Self-Supervised Deep-Learning Encodes High-Resolution Features of Protein Subcellular Localization

Hirofumi Kobayashi[1*], Keith C. Cheveralls[1], Manuel D. Leonetti[1*], & Loic A. Royer[1*]

[1] *CZ Biohub, San Francisco, USA.*

Abstract

Elucidating the diversity and complexity of protein localization is essential to fully understand cellular architecture. Here, we present *cytoself*, a deep learning-based approach for fully self-supervised protein localization profiling and clustering. *cytoself* leverages a self-supervised training scheme that does not require pre-existing knowledge, categories, or annotations. Applying *cytoself* to images of 1311 endogenously labeled proteins from the recently released OpenCell database creates a highly resolved protein localization atlas. We show that the representations derived from *cytoself* encapsulate highly specific features that can be used to derive functional insights for proteins on the sole basis of their localization. Finally, to better understand the inner workings of our model, we dissect the emergent features from which our clustering is derived, interpret these features in the context of the fluorescence images, and analyze the performance contributions of the different components of our approach.

Systematic and large-scale microscopy-based cell assays are becoming an increasingly important tool for biological discovery[1,2], playing a key role in drug screening[3,4], drug profiling[5,6], and for mapping sub-cellular localization of the proteome[7,8]. In particular, large-scale datasets based on immuno-fluorescence or endogenous fluorescent tagging comprehensively capture localization patterns across the human proteome[9,10]. Together with recent advances in computer vision and deep learning[11], such datasets are poised to help systematically map the cell's spatial architecture. This situation is reminiscent of the early days of genomics, when the advent of high-throughput and high-fidelity sequencing technologies was accompanied by the development of novel algorithms to analyze, compare, and categorize these sequences, and the genes therein. However, images pose unique obstacles to analysis. While sequences can be compared against a frame of reference (i.e. genomes), there are no such references for microscopy images. Indeed, cells exhibit a wide variety of shapes and appearances that reflect a plurality of states. This rich diversity is much harder to model and analyze than, for example, sequence variability. Moreover, much of this diversity is stochastic, posing the additional challenge of separating information of biological relevance from irrelevant variance. The fundamental computational challenge posed by image-based screens is therefore to extract well-referenced vectorial representations that faithfully capture only the relevant biological information and allow for quantitative comparison, categorization, and biological interpretation.

Previous approaches to classify and compare images have relied on engineered features that quantify different aspects of image content – such as cell size, shape and texture[12–15]. While these features are, by design, relevant and interpretable, the underlying assumption is that all the relevant features needed to classify an image can be identified and appropriately quantified. This assumption has been challenged by deep learning's recent successes[11]. On a wide range of computer vision and image classification tasks, hand-designed features cannot compete against learned features that are automatically discovered from the data itself[16,17]. In all cases, once features are available, the typical approach consists of boot-strapping the annotation process by either (i) unsupervised clustering techniques[18,19], or (ii) manual curation and supervised learning[20,21]. In the case of supervised approaches, human annotators examine images and assign localization, and once sufficient data is garnered, a machine learning model is trained in a supervised manner, and later applied to unannotated data[16,21,22]. Another approach consists of reusing models trained on natural images to learn generic features upon which supervised training can be bootstrapped[5,23]. While successful, these approaches suffer from potential biases, as manual annotation imposes our own preconceptions. Overall, the ideal algorithm should not rely on human knowledge or judgments, but instead automatically synthesize features and classify images without a priori assumptions – that is, solely on the basis of the images themselves.

Recent advances in computer vision and machine learning have shown that forgoing manual labeling is possible and nears the performance of supervised approaches[24,25]. Instead of annotating datasets, which is inherently non-scalable and labor-intensive, self-supervised models can be trained from large unlabeled datasets[26–30]. Self-supervised models are trained by formulating an auxiliary *pretext task*, typically one that withholds parts of the data and instructs the model to predict them [31]. This works because the task-relevant information within a piece of data is often distributed over multiple observed dimensions[27]. For example, given the picture of a car, we can recognize the presence of a vehicle even if many pixels are hidden, perhaps even when half of the image is occluded. Now consider a large dataset of pictures of real-world objects (e.g. ImageNet[32]). Training a model to predict missing parts from these images forces it to identify their characteristic features[30]. Once trained, the *representations* that emerge from pretext tasks capture the essential features of the data, and can be used to compare and

[*] Correspondence: hirofumi.kobayashi@czbiohub.org, manuel.leonetti@czbiohub.org, loic.royer@czbiohub.org

categorize images.

What first principles can underpin the self-supervised analysis, comparison, and classification of protein subcellular localization patterns? We know that protein localization is highly correlated with protein function and activity, yet localization patterns can also vary from cell to cell, depending on cell shape, density, cell state, etc. Therefore, when training a self-supervised model to distill a protein's localization signature, regardless of these variations, an effective strategy is to ensure that the model can identify a given labeled protein solely from fluorescence images. This is the key insight that underpins our work and from which our self-supervised model is derived.

Here, we describe the development, validation and utility of *cytoself*, a deep learning-based approach for fully self-supervised protein localization profiling and clustering. The key innovation is a pretext task that ensures that the localization features that emerge from different images of the same protein are sufficient to identify the target protein. We further demonstrate the ability of *cytoself* to reduce images to feature profiles characteristic of protein localization.

Results

A robust and comprehensive image dataset. A prerequisite to our deep-learning approach is a collection of high-quality images of fluorescently labeled proteins obtained under uniform conditions. Our OpenCell[10] dataset of live-cell confocal images of 1311 endogenously labeled proteins (opencell.czbiohub.org) meets this purpose. We reasoned that providing a fiducial channel could provide a useful reference frame for our model to capture protein localization. Hence, in addition to the labeled protein channel (mNeonGreen2), we also imaged a nuclear fiducial channel (Hoechst 33342). On average, we imaged the localization of a given protein in 18 fields of views, from each of which 45 cropped images containing 1-3 cells were extracted (for a total of 800 cropped images per protein). This scale, as well as the uniform conditions under which the images were collected, are important because our model must learn to ignore image variance and instead focus on protein localization. Finally, in our approach all images that represent the same protein are labeled by the same unique identifier (we used the protein's gene name, but the identifier can be arbitrary). This identifier does not carry any explicit localization information, nor is it linked to any metadata or annotations, but rather is used to link together all the different images that represent the localization of the same protein.

