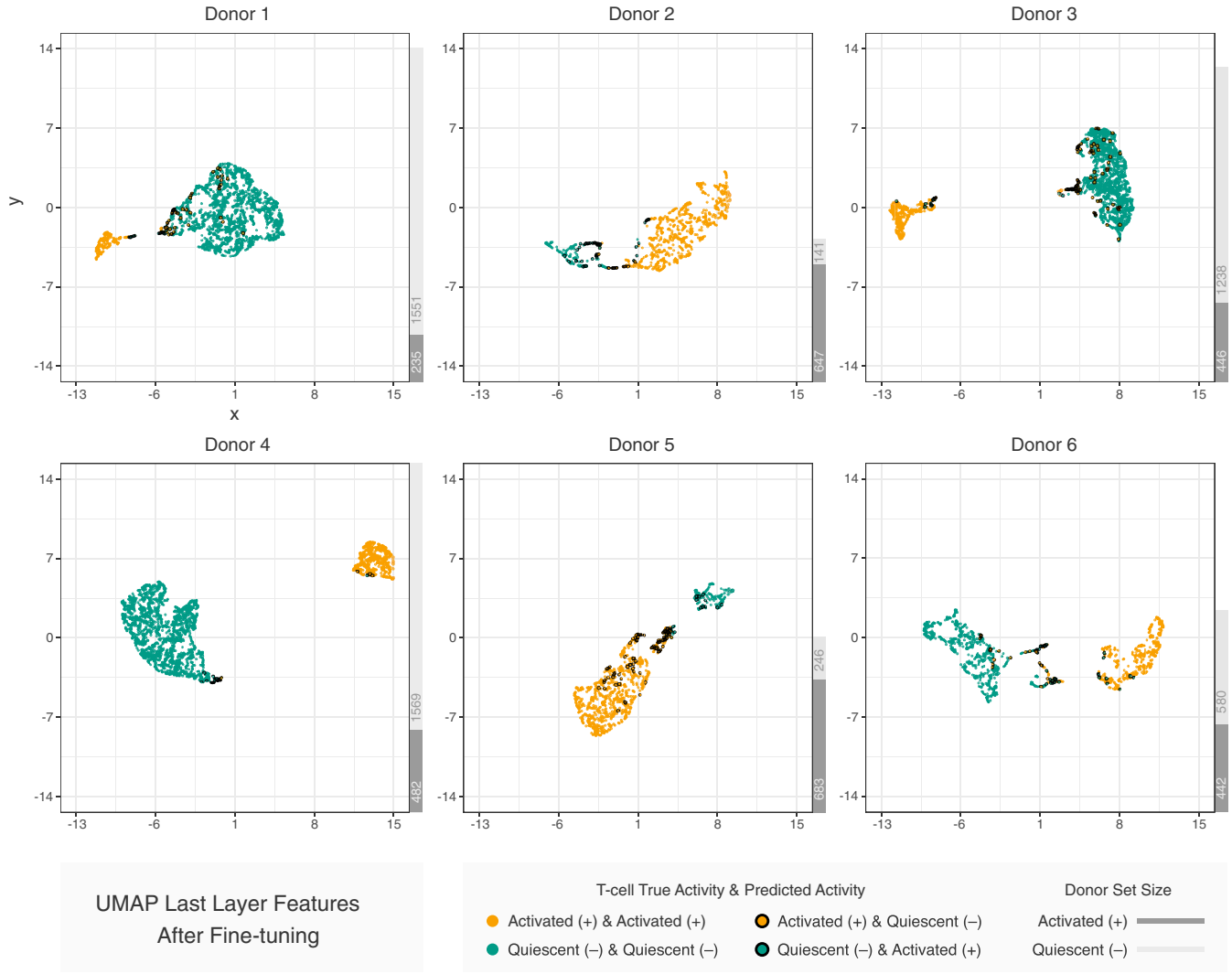


FIGURE 6 2D representations of T cell features extracted from the pre-trained CNN with fine-tuning. Dimensions are reduced from 2048 using UMAP. The thick outlines indicate incorrect cell activity state predictions

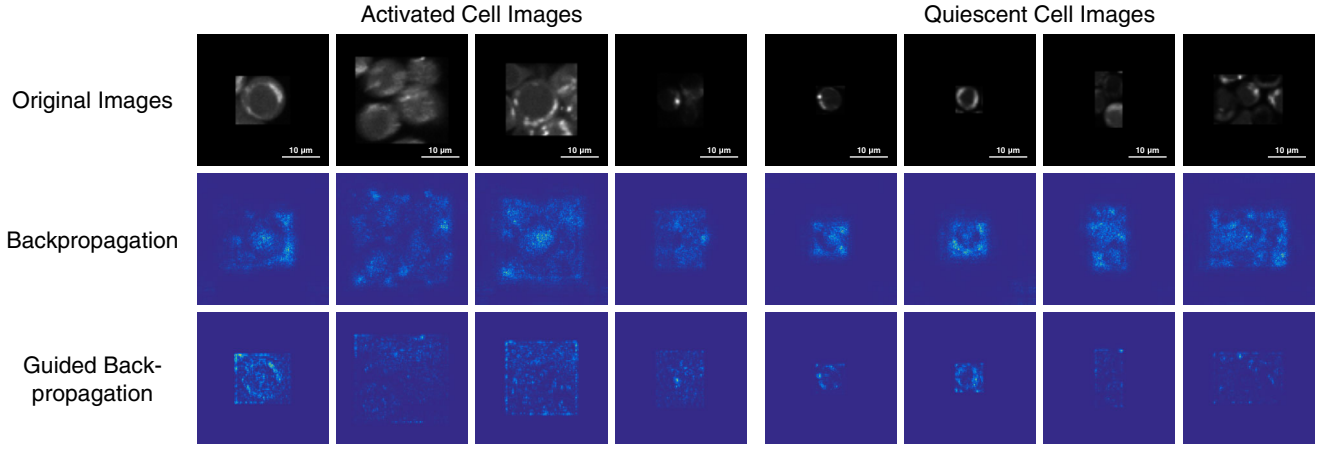


FIGURE 7 Saliency maps of randomly selected cell images from donor 1 (scale bar: 10 µm). The backpropagation and guided backpropagation rows show two different techniques for generating saliency maps from the same T cell images in the first row

multiple layers of the pre-trained CNN (Figures S3 and S4). The fine-tuning jobs took 5096 hours (212 days) in total to train on graphics processing units (GPUs). Therefore, we do not train each model multiple times to assess the variability in model performance due to random sampling, computer hardware, non-deterministic algorithms and other factors. Slight differences in performance should not be over-interpreted.