A Deep Learning model to identify protein localization features. Our deep learning model is based on the Vector Quantized Variational Autoencoder architecture (VQ-VAE[34,35]). In a classical VQ-VAE, images are encoded into a discrete *latent* representation, a vector, and then decoded to reconstruct the input image (see Fig. 1). The encoder and decoder are trained so as to minimize any distortion between input and output images. The discrete representation produced by the encoder is assembled by arraying a finite number of *symbols* (indices) that stand for vectors in a *codebook* (Fig. 1b, Supp. Fig.7). The codebook vectors themselves evolve during training so as to be most effective for the encoding-decoding task[34]. The latest incarnation of this architecture (VQ-VAE-2[33]) introduces a hierarchy of representations that operate at multiple spatial scales (termed VQ1 and VQ2 in the original VQ-VAE-2 study). We chose this architecture as a starting point because of the large body of evidence that suggests that quantized architectures currently learn the best image representations[34,35]. As shown in Fig. 1b we developed a variant that utilizes a split vector quantization scheme to improve quantization at large spatial scales (see methods section, Supp. Fig. 7).

Better protein localization encoding via self-supervision. Our model consists of two pretext tasks applied to each individual cropped image: First, it is tasked to encode and then decode the image (VQ-VAE). Second, it is tasked to predict the identifier associated with the image solely on the basis of the encoded representation. In other words, that second task aims to predict, for each single cropped image, which one of the 1,311 proteins in our library the image corresponds to. The first task forces our model to distill lower-dimensional representations of the images, while the second task forces these representations to be strong predictors of protein identity. This second task assumes that protein localization is the primary image information that is correlated to protein identity. Therefore, predicting the identifier associated with each image is key to encourage our model to learn localization-specific representations. Interestingly, it is acceptable, and in some cases perfectly reasonable, for these tasks to fail. For example, when two proteins have identical localization, it is impossible to resolve the identity of the tagged proteins from images alone. Moreover, the autoencoder might be unable to perfectly reconstruct an image from the intermediate representation, when constrained to make that representation maximally predictive of protein identity. It follows that the real output of our model is not the reconstructed image, nor the predicted identity of the tagged protein, but instead the distilled image representations, which we refer to as 'localization encodings' obtained as a necessary by-product of satisfying both the pretext tasks. More precisely, our model encodes for each image two representations that correspond to two different spatial scales: the global and local representations. The global representation captures large-scale image structure with each representation being a scaled-down 4×4 pixels image with 576 features (values) per pixel. The local representation captures finer spatially resolved details with each representation being a 25×25 pixels image with 64 features per pixel. We use the global representations to perform localization clustering, and the local representations to provide a finer and spatially resolved decomposition of protein localization. Overall, imposing the two pretext tasks defines a set of localization features capable of quantitatively and precisely representing protein localization patterns within cells. It follows that features

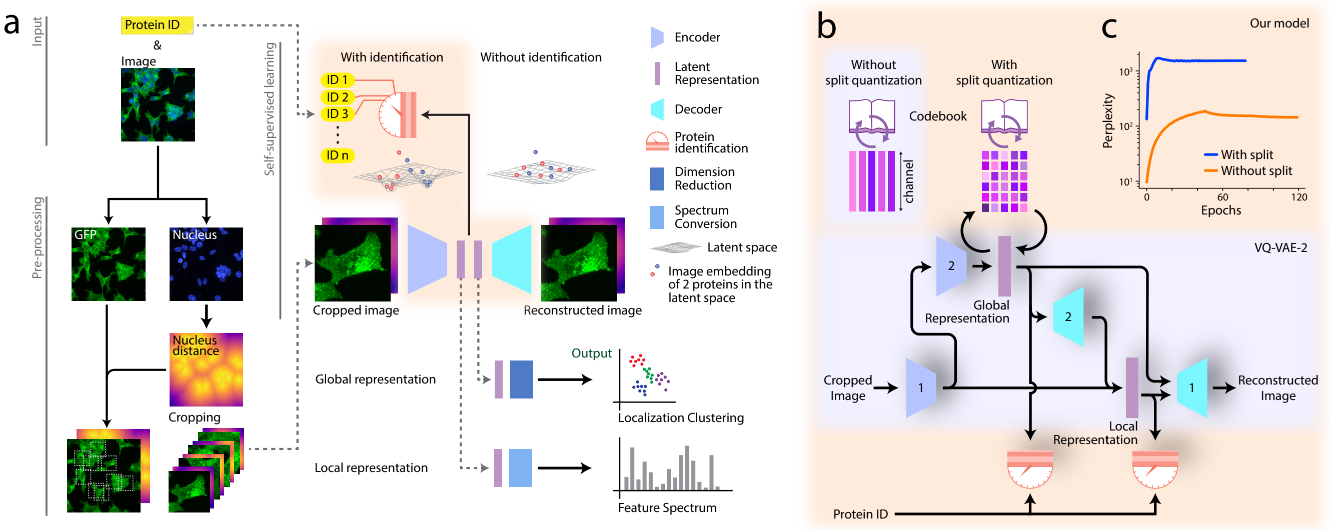

Figure 1: Self-supervised deep learning of protein subcellular localization with *cytoself*. **(a)** Workflow of the learning process. Only images and the identifiers of proteins are required as input. We trained our model with a fiducial channel, but its presence is optional as its performance contribution is negligible (see Table. 1). The protein identification pretext task ensures that images corresponding to the same or similar proteins have similar representations. **(b)** Architecture of our VQ-VAE-2[33] based Deep Learning model featuring our two innovations: split-quantization and protein identification pretext task. Numbers in the encoders and decoders indicate *encoder1*, *encoder2*, *decoder1* or *decoder2* in Supp. Fig. 10. **(c)** The level of utilization of the codebook (i.e. perplexity) increases with each training iteration and is enhanced by applying split quantization.

identified by *cytoself* can create a high-resolution protein localization atlas.

Mapping protein localization with *cytoself*. Obtaining image representations that are highly correlated with protein localization and invariant to other sources of heterogeneity (i.e. cell state, density, and shape) is only the first step for biological interpretation. Indeed, while these representations are lower dimensional than the images themselves, they still have too many dimensions for direct inspection and visualization. Therefore, we performed dimensionality reduction using the Uniform Manifold Approximation and Projection (UMAP) algorithm on the set of global localization-encodings obtained from all images (see methods). The result is visualized as a scatterplot (Fig. 2) in which each point represents a single (cropped) image in our dataset (test set only, 10% of entire dataset) to generate a highly detailed map reflecting the full diversity of protein localizations. The resulting UMAP corresponds to a protein localization atlas that reveals a hierarchy of clusters and sub-clusters reflective of the subcellular architecture. We can evaluate and explore this map by labeling each protein according to its sub-cellular localization obtained from manual annotations of the proteins in our image dataset (Supp. File 2). The most pronounced delineation corresponds to nuclear versus non-nuclear localizations (Fig. 2, top right and bottom left, respectively). Within the nuclear cluster, sub-clusters are resolved that correspond to nucleoplasm, chromatin, nuclear membrane, and the nucleolus. Strikingly, within each region, tight clusters that correspond to specific cellular functions can be resolved. For example, subunits involved in splicing, transcription or nuclear import cluster tightly together (outlined in Fig. 3). Similarly, sub-domains emerge within the non-nuclear cluster, the largest corresponding to cytoplasmic and vesicular localizations. Within these domains are several very tight clusters corresponding to mitochondria, ER exit sites, ribosomes, clathrin coated vesicles, etc. (Fig. 2). Outside of these discrete localization domains, are many proteins which exhibit mixed localization patterns (see grey points in Fig. 2). Prominent among these is a band of proteins interspersed between the nuclear and non-nuclear regions (Fig. 3a). Progressing along this path from lower left to upper right are proteins having a mostly diffuse localization (e.g. FNBP1L, CHN1, and ATGB), followed by proteins with mixed localizations (e.g. RANBP9 and ANAPC4) and finally proteins with mostly diffuse nuclear localization (e.g. CDK2, POLR2B, and CHEK1). These results confirm that our model learns image representations that are accurate and high-resolution signatures of protein localization. Our feature embedding is qualitatively comparable to previous results obtained by supervised classification of protein localization[21]. However, in contrast to the extensive manual annotation required in previous studies, our approach is entirely self-supervised. Moreover, as described, it offers higher resolution clustering and the ability to extract highly resolved information on protein localization.

High resolution clustering identifies protein complexes. The resolving power of our approach is further illustrated by examining well known stable protein complexes and discrete cellular machines. For example, all subunits of the SF3 spliceosome, mediator, 20S proteasome, core RNA polymerase, nu-

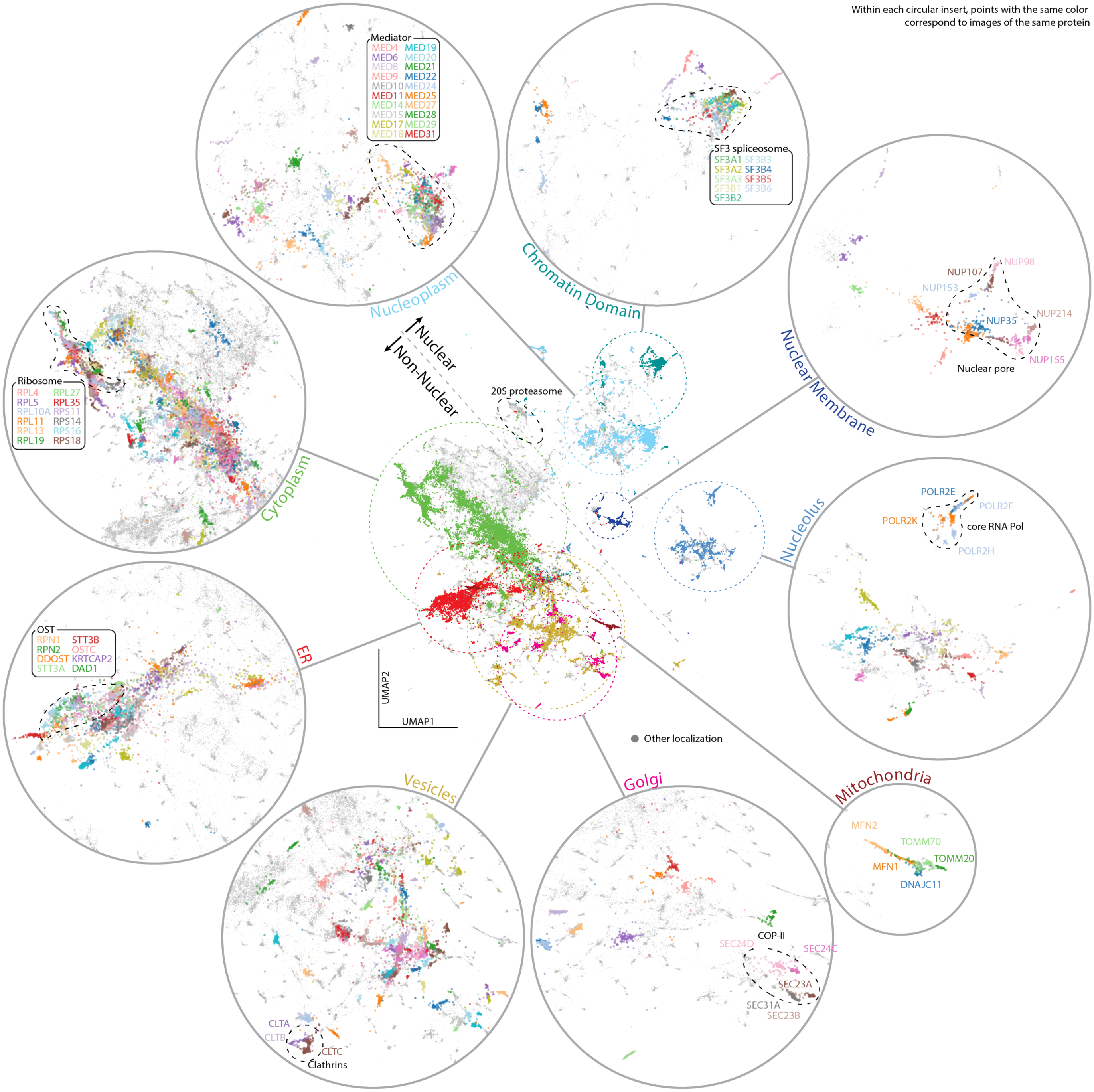

Figure 2: High-resolution Protein Localization Atlas. Each point corresponds to a single image from our test dataset of 109,751 images. To reveal the underlying structure of our map, each point in the central UMAP is colored according to 9 distinct protein localization families (mitochondria, vesicules, nucleoplasm, cytoplasm, nuclear membrane, ER, nucleolus, Golgi, chromatin domain). Tight clusters corresponding to functionally-defined protein complexes can be identified within each localization family. Only proteins with a clear and exclusive localization pattern are colored, other (grey) points correspond to proteins with other or mixed localizations. Within each localization domains, we further represent the resolution of *cytoself* representations by labeling the images corresponding to individual proteins in different colors (circular inserts). Note that while the colors in the central UMAP represent different cellular territories, colors in the inserts are only used to delineate individual proteins, and are not meant to be directly comparable.