Based on the misclassified images, the performance of the pre-trained CNN model with fine-tuning is limited by the image cropping quality. Some images contain multiple cells. Others do not contain any T cells. Developing a better filter to detect images with artifacts and adopting state-of-the-art segmentation approaches [16, 50] could further boost classification accuracy. We highlight the images that are misclassified by the pre-trained CNN with fine-tuning on the UMAP plots for three alternative image representations (Figures S12-S14). Some of these misclassified images are better aligned with other images from the correct class in these alternative feature spaces than in Figure 6. These images may be easier to classify correctly with the alternative image representations. Therefore, ensemble methods combining multiple image representations and classifiers may further improve performance. However, an ensemble learning approach would still be limited by the image cropping quality and any errors in the image labelling.

Although the pre-trained CNN with fine-tuning has strong predictive performance in this study, there are several caveats regarding how these results may translate to other autofluorescence intensity image classification tasks. We have a small set of donors. The nested cross-validation involves only five donors, and the generalization test uses a single held out donor. In addition, all donors are from a narrow, young and healthy population. They may not adequately represent the general public or cancer patients, and further study is needed to see if the model is still applicable in more challenging populations. Also, T cells in our study are isolated from the bulk blood cell population, and two subtypes of T cells are separated during data acquisition. It is possible to classify some blood cell types in a bulk population with only label-free autofluorescence parameters [51], but future work is needed to assess the performance of our approach on image samples with additional immune cells and mixed activation states. Finally, quantitative fluorescence intensity imaging has technical limitations, requiring consistent imaging settings such as illumination power and detector gain.

Future work also can assess how robust our trained CNN models are to more diverse imaging settings and whether new training strategies are required to adapt to other domains. To sort T cells in practice and improve cell manufacturing processes, the classifier would need to be coupled to a flow sorter. Transitioning from the current imaging platform to a commercial imaging flow cytometer would make our approach more widely available. However, imaging flow cytometry is typically single-photon and requires UV excitation to measure NAD(P)H intensity, which can be damaging to cells. In addition, lower resolution imaging and the flowing nature of the cells may make it more difficult to detect the intracellular structures of small T cells. Although the resulting images may be different enough to require a distinct CNN, recent advances in CNNs for imaging flow cytometry [10, 23, 24, 38, 52, 53] suggest our pipeline could be optimized for this practical setting. Overall, our strong results demonstrate the feasibility of classifying T cells directly from autofluorescence intensity images, which can guide future work to bring this technology to pre-clinical and clinical applications.

4 | METHODS

4.1 | Cell preparation and imaging

This study was approved by the Institutional Review Board of the University of Wisconsin-Madison (#2018–0103). Informed consent was obtained from all donors. All NAD(P)H intensity images were created from a subset of the NAD(P)H fluorescence lifetime images acquired in Walsh et al. [6]. No new images were generated for this study. The protocols summarized below are described in more detail in Walsh et al. [6].

CD3 and CD8 T cells were isolated using negative selection methods (RosetteSep, StemCell Technologies) from the peripheral blood of six healthy donors (three male, three female, mean age = 26). The T cells were divided into quiescent and activated groups. The quiescent group came directly from the body without any antibody or chemical treatment, whereas the activated group was stimulated with a tetrameric antibody against CD2, CD3 and CD28 (StemCell Technologies). CD69 immunofluorescence labelling in a subset of cells verified the quiescent (CD69-) and activated (CD69+) phenotypes due to the culture conditions. In our dataset, all treated cells were assigned to the activated class, which could introduce a small amount of noise in the class labels. T cell populations were cultured for 48 hours at 37C, 5% CO2 and 99% humidity and then plated before imaging. The T cells were not adherent or immobilized. For all donors, an equal number of quiescent and activated cells were initially cultured. However, different numbers of cells were imaged for each donor and activation state due to differences in the proliferation rates of quiescent and activated cells, differences in the clumping behaviors of the

cells, potential bias by the operator in the selection of the imaging fields of view and dilution of the cell concentration due to parallel experiments (particularly for the antibody experiments reported by Walsh et al. [6]). Therefore, there are different activated and quiescent class skews among donors (Table 1).

NAD(P)H intensity images were created by integrating the photon counts of fluorescence lifetime decays at each pixel within the fluorescence lifetime images acquired, as described by Walsh et al. [6]. Briefly, images were acquired using an Ultima (Bruker Fluorescence Microscopy) two-photon microscope coupled to an inverted microscope body (TiE, Nikon) with an Insight DS+ (Spectra Physics) as the excitation source. A 100X objective (Nikon Plan Apo Lambda, NA 1.45), lending an approximate field of view of 110 μm, was used in all experiments with the laser tuned to 750 nm for NAD(P)H two-photon excitation and a 440/80 nm bandpass emission filter in front of a GaAsP photomultiplier tube (PMT; H7422, Hamamatsu). Images were acquired for 60 seconds with a laser power at the sample of 3.0 to 3.2 mW and a pixel dwell time of 4.6 µs. Grayscale microscopy images were labeled with a deidentified donor ID and T cell activity state according to the culture conditions: quiescent for T cells not exposed to the activating antibodies or activated for T cells exposed to the activating antibodies.

4.2 | Image processing

We segmented cell images using CellProfiler [35]. Each cell was cropped according to the bounding box of its segmented mask. Cell short NAD(P)H lifetime was used to filter out other visually indistinguishable cells (eg, red blood cells) by removing cells with a mean fluorescence lifetime less than 200 ps. To remove very dim images and images containing no cells, we further filtered the segmented images by thresholding the combination of image entropy and total intensity (Figure S15). The threshold values in Figure S15 were chosen based on the distributions of entropy and intensity with a Gaussian approximation. This filter was conservative. We manually inspected the removed images to ensure none of them contained T cells.