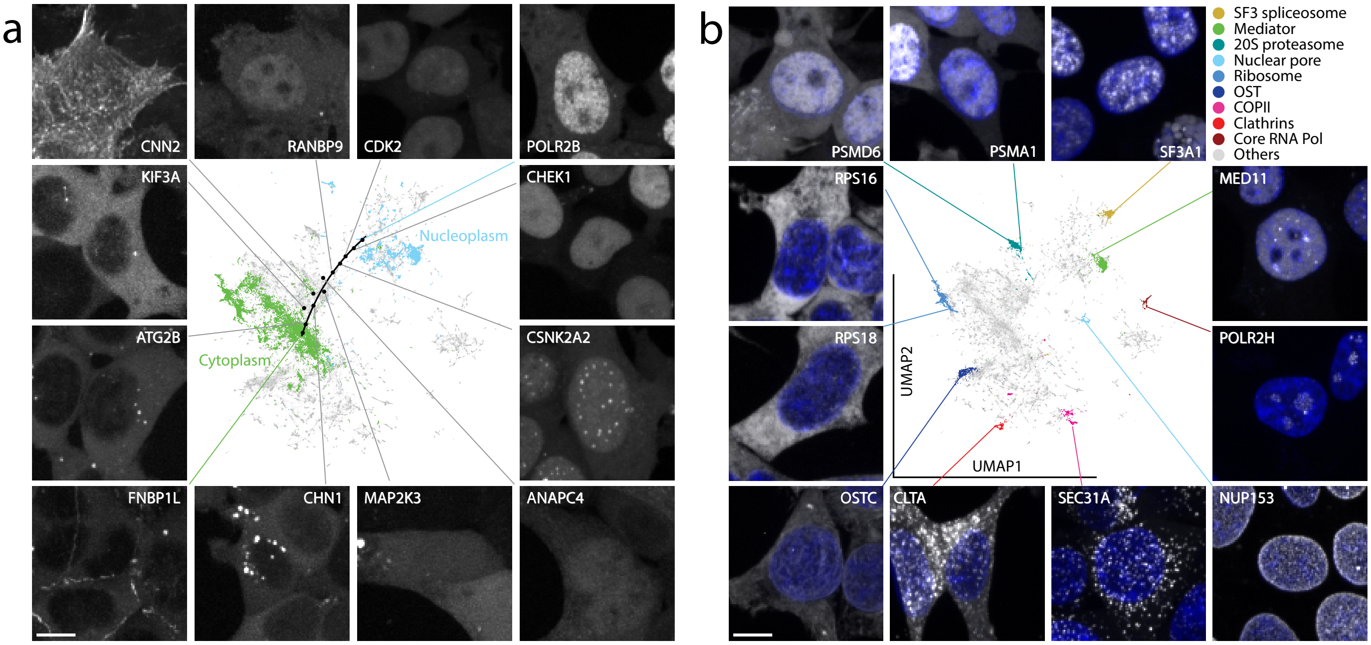

Figure 3: Exploring the Protein Localization Atlas. **(a)** Representative images of proteins localized within the transition 'grey' space between the cytoplasm and nucleoplasm regions. **(b)** The subunits of well-known and stable protein complexes tightly cluster together. Moreover, the complexes themselves are placed in their correct cellular contexts. Scale bars, 10 μm.

clear pore, ribosome, the cotranslational oligosaccharyltransferase complex (OST), as well as COPII, and clathrin-coats (full list of subunits per complex in Supp. File. 3) form tight and well-resolved clusters, which importantly, are placed within their respective cellular domains (Fig. 3b). Fluorescent images of 11 representative subunits from these complexes (SF3A1, MED11, PSMD1, PSMA1, POLR2H, NUP153, RSP16, RSP18, OSTC, SEC31A, CLTA) illustrate these discrete localization patterns. Notably, despite the diversity of cell shapes and sizes, the localization pattern for subunits of the same complex (e.g. the 20S proteasome subunits PSMD6 and PSMA1) converge. Thus, *cytoself* accurately identifies and spatially clusters protein complexes solely on the basis of the fluorescence images. An analysis of the relationship between localization patterns and protein-protein interactions is detailed in our companion study[10] and reveals that, by and large, the encodings derived form *cytoself* are sufficient to predict stable molecular interactions between proteins.

Extracting feature spectra for quantitative analysis of protein localization. We have shown that *cytoself* can generate a highly resolved map of protein localization on the basis of distilled image representations, i.e. each protein's 'localization encoding'. Can we dissect and understand the features that make up these representations and interpret their meaning? To answer this question, we created, a *feature spectrum* of the main components contributing to each protein's localization encoding. The spectra were constructed by calculating the histogram of codebook feature indices present in each image – as if each codebook feature was an ingredient present in the images at different concentrations (see Supp. Fig. 9, and Fig. 1a, and methods for details). To group related and possibly redundant features together, we performed hierarchical biclustering[36] (Fig. 4a), and thus obtained a meaningful linear ordering of features by which the spectra can be sorted. Plotting these results on a heatmap, reveals 11 feature clusters from the top levels of the feature hierarchy (Fig. 4a, bottom). To understand and interpret the image localization patterns represented by these clusters, we chose four representative images (as described in the methods section) from each (see bottom Fig. 4a and Supp. Fig. 11). These images illustrate the variety of distinctive localization patterns that are present at different levels across all proteins. For example, the features in the first clusters (*i*, *ii*, *iii*, and *iv*) corresponds to a wide range of diffuse cytoplasmic localizations. Cluster *v* features are unique to nucleolus proteins. Features making up cluster *vi* correspond to very small and bright punctate structures, which are often characteristic of centrosomes, vesicules, or cytoplasmic condensates. Clusters *vii*, *viii*, and *x* correspond to different types of nuclear localization patterns. Cluster *ix* are dark features corresponding to non-fluorescent background regions. Finally, cluster (*xi*) corresponds to a large variety of more abundant, punctate structures occurring throughout the cells, primarily vesicular, but also Golgi, mitochondria, cytoskeleton, and subdomains of the ER. The specific and high-resolution localization encoding required to place each protein in the UMAP results from quantitatively different contributions of these features. To make this analysis more quantitative we computed the average feature spectra for all proteins belonging to each localization family (see Fig. 4b), again using the manual annotations as reference (as in Fig. 3, Supp. File. 2). This analysis confirms that certain spectral clusters are specific to certain localization families and thus correspond to characteristic textures

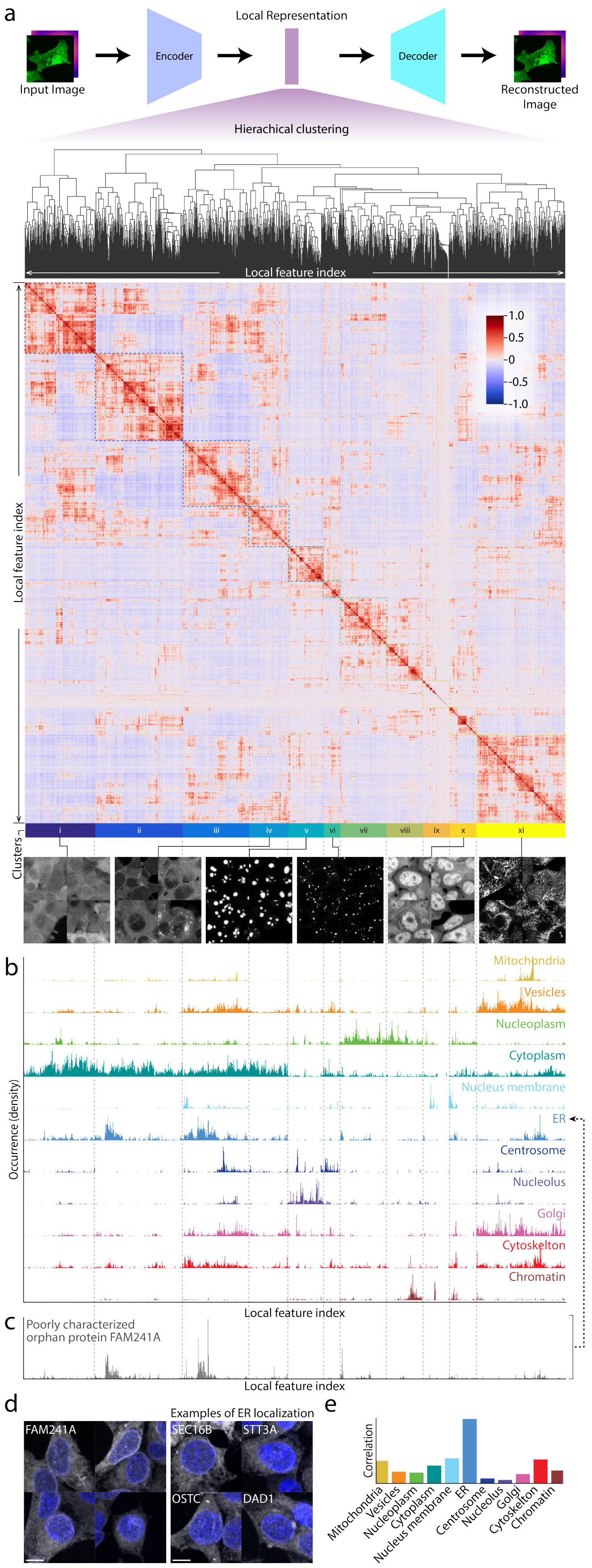

Figure 4: Feature spectral analysis.

Figure 4: (Continued) **(a)**, Quantized features in the local representation are reordered by hierarchical clustering to form a feature spectra (cf. Supp. Fig. 9). The color bar indicates the strength of correlation. Negative values indicate anti-correlation. On the basis of the feature clustering, we identified 11 primary top-level clusters, which are illustrated them with representative images (see also Supp. Fig. 11). **(b)**, Feature spectrum for each unique localization family. Occurrence indicates how many times a feature vector is found in the local representation of an image. **(c)**, The feature spectrum of FAM241A, a poorly characterized orphan protein. **(d)**, Fluorescence images for FAM241A versus representative images of other ER localized proteins. **(e)**, Correlation between FAM241A and other unique localization categories. All spectra, as well as the heatmap are vertically aligned.

and patterns in the images. For example, the highly specific chromatin and mitochondrial localizations both appear to elicit very narrow responses in their feature spectra. Finally, we ask whether this feature spectrum could be used to determine the localizations of proteins not present in our training data. For demonstration purposes, we computed the feature spectrum of FAM241A – a protein of unknown function (see our companion paper[10] for a alternate analysis). Visually (Fig. 4bc) and quantitatively (Fig. 4e), the spectrum of FAM241A is most correlated to the consensus spectrum of proteins belonging to the Endoplasmic Reticulum (see Fig. 4e). As shown in Fig. 4d, images for FAM241A do exhibit a localization pattern very similar to that of proteins belonging to the endoplasmic reticulum (ER). In our companion study[10], we show that FAM241A is in fact a new subunit of the OST (oligosaccharyltransferase) complex, responsible for co-translational glycosylation at the ER membrane.

Interpreting the features as patterns in the images. An important and very active area of research in deep learning is the visualization, interpretation, and reverse-engineering of the inner working of deep neural networks[37,38]. To better understand the relationship between our input images and the emergent features obtained by *cytoself*, we conducted an experiment in which we passed images into the autoencoder but prevented usage of a given feature by zeroing it before decoding. By computing the delta between the input and reconstructed images, we identify specific regions of the images that are impacted, and thus causally linked, to that feature. Three examples are illustrated in Fig.5: (a) POLR2E a core subunit shared between RNA polymerases I, II and III, (b) SEC22B a vesicle-trafficking protein, and (c) RPS18 a ribosomal protein. Highlighted in red on the images for each protein are the consequences of individually subtracting one of the three strongest peaks in their respective spectra. These difference maps reveal the image patterns that are lost and hence linked to that peak. The strongest peak (leftmost) of POLR2E's spectrum clearly corresponds to high intensity punctate structures within nucleoli, a localization recently established by Abraham et al.[39], while the two