Because the classifiers that used image pixels as input required uniform size and some required square images, we padded all activated and quiescent cell images with black borders. The padding size of 82×82 was chosen based on the largest image in the dataset after removing extremely large outliers. Also, we augmented the dataset by rotating each original image by 90, 180 and 270 degrees and also by flipping the original image horizontally and vertically, which added five extra images for each cell (Figure 1). We implemented this image processing pipeline using the Python package OpenCV [54].

4.3 | Nested cross-validation

We trained and evaluated eight classifiers of increasing complexity (Table 3). We used the same leave-one-donor-out test principle to measure the performance of all models. For example, when using donor 1 as the test donor, the frequency classifier counts the positive proportion among all images in the augmented dataset from donors 2, 3, 5 and 6. Then, it uses this frequency to predict the activity for all unaugmented images from donor 1. By testing in this way, the classification result tells us how well each model performs on images from new donors. Donor 4 was not included in this cross-validation because we randomly selected it as a complete hold-out donor. All images from donor 4 were only used after hyper-parameter tuning and model selection as a final independent test to assess the generalizability of our pipeline to a new donor.

Following the leave-one-donor-out test principle [31, 38], we wanted the selection of the optimal hyper-parameters to be generalizable to new donors as well. Therefore, we applied a nested cross-validation scheme [55, 55, 56] (Figure 8). For each test donor, within the inner loop we performed 4-fold cross-validation to measure the average performance of each hyper-parameter combination (grid search). Each fold in the inner loop cross-validation corresponds to one donor's augmented images. The outer cross-validation loop used the selected hyper-parameters from the inner loop cross-validation to train a new model with the four other donors' augmented images. We evaluated the trained model on the outer loop test donor. For models requiring early stopping, we constructed an early stopping set by randomly sampling one-fourth of the unaugmented images from the training set and removing their augmented copies. Then, training continued as long as the performance on images in the early stopping set improved. Similarly, we did not include augmented images in the validation set or the test set.

No single evaluation metric can capture all the strengths and weaknesses of a classifier, especially because our dataset was class imbalanced and not skewed in the same way for all donors. Therefore, we considered multiple evaluation metrics in the outer loop. Accuracy measures the percentage of correct predictions. It is easy to interpret, but it does not necessarily characterize a useful classifier. For example, when positive samples are rare, a trivial classifier that predicts all samples as negative yields high accuracy. Precision and recall (sensitivity), on the other hand, consider the costs of false positive and false negative predictions, respectively. Graphical metrics such as the ROC curve and PR curve avoid setting a specific classification threshold. We used AUC to summarize ROC curves and average precision for the PR curves. The ROC curve performance of a random classifier is independent

TABLE 3 The eight classifiers with their input features and hyper-parameters

Model	Description
Frequency Classifier	Predict class probability using the class frequencies in the training set.
Logistic Regression with Pixel Intensity	Regularized logistic regression model with pixel intensity as input. Regularization power λ of ℓ_1 penalty is tuned.
Logistic Regression with Total Intensity and Size	Regularized logistic regression model fitted with two numerical values: image total intensity and cell mask size. Regularization power λ of ℓ_1 penalty is tuned.
Logistic Regression with CellProfiler Features	Regularized logistic regression model fitted with 123 features extracted from CellProfiler related to intensity, texture and area. Regularization power λ of ℓ_1 penalty is tuned.
One-layer Fully Connected Neural Network	Fully connected one-hidden-layer neural network with pixel intensity as input. Number of neurons, learning rate and batch size are tuned.
LeNet CNN	CNN with the LeNet architecture with pixel intensity as input. Learning rate and batch size are tuned.
Pre-trained CNN Off-the-shelf Model	Freeze layers of a pre-trained Inception v3 CNN. Train a final added layer from scratch with extracted off-the-shelf features. Learning rate and batch size are tuned.
Pre-trained CNN with Fine-tuning	Fine-tune the last n layers of a pre-trained Inception v3 CNN. The number of layers n, learning rate and batch size are tuned.

of the class distribution, while the PR curve is useful when the classes are imbalanced [39]. For this reason, we used mean average precision of the inner loop 4-fold cross-validation to select optimal hyper-parameters.

During the nested cross-validation, we trained the LeNet CNN and pre-trained CNN with fine-tuning using GPUs. These jobs ran on GTX 1080, GTX 1080 Ti, K40, K80, P100 or RTX 2080 Ti GPUs. All other models were trained using CPUs.

4.4 | Linear classifiers

We used a trivial frequency classifier as a baseline model. This model computes the positive sample percentage in the training set. Then, it uses this frequency as a positive class prediction score (between 0 and 1) for all samples in the test set.

Logistic regression with Lasso regularization is a standard and interpretable statistical model used to classify microscopy images [8]. The Lasso approach reduces the number of effective parameters by shrinking the parameters of less predictive features to zero. These features are ignored when making a new prediction. We fitted and tested three Lasso logistic regression models with different types of features using the Python package scikit-learn [57]. An image intensity matrix with dimension 82×82 and values from 0 to 255, reshaped into a vector with length 6724, was used to fit the first model. The second model was trained with two scalar features, cell size and image total intensity, where cell size was computed using the pixel count in the cell mask generated by CellProfiler. The last model used 123 features relating to cell intensity, texture and area, which were extracted from cell images using a CellProfiler pipeline with modules *MeaureObjectSizeShape*, *MeasureObjectIntensity* and *MeasureTexture*. The Lasso regularization parameter λ was tuned for all three classifiers with nested cross-validation (Table S9). We also applied inverse class frequencies in the training data as class weights to adjust the imbalanced dataset.

4.5 | Simple neural network classifiers

We developed a fully connected neural network with one hidden layer (Figure S16) using the Python package Keras with the TensorFlow backend [58, 59]. The input layer uses the flattened image pixel vector with dimension 6724×1. Network hyper-parameters—number of hidden neurons, learning rate and batch size—were tuned using nested cross-validation (Table S9). The cross-entropy loss function was weighted according to the class distribution in the training set.

Also, we trained a CNN with the LeNet architecture [36] with randomly initialized weights (no pre-training). The LeNet architecture has two convolutional layers and two pooling layers (Figure S17). We used the default number of neurons specified in the original LeNet paper in each layer. The input layer was modified to support 82×82 one-channel images, so we could train this network with image pixel intensities. Similar to the fully connected neural network, we used nested cross-validation to tune the learning rate and batch size (Table S9) and applied class weighting. We used early stopping with a patience of 10 for both models, which means we stopped training if the loss function failed to improve on the early stopping set in 10 consecutive epochs.

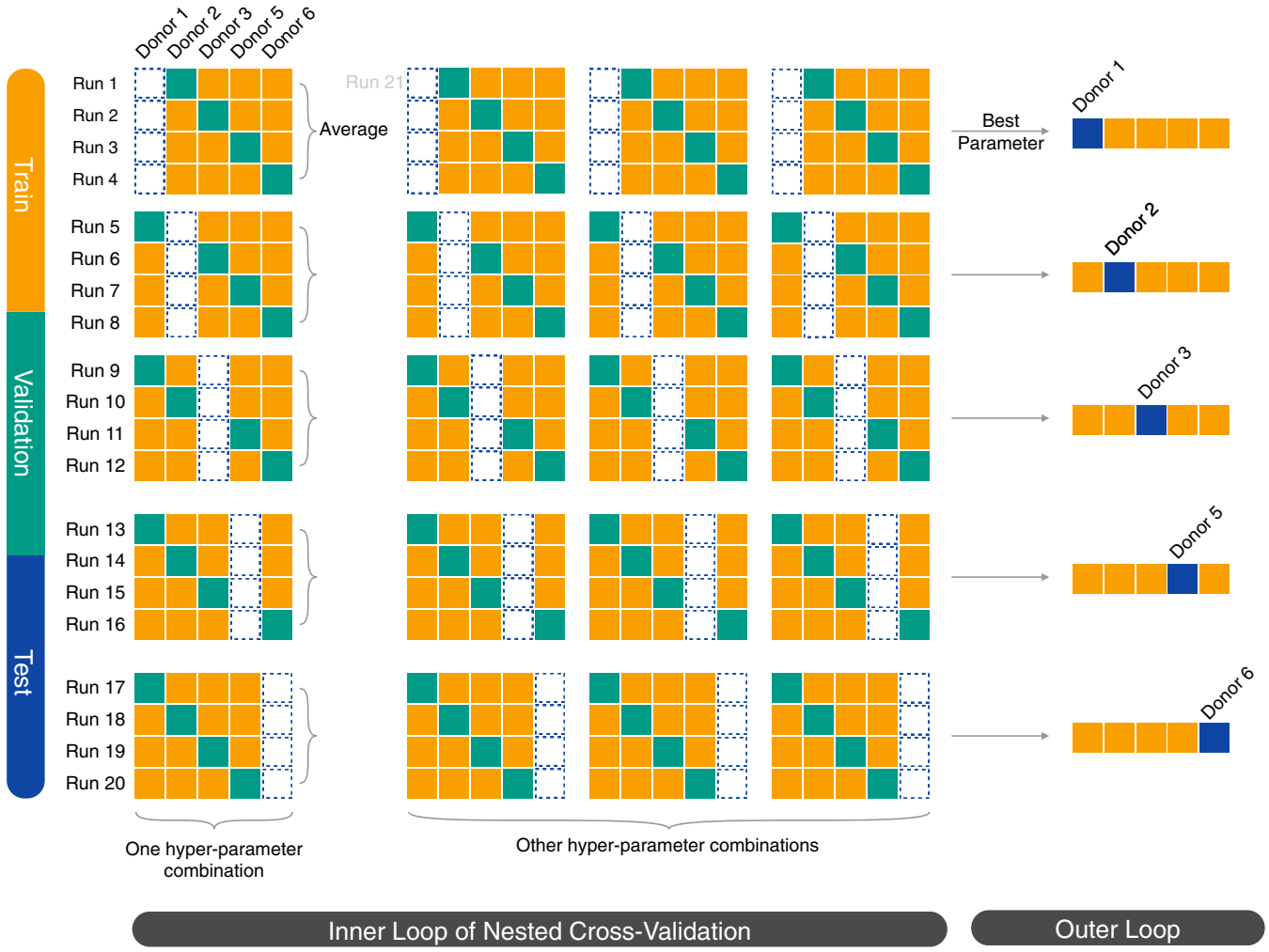


FIGURE 8 5 × 4 nested cross-validation scheme. For each test donor (blue), we used an inner cross-validation loop to optimize the hyper-parameters. We trained a model for each hyper-parameter combination using the training donors' augmented images (yellow) and selected the hyper-parameters that performed best on the validation donor's images (green). The validation donor is sometimes referred to as a tuning donor in cross-validation. Then, we trained a final model for each test donor using the selected hyper-parameters

4.6 | Pre-trained CNN classifiers

We developed a transfer learning classifier that uses the Inception v3 CNN with pre-trained ImageNet weights [60, 61]. Instead of training the whole network end-to-end from scratch, we took advantage of the pre-trained weights by extracting and modeling off-the-shelf features or fine-tuning the last n Inception modules, where n was treated as a hyper-parameter (Figure 9). Inception modules are mini-networks that constitute the overall Inception v3 architecture. Our first approach is a popular practice for transfer learning with Inception v3. We freeze the weights of all layers before the output layer and use them to extract generic image characteristics. Then, we train a light-weight classifier from scratch, specifically a neural network with an average pooling layer and a fully connected hidden layer with 1024 neurons, using these off-the-shelf features. We refer to this model as the pre-trained CNN off-the-shelf model.

An alternative is to fix some earlier layers and fine-tune the higher-level n layers by initializing them with pre-trained weights and continuing training on a new dataset. For this model, we modified the output layer to support binary classification, and we did not add new layers. In addition, we used the nested cross-validation scheme to optimize n along with the learning rate and batch size (Table S9), creating the pre-trained CNN with fine-tuning.