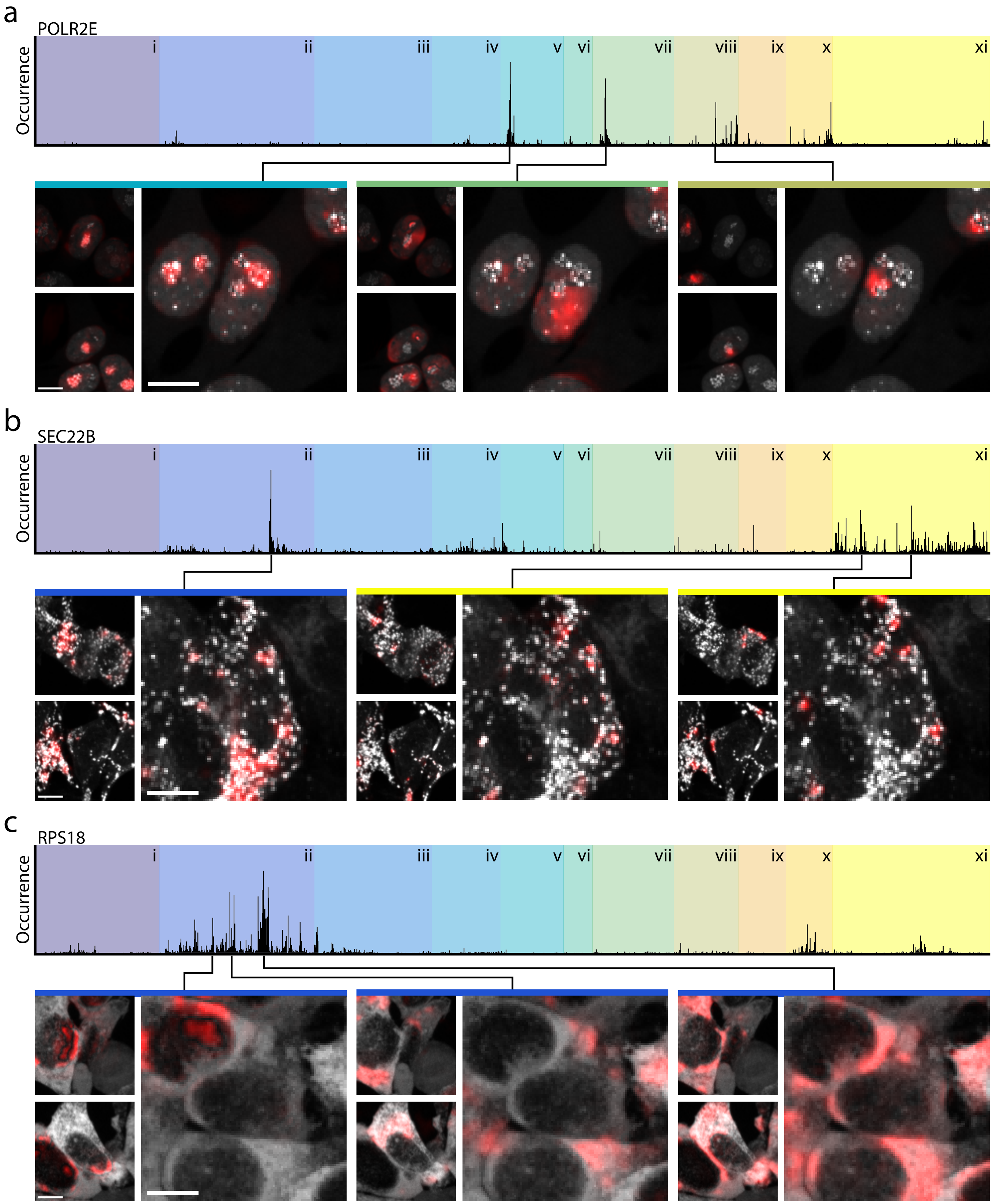

Figure 5: Interpreting image spectral features. Image spectra were computed for each example proteins **(a)** POLR2E, **(b)** SEC22B, and **(c)** RPS18. Subsequently, information derived from the indicated major peaks of their image spectra was removed by zeroing them out before passing the images again through the decoder. Highlighted in red are the differences between the original image and resulting output images for the corresponding features. The feature classes outlined in Fig. 4 are shown as background color for reference.

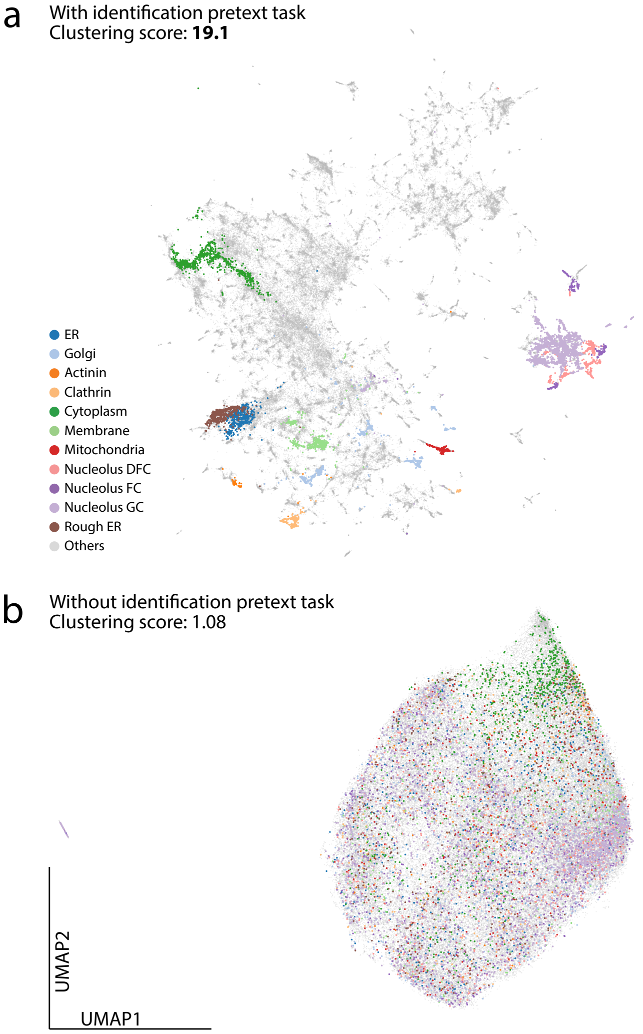

Figure 6: Identifying the essential components of our model. Protein localization maps were derived after removing individual components of our model (see Table. 1). A clustering score was calculated to compare maps. (a) Protein localization map derived by applying the complete model (b) Collapsed protein localization map derived when the protein identification pretext task is removed. See Supp. Fig. 8 for other examples.

other peaks correspond to lower intensity and more diffuse patterns. In the case of SEC22B the strongest peak (leftmost) corresponds to cytoplasmic regions with high-densities of vesicles. Other peaks of SEC22B's spectra corresponds to regions with sparse punctate expression. Finally, for RPS18, the strongest peak (rightmost) corresponds to large, diffuse, and uniform cytoplasmic regions in the images, whereas the two other selected peaks correspond to brighter and more speckled regions (middle) as well as regions adjacent to the nuclear boundary (leftmost). This analysis highlights both the interpretability but also the high complexity of the representations obtained.