To implement these two pre-trained CNN models, we resized the padded cell images with bilinear interpolation to fit the input layer dimension (299 × 299 × 3) and generated three-channel images by merging three copies of the same grayscale image. For the pre-trained CNN with fine-tuning, we first used the resized cell images to

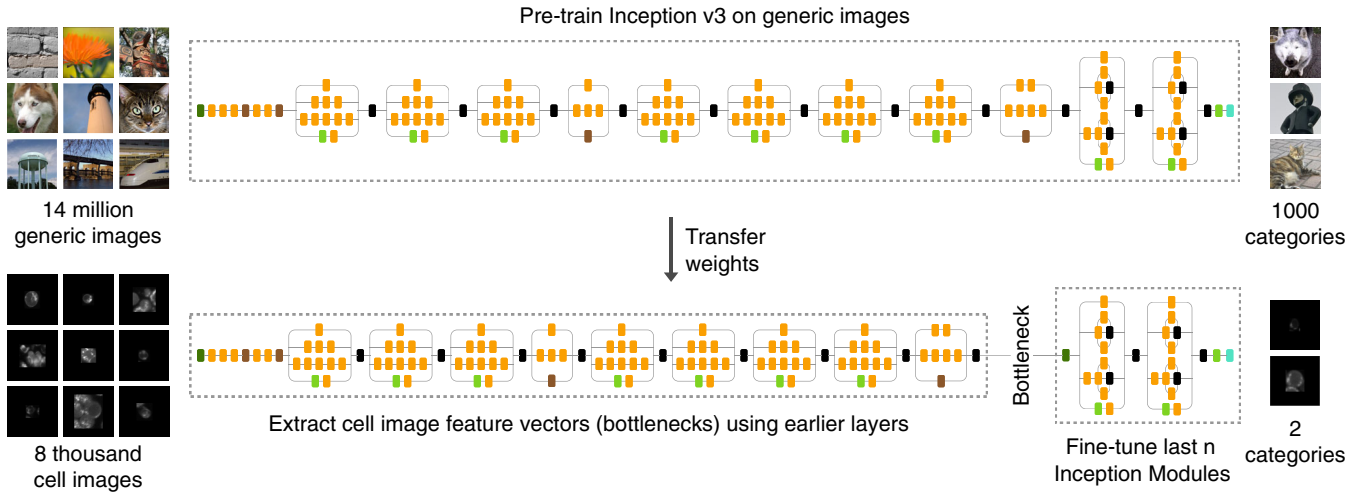


FIGURE 9 Fine-tuning the Inception v3 CNN to predict T cell activity. The example generic images are adapted from ImageNet

generate intermediate features ("bottlenecks"). Then, we used these features to fine-tune a sub-network. This approach significantly shortened the training time. Finally, we used class weighting and early stopping with a patience of 10 for both models. We implemented these two models using Keras with the TensorFlow backend.

4.7 | Pre-trained CNN interpretation

We implemented multiple approaches for interpreting the pre-trained CNNs. Computing classification confidence on misclassified images can help us understand why classifiers make certain errors. The Softmax score is sometimes used as a confidence prediction. Softmax is a function that maps the output real-valued number (Logit) from a neural network into a score between 0 and 1, which is then used to make a classification as a class probability. However, using the Softmax score from a neural network as a confidence calibration does not match the real accuracy [40]. Therefore, we used temperature scaling to better calibrate the predictions [40]. After training, for each donor, we optimized the temperature T on the nested cross-validation outer loop validation set. Then, we applied T to scale the Logit before Softmax computation and used the new Softmax score to infer classification confidence.

In addition to confidence calibration, we used dimension reduction to investigate the high-dimensional representations learned by our pre-trained CNN models. Dimension reduction is a method to project high-dimensional features into lower dimensions while preserving the characteristics of the data. Therefore, it provides a good way to visualize how trained models represent different cell image inputs. In our study, we choose UMAP [41, 62] as our dimension reduction algorithm. UMAP uses manifold learning techniques to reduce feature dimensions. It arguably preserves more of the global structure and is more scalable than the standard form of t-SNE [42], an alternative approach. Using UMAP, we projected the image features, extracted from the CNN layer right before the output layer, from 2048 dimensions to two dimensions. We used the default UMAP parameter values: "n_neighbors" as 15, "metrics" as "euclidean" and "min_dist" as 0.1. Then, we visualized and analyzed these projected features of T cell images using 2D scatter plots. When comparing UMAP with t-SNE, we used the default t-SNE parameters: "perplexity" as 30 and "metric" as "euclidean".

For the pre-trained CNN with fine-tuning, each test donor has different tuned hyper-parameters and a different fine-tuned CNN. Therefore, we performed feature extraction and dimension reduction independently for each test donor. There is no guarantee that these five scatter plots share the same 2D basis. In contrast, the image pixel features, CellProfiler features and off-the-shelf last layer features from a pre-trained CNN do not vary by test donor. For these three UMAP applications, we performed feature extraction and dimension reduction in one batch for all donors simultaneously.

Finally, we used saliency maps to further analyze what morphology features were used in classification [43]. A saliency map is a straightforward and efficient way to detect how prediction value changes with respect to a small change in the input cell image pixels. It is generated by computing the gradient of the output class score with respect to the input image. We compared two ways to compute this gradient: standard backpropagation and guided backpropagation [44]. Backpropagation is a method to calculate the gradient of loss function with

respect to the neural network's weights. Guided backpropagation is a variant that only backpropagates positive gradients. We generated saliency maps of the output layer for the pre-trained CNN with fine-tuning model for test donor 1 with a few randomly sampled images from the test set. The saliency map interpretations help us assess whether the classification basis is intuitive and whether the predictions derive from image artifacts instead of cell morphology.

4.8 | Software and data availability

Our GitHub repository https://github.com/gitter-lab/t-cell-classification contains Jupyter notebooks demonstrating how to run our Python code to pre-process images and train each of the classifiers. The notebooks can be run in a web browser using Binder and the links in the repository. The software is available under the BSD 3-Clause Clear License. This repository also contains a randomly selected subset of the T cell images that can be used to quickly test our software as well as the CellProfiler segmentation and feature extraction pipeline files. We archived the GitHub repository on Zenodo (DOI:10.5281/zenodo.3455314). In addition, our Zenodo dataset (DOI:10.5281/zenodo.2640835) contains bottleneck features from the Inception v3 model and trained model weights.

ACKNOWLEDGMENTS

We thank Tiffany Heaster for assistance with the T cell image processing; Quan Yin for CNN transfer learning advice; Shengchao Liu and Christine Walsh for general machine learning feedback; Katie Mueller, Steve Trier and Kelsey Tweed for discussion of the classification results; and Jaime Frey and Zach Miller for assistance with the Cooley cluster. This research was funded by NIH R01 CA205101, the UW Carbone Cancer Center Support Grant NIH P30 CA014520, the Morgridge Institute for Research and a UW-Madison L&S Honors Program Summer Senior Thesis Research Grant. In addition, this research benefited from GPU hardware from NVIDIA, resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under contract DE-AC02-06CH11357, the use of credits from the NIH Cloud Credits Model Pilot, a component of the NIH Big Data to Knowledge (BD2K) program, and the compute resources and assistance of the UW-Madison Center for High Throughput Computing (CHTC) in the Department of Computer Sciences. The CHTC is supported by UW-Madison, the Advanced Computing Initiative, the Wisconsin Alumni Research Foundation, the Wisconsin Institutes for Discovery and the National Science Foundation and is an active member of the Open Science Grid.

CONFLICT OF INTEREST

The authors and the Wisconsin Alumni Research Foundation have filed a provisional patent application based on these results.

ORCID

Zijie J. Wang https://orcid.org/0000-0003-4360-1423
Alex J. Walsh https://orcid.org/0000-0003-3832-8207
Melissa C. Skala https://orcid.org/0000-0002-6320-7637
Anthony Gitter https://orcid.org/0000-0002-5324-9833

REFERENCES

[1] E. L. Pearce, *Curr. Opin. Immunol.* **2010**, *22*, 314. https://doi.org/10.1016/j.coi.2010.01.018.
[2] D. M. Pardoll, *Nat. Rev. Cancer* **2012**, *12*, 252. https://doi.org/10.1038/nrc3239.
[3] N. P. Restifo, M. E. Dudley, S. A. Rosenberg, *Nat. Rev. Immunol.* **2012**, *12*, 269. https://doi.org/10.1038/nri3191.
[4] N. Marek-Trzonkowska, M. Myśliwiec, A. Dobyszuk, M. Grabowska, I. Techmańska, J. Juścińska, M. A. Wujtewicz, P. Witkowski, W. Młynarski, A. Balcerska, J. Myśliwska, P. Trzonkowski, *Diabetes Care* **2012**, *35*, 1817. https://doi.org/10.2337/dc12-0038.
[5] M. Cazaux, C. L. Grandjean, F. Lemaître, Z. Garcia, R. J. Beck, I. Milo, J. Postat, J. B. Beltman, E. J. Cheadle, P. Bousso, *J. Exp. Med.* **2019**, *216*, 1038. https://doi.org/10.1084/jem.20182375.
[6] A. Walsh, K. Mueller, I. Jones, C. M. Walsh, N. Piscopo, N. N. Niemi, D. J. Pagliarini, K. Saha, M. C. Skala, *bioRxiv* **2019**, 536813. https://doi.org/10.1101/536813.
[7] J. M. Szulczewski, D. R. Inman, D. Entenberg, S. M. Ponik, J. Aguirre-Ghiso, J. Castracane, J. Condeelis, K. W. Eliceiri, P. J. Keely, *Sci. Rep.* **2016**, *6*, 25086. https://doi.org/10.1038/srep25086.
[8] N. Pavillon, A. J. Hobro, S. Akira, N. I. Smith, *Proc. Natl. Acad. Sci.* **2018**, *115*, E2676. https://doi.org/10.1073/pnas.1711872115.
[9] J. Yoon, Y. J. Jo, M. H. Kim, K. Kim, S. Y. Lee, S. J. Kang, Y. K. Park, *Sci. Rep.* **2017**, *7*, 6654. https://doi.org/10.1038/s41598-017-06311-y.
[10] A. Gupta, P. J. Harrison, H. Wieslander, N. Pielawski, K. Kartasalo, G. Partel, L. Solorzano, A. Suveer, A. H. Klemm, O. Spjuth, I. M. Sintorn, C. Wählby, *Cytometry A* **2019**, *95*, 366. https://doi.org/10.1002/cyto.a.23701.
[11] M. Doan, A. E. Carpenter, *Nat. Mater.* **2019**, *18*, 414. https://doi.org/10.1038/s41563-019-0339-y.
[12] E. Moen, D. Bannon, T. Kudo, W. Graf, M. Covert, D. Van Valen, *Nat. Methods* **2019**, *1*. https://doi.org/10.1038/s41592-019-0403-1.
[13] W. J. Godinez, I. Hossain, S. E. Lazic, J. W. Davies, X. Zhang, *Bioinformatics* **2017**, *33*, 2010. https://doi.org/10.1093/bioinformatics/btx069.
[14] O. Dürr, B. Sick, *J. Biomol. Screen.* **2016**, *21*, 998. https://doi.org/10.1177/1087057116631284.
[15] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation. in *Medical*

Image Computing and Computer-Assisted Intervention – MICCAI 2015, vol. 9351 of Lecture Notes in Computer Science (Eds: N. Navab, J. Hornegger, W. M. Wells, A. F. Frangi), Springer International Publishing, Cham, Switzerland **2015**, p. 234. https://doi.org/10.1007/978-3-319-24574-4_28.

[16] D. Bannon, E. Moen, M. Schwartz, E. Borba, S. Cui, K. Huang, I. Camplisson, N. Koe, D. Kyme, T. Kudo, B. Chang, E. Pao, E. Osterman, W. Graf, D. Van Valen, *bioRxiv* **2018**, 505032. https://doi.org/10.1101/505032.