Identifying the essential components of our model. To evaluate the impact of different aspects of our model on its clustering performance, we conducted an *ablation study*. For this, we retrained our model and recomputed the UMAP, after individually removing each component or input of our model, including: (a) the nuclear fiducial channel, (b) the distance transform applied to nuclear fiducial channel, (c) the split vector quantization, and (d) the identification pretext task. To quantitatively evaluate the effects of their ablation on our clustering results we developed a clustering score (see Methods section) on the basis of a curated dataset of proteins with exclusive localization patterns. As shown in Fig. 6 and Table. 1 the single most important component of *cytoself*, in terms of clustering performance, is the protein identification pretext task – the heart of our self-supervised approach. Removing that component leads to a complete collapse in performance (see Fig. 6 and Supp. Fig. 8). The next most impactful change is the use of a fiducial channel but without the distance transform. Removing split quantization also induced a small decrease in performance. Finally, forgoing the fiducial nuclear channel entirely led to the smallest decrease in clustering score, suggesting that our approach remains viable even in the absence of any fiducial marker – a notable advantage that widens the applicability of our approach and greatly simplifies experimental design. Overall, while all features contribute to the overall performance of our model, the identification pretext task is unsurprisingly the key and necessary ingredient.

Model variation	Clustering score
Full model	13.3 ± 2.4
Without nuclear channel	12.4 ± 1.3
Without split quantization	12.1 ± 1.46
Without distance transform	11.2 ± 1.3
Without id. pretext task	0.564 ± 0.14

Table 1: Clustering performance quantifies the effect of removing the indicated aspects of our model on its performance. We train the models 5 times, compute 10 different UMAPs per trained model, and then report mean and standard deviation ($\mu \pm \sigma$).

Discussion

We have shown that a self-supervised training scheme can produce image representations that capture the hierarchical organization of protein subcellular localization, solely on the basis of a large dataset of fluorescence images. Our model generates a high-resolution subcellular localization atlas capable of classifying not only discrete organelles, but also discrete protein complexes. Moreover, we can extract feature spectra to quantitatively define the composition in localization patterns of protein. Assuming that a protein's localization is highly correlated with its cellular function, *cytoself* will be an invaluable tool for functionally classifying many unknown or poorly studied proteins, and for studying the effect of cellular perturbations and cell state changes on protein subcellular localization.

Compared to previous attempts, our method makes few assumptions, but imposes two pretext tasks. Of these, requiring the model to identify proteins based solely on their localization encodings was essential. We also included Hoescht DNA-staining as a second fiducial marker, assuming that this would provide a spatial reference frame against which to interpret localization. Surprisingly however, this added little to the accuracy of our model. By comparison, Lu et al.[29] applied a pretext task that predicts a cell's fluorescence pattern from fiducial channels and from other cells in the same field of view. This assumes that fiducial channels are available, and that protein fluorescence is always sufficiently correlated to these fiducials. In contrast, our approach only requires a single fluorescence channel and sufficiently many images per labeled protein.

While powerful, there remains a few avenues for further development of *cytoself*. For example, we trained our model using a maximum intensity projection confocal images and have not yet leveraged the full 3D image data available in OpenCell[10]. The third dimension might confer an advantage for specific protein localization patterns that only vary along the basal-apical axis of the cells. Other important topics to explore are the automatic suppression of residual batch effects, improved cell segmentation via additional fiducial channels, as well as automatic rejection of anomalous or uncharacteristic cells from our training dataset. More fundamentally, significant conceptual improvements will require an improved self-supervised model that explicitly disentangles cellular heterogeneity from localization diversity.

Novel methods are being developed to tackle the complexity and heterogeneity of cellular fluorescence images. Recent computational methods focus on specific cellular events, for example in a computational tour-de-force Cai et al.[8] develop an integrated map of the three-dimensional concentration of proteins during cell division. By performing a spatio-temporal registration, much of the variance in the images is eliminated, thus aiding comparison and analysis. In our case, while including a nuclear fiducial channel was not strictly necessary, applying a distance transform and thus creating a rudimentary cellular coordinate system improved clustering performance. More work is needed to understand how to aid our models to robustly filter out irrelevant information, and interpret potential relevant information from cell shape changes.

Arguably, our ability to generate data is exponentially outpacing the human ability to manually annotate it. Moreover, there is already ample evidence that abundance of image data has a *quality all its own*, i.e. increasing the size of an image dataset often has higher impact on performance than improving the algorithm itself[40]. Hence our conviction that self-supervision is key to fully harness the deluge of data produced by novel instruments, end-to-end automation, and high-throughput image-based assays.

Methods

Fluorescence image dataset. All experimental and imaging details can be found in our companion study[10]. Briefly, HEK293T cells were genetically tagged with split-fluorescent proteins (FP) using CRISPR-based techniques[41]. After nuclear staining with Hoechst 33342, live cells were imaged with a spinning-disk confocal microscope (Andor Dragonfly). Typically, 18 fields of view were acquired for each one of the 1311 tagged protein, for a total of 24,382 three-dimensional images of dimension $1024 \times 1024 \times 22$ voxels.

Image data pre-processing. Each 3D confocal images was first reduced to two dimensions using a maximum-intensity projection along the z-axis to obtain a single 2D image per field of view. To help our model make use of the nuclear fiducial label we applied a distance transform to a nucleus segmentation mask (see below). The distance transform is constructed so that pixels within the nucleus were assigned a positive value that represents the shortest distance from the pixel to the nuclear boundary, and pixel values outside of the nucleus were assigned a negative value that represents the shortest distance to the nuclear boundary (see Fig. 1a). For each dual-channel and full field-of-view image, multiple regions of dimension 100×100 pixels were computationally chosen so that at least one cell is present and centered, resulting in a total of 1,100,253 cropped images. Cells (and their nuclei) that are too close to image edges are ignored. The raw pixel intensities in the fluorescence channel are normalized between 0 and 1, and the nuclear distance channel is normalized between -1 and 1. Finally, we augmented our training data by randomly rotating and flipping the images.