[17] M. Weigert, U. Schmidt, T. Boothe, A. Müller, A. Dibrov, A. Jain, B. Wilhelm, D. Schmidt, C. Broaddus, S. Culley, M. Rocha-Martins, F. Segovia-Miranda, C. Norden, R. Henriques, M. Zerial, M. Solimena, J. Rink, P. Tomancak, L. Royer, F. Jug, E. W. Myers, *Nat. Methods* **2018**, *15*, 1090. https://doi.org/10.1038/s41592-018-0216-7.

[18] T. Pärnamaa, L. Parts, *G3: Genes, Genomes, Genetics* **2017**, *7*, 1385. https://doi.org/10.1534/g3.116.033654.

[19] O. Z. Kraus, J. L. Ba, B. J. Frey, *Bioinformatics* **2016**, *32*, i52. https://doi.org/10.1093/bioinformatics/btw252.

[20] F. Buggenthin, F. Buettner, P. S. Hoppe, M. Endele, M. Kroiss, M. Strasser, M. Schwarzfischer, D. Loeffler, K. D. Kokkaliaris, O. Hilsenbeck, T. Schroeder, F. J. Theis, C. Marr, *Nat. Methods* **2017**, *14*, 403. https://doi.org/10.1038/nmeth.4182.

[21] M. Hofmarcher, E. Rumetshofer, D.-A. Clevert, S. Hochreiter, G. Klambauer, *J. Chem. Inf. Model.* **2019**, *59*, 1163. https://doi.org/10.1021/acs.jcim.8b00670.

[22] S. H. Karandikar, C. Zhang, A. Meiyappan, I. Barman, C. Finck, P. K. Srivastava, R. Pandey, *Anal. Chem.* **2019**, *91*, 3405. https://doi.org/10.1021/acs.analchem.8b04895.

[23] P. Eulenberg, N. Köhler, T. Blasi, A. Filby, A. E. Carpenter, P. Rees, F. J. Theis, F. A. Wolf, *Nat. Commun.* **2017**, *8*, 463. https://doi.org/10.1038/s41467-017-00623-3.

[24] N. Nitta, T. Sugimura, A. Isozaki, H. Mikami, K. Hiraki, S. Sakuma, T. Iino, F. Arai, T. Endo, Y. Fujiwaki, H. Fukuzawa, M. Hase, T. Hayakawa, K. Hiramatsu, Y. Hoshino, M. Inaba, T. Ito, H. Karakawa, Y. Kasai, K. Koizumi, S. W. Lee, C. Lei, M. Li, T. Maeno, S. Matsusaka, D. Murakami, A. Nakagawa, Y. Oguchi, M. Oikawa, T. Ota, K. Shiba, H. Shintaku, Y. Shirasaki, K. Suga, Y. Suzuki, N. Suzuki, Y. Tanaka, H. Tezuka, C. Toyokawa, Y. Yalikun, M. Yamada, M. Yamagishi, T. Yamano, A. Yasumoto, Y. Yatomi, M. Yazawa, D. Di Carlo, Y. Hosokawa, S. Uemura, Y. Ozeki, K. Goda, *Cell* **2018**, *175*, 266. https://doi.org/10.1016/j.cell.2018.08.028.

[25] C. L. Chen, A. Mahjoubfar, L. C. Tai, I. K. Blaby, A. Huang, K. R. Niazi, B. Jalali, *Sci. Rep.* **2016**, *6*, 21471. https://doi.org/10.1038/srep21471.

[26] M. D. Abràmoff, Y. Lou, A. Erginay, W. Clarida, R. Amelon, J. C. Folk, M. Niemeijer, *Investigative Opthalmology & Visual Science* **2016**, *57*, 5200. https://doi.org/10.1167/iovs.16-19964.

[27] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, R. M. Summers, ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. In 2017 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3462–3471 (IEEE, Honolulu, HI, 2017). https://doi.org/10.1109/CVPR.2017.369.

[28] M. Raghu, C. Zhang, J. Kleinberg, S. Bengio, *arXiv* **2019**, 1902.07208.

[29] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P. M. Agapow, M. Zietz, M. M. Hoffman, W. Xie, G. L. Rosen, B. J. Lengerich, J. Israeli, J. Lanchantin, S. Woloszynek, A. E. Carpenter, A. Shrikumar, J. Xu, E. M. Cofer, C. A. Lavender, S. C. Turaga, A. M. Alexandari, Z. Lu, D. J. Harris, D. DeCaprio, Y. Qi, A. Kundaje, Y. Peng, L. K. Wiley, M. H. S. Segler, S. M. Boca, S. J. Swamidass, A. Huang, A. Gitter, C. S. Greene, *Journal of The Royal Society Interface* **2018**, *15*, 20170387. https://doi.org/10.1098/rsif.2017.0387.

[30] M. Habibzadeh Motlagh, M. Jannesari, Z. Rezaei, M. Totonchi, H. Baharvand, Automatic white blood cell classification using pre-trained deep learning models: ResNet and Inception. in *Tenth International Conference on Machine Vision (ICMV 2017), 105* (Eds: J. Zhou, P. Radeva, D. Nikolaev, A. Verikas), SPIE, Vienna, Austria **2018**. https://doi.org/10.1117/12.2311282.

[31] H. T. H. Phan, A. Kumar, J. Kim, D. Feng. Transfer learning of a convolutional neural network for HEp-2 cell image classification. In 2016 *IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, 1208–1211 (IEEE, Prague, Czech Republic, 2016). https://doi.org/10.1109/ISBI.2016.7493483.

[32] A. Kensert, P. J. Harrison, O. Spjuth, *SLAS DISCOVERY: Advancing Life Sciences R&D* **2019**, *24*, 466. https://doi.org/10.1177/2472555218818756.

[33] C. Kandaswamy, L. M. Silva, L. A. Alexandre, J. M. Santos, *J. Biomol. Screen.* **2016**, *21*, 252. https://doi.org/10.1177/1087057115623451.

[34] N. Pawlowski, J. C. Caicedo, S. Singh, A. E. Carpenter, A. Storkey, *bioRxiv* **2016**, 085118. https://doi.org/10.1101/085118.