Nucleus segmentation. Nuclei are segmented by first thresholding the nucleus channel (Hoechst staining) and then applying a custom iterative refinement algorithm to eliminate under segmentation of adjacent nuclei. In the thresholding step, a low-pass Gaussian filter is first applied, followed by intensity thresholding using a threshold value calculated by Li's iterative Minimum Cross Entropy method[42,43]. The resulting segmentation is refined by applying the following steps: (i) we generate a 'refined' background mask by thresholding the laplace transform at zero, (ii) we morphologically close this mask and fill holes to eliminate intra-nuclear holes or gaps (empirically, this requires a closing disk of radius at least 4 pixels), (iii) we multiply this 'refined' mask by the existing background mask to restore any 'true' holes/gaps that were present in the background mask, (iv) we generate a mask of local minima in the laplace transform, using an empirically-selected percentile threshold, and finally (v) we iterate over regions in this local-minima mask and remove them from the refined mask if they partially overlap with the background of the refined mask.

Detailed model architecture. All details of our model architecture are given in Suppl. Fig. 10 and diagrammed in Fig. 1b.

First, the input image ($100 \times 100 \times 2$ pixels) is fed to *encoder1* to produce a set of latent vectors which have two destinations: *encoder2* and *VQ1 VectorQuantizer* layer. In the *encoder2*, higher level representations are distilled from these latent vectors and passed to the output. The output of *encoder2* is quantized in the *VQ2 VectorQuantizer* layer to form what we call in this work the global representation. The global representation is then passed to the *fc2* classifier for purposes of the classification pretext task. It is also passed on to *decoder2* to reconstruct the input data of *encoder2*. In this way, *encoder2* and *decoder2* form an independent autoencoder. The function of layer *mselyr1* is to adapt the output of *decoder2* to match the dimensions of the output of *encoder1*, which is identical to the dimensions of the input of *encoder2*. In the case of the *VQ1 VectorQuantizer* layer, vectors are quantized to form what we call the local representations. The local representation is then passed to the *fc1* classifier for purposes of the classification pretext task, as well as concatenated to the global representation that is resized to match the local representations' dimensions. The concatenated result is then passed to the *decoder1* to reconstruct the input image. Here, *encoder1* and *decoder1* form another autoencoder.

Split quantization. In the case of our global representation, we observed that the high level of spatial pooling required (4×4 pixels) led to codebook under-utilization because the quantized vectors have too many dimensions relative to their number (Fig. 1b). To solve this challenge we introduce the concept of *split quantization*. Instead of quantizing all the dimensions of a vector at once, we first split the vectors into subvectors of equal length, and then quantize each sub-vectors using a shared codebook. The main advantage of split quantization when applied to the VQ-VAE architecture is that one may vary the degree of spatial pooling without changing the total number of quantized vectors per representation. In practice, to maintain the number of quantized vectors while increasing spatial pooling, we simply split along the channel dimension. We observed that the global representations' perplexity, which indicates the level of utilization of the codebook, substantially increases when split quantization is used compared to standard quantization (Fig. 1c). As shown in Supp. Fig. 7, split quantization is performed along the channel dimension by splitting each channel-wise vector into nine parts, and quantizing each of the resulting 'sub-vectors' against the same codebook. Split quantization is only needed for the global representation.

Global and local representations. The dimensions of the global and local representations are $4 \times 4 \times 576$ and $25 \times 25 \times 64$ voxels, respectively. These two representations are quantized with two separate codebooks consisting of 2048 64-dimensional features (or codes).

Identification pretext task. The part of our model that is tasked with identifying the protein identifier is implemented as a 2-layer perceptron built by alternatively stacking fully connected layers with 1000 hidden units and non-linear ReLU layers. The output of the classifier is a one-hot encoded vector for which each coordinate corresponds to one of the 1311 proteins. We use categorical cross entropy as classification loss during training.

Computational efficiency. Due to the large size of our image data (1,100,253 cropped images of dimensions $100 \times 100 \times 2$ pixels) we recognized the need to make our architecture more efficient and thus allow for more design iterations. We opted to implement the encoder using principles from the *EfficientNet* architecture in order to increase computational efficiency without loosing learning capacity[44]. Specifically, we split the model of *EfficientNetB0* into two parts to make the two encoders in our model (Supp.Fig. 10). While we did not notice a loss of performance for the encoder, we did observed that *EfficientNet* did not perform as well for decoding. Therefore, we opted to keep a standard architecture based on a stack of residual blocks for the decoder[45]

Training protocol The whole dataset (1,100,253 cropped images) were split into training, validation and testing data by 8:1:1. All results shown in the figures are from testing data. We used the Adam optimizer with the initial learning rate of 0.0004. The learning rate was multiplied by 0.1 every time the validation loss did not improve for 4 epochs, and the training was terminated when the validation loss did not improve for more than 12 consecutive epochs.

Dimensionality reduction and clustering. Dimensionality reduction is performed using Uniform Manifold Approximation and Projection (UMAP)[46] algorithm. We used the reference open-source python package *umap-learn* (version 0.5.0) with default values for all parameters (i.e. the Euclidean distance metric, 15 nearest neighbors, and a minimal distance of 0.1).

Clustering score. We use a list of hand-picked proteins that have well-defined localizations (Supp. File 1) as the ground truth to compute clustering scores and thus evaluate the quality of clustering. To calculate a clustering score, we assume a collection of n points (vectors) in $\mathbb{R}^m$: $S = \{x_i \in \mathbb{R}^m | 0 \leq i \leq n\}$, and that we have a (ground truth) assignment of each point to a class forming a partition:

$$S = \bigcup_j C_j$$

Ideally, the vectors x_i are such that all points in a class are tightly grouped together, and that the centroids of each class are as far apart from each other as possible. This intuition is captured in the following definition of our clustering score:

$$\Gamma(C_i) = \frac{\sigma^*(\{\mu^*(C_j)\}_j)}{\mu^*(\{\sigma^*(C_j)\}_j)}$$

Where $\{.\}_k$ denotes the set of values obtained by evaluating the expression for each value of parameter k, and where μ^*

and σ^* stand for the robust mean (median) and robust standard deviation (computed using medians).

Clustering of protein subunits. The list of known protein complexes and their subunits used to demonstrate clustering performance (Fig. 3b) can be found in Supp