[35] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, P. Golland, D. M. Sabatini, *Genome Biol.* **2006**, *7*, R100. https://doi.org/10.1186/gb-2006-7-10-r100.

[36] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, *Proc. IEEE* **1998**, *86*, 2278. https://doi.org/10.1109/5.726791.

[37] J. Simm, G. Klambauer, A. Arany, M. Steijaert, J. K. Wegner, E. Gustin, V. Chupakhin, Y. T. Chong, J. Vialard, P. Buijnsters, I. Velter, A. Vapirev, S. Singh, A. E. Carpenter, R. Wuyts, S. Hochreiter, Y. Moreau, H. Ceulemans, *Cell Chem. Biol.* **2018**, *25*, 611. https://doi.org/10.1016/j.chembiol.2018.01.015.

[38] M. Nassar, M. Doan, A. Filby, O. Wolkenhauer, D. K. Fogg, J. Piasecka, C. A. Thornton, A. E. Carpenter, H. D. Summers, P. Rees, H. Hennig, *Cytometry A* **2019**, *95*, 836. https://doi.org/10.1002/cyto.a.23794.

[39] J. Lever, M. Krzywinski, N. Altman, *Nat. Methods* **2016**, *13*, 603. https://doi.org/10.1038/nmeth.3945.

[40] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger, *arXiv* **2017**, 1706.04599.

[41] L. McInnes, J. Healy, J. Melville, *arXiv* **2018**, 1802.03426.

[42] L. v. d. Maaten, G. Hinton, *J. Mach. Learn. Res.* **2008**, *9*, 2579.

[43] K. Simonyan, A. Vedaldi, A. Zisserman, *arXiv* **2013**, 1312.6034.

[44] J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, *arXiv* **2014**, 1412.6806.

[45] N. A. Ramey, C. Y. Park, P. L. Gehlbach, R. S. Chuck, *Photochem. Photobiol.* **2007**, *83*, 1325. https://doi.org/10.1111/j.1751-1097.2007.00162.x.

[46] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, B. Kim, Sanity Checks for Saliency Maps. in *Advances in Neural Information Processing Systems 31* (Eds: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett), Curran Associates, Inc., Montréal, Canada **2018**, p. 9505. https://papers.nips.cc/book/advances-in-neural-information-processing-systems-31-2018

[47] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, Jupyter development team, Jupyter Notebooks - a publishing format for reproducible computational workflows. in *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (Eds: F. Loizides, B. Scmidt), IOS Press, Amsterdam, Netherlands **2016**, p. 87. https://doi.org/10.3233/978-1-61499-649-1-87.

[48] Project Jupyter, M. Bussonnier, J. Forde, J. Freeman, B. Granger, T. Head, C. Holdgraf, K. Kelley, G. Nalvarte, A. Osherof, M. Pacer, Y. Panda, F. Perez, B. Ragan-Kelley, C. Willing, Binder 2.0 - Reproducible, interactive, sharable environments for science at scale. *Proceedings of the 17th Python in Science Conference* 113–120 (2018). https://doi.org/10.25080/Majora-4af1f417-011.

[49] E. Gavgiotaki, G. Filippidis, I. Zerva, G. Kenanakis, E. Archontakis, S. Agelaki, V. Georgoulias, I. Athanassakis, *J. Biophotonics* **2019**, *12*, e201800277. https://doi.org/10.1002/jbio.201800277.

[50] T. Falk, D. Mai, R. Bensch, Ö. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, A. Dovzhenko, O. Tietz, C. Dal Bosco, S. Walsh, D. Saltukoglu, T. L. Tay, M. Prinz, K. Palme, M. Simons, I. Diester, T. Brox, O. Ronneberger, *Nat. Methods* **2019**, *16*, 67. https://doi.org/10.1038/s41592-018-0261-2.

[51] B. P. Yakimov, M. A. Gogoleva, A. N. Semenov, S. A. Rodionov, M. V. Novoselova, A. V. Gayer, A. V. Kovalev, A. I. Bernakevich, V. V. Fadeev, A. G. Armaganov, V. P. Drachev, D. A. Gorin, M. E. Darvin, V. I. Shcheslavskiy, G. S. Budylin, A. V. Priezzhev, E. A. Shirshin, *Biomed. Opt. Express* **2019**, *10*, 4220. https://doi.org/10.1364/BOE.10.004220.

[52] Y. J. Heo, D. Lee, J. Kang, K. Lee, W. K. Chung, *Sci. Rep.* **2017**, *7*, 11651. https://doi.org/10.1038/s41598-017-11534-0.

[53] M. Lippeveld, C. Knill, E. Ladlow, A. Fuller, L. J. Michaelis, Y. Saeys, A. Filby, D. Peralta, *bioRxiv* **2019**, 680975. https://doi.org/10.1101/680975.

[54] G. Bradski, *Dr. Dobb's J. Soft. Tools* **2000**. https://github.com/opencv/opencv/wiki/CiteOpenCV

[55] S. Raschka, *arXiv* **2018**, 1811.12808.

[56] S. Varma, R. Simon, *BMC Bioinformat* **2006**, *7*, 91. https://doi.org/10.1186/1471-2105-7-91.

[57] 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, É. Duchesnay. *J. Mach. Learn. Res.* **2011**, *12*, 2825.

[58] F. Chollet, *Keras* **2015**. https://keras.io/

[59] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, *arXiv* **2016**, 1603.04467

[60] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, *arXiv* **2015**, 1512.00567.

[61] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database. *in 2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Piscataway, New Jersey **2009**. https://doi.org/10.1109/CVPR.2009.5206848.

[62] L. McInnes, J. Healy, N. Saul, L. Großberger, *J. Open Source Software* **2018**, *3*, 861. https://doi.org/10.21105/joss.00861.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Wang ZJ, Walsh AJ, Skala MC, Gitter A. Classifying T cell activity in autofluorescence intensity images with convolutional neural networks. *J. Biophotonics*. 2020;13:e201960050. https://doi.org/10.1002/jbio.201960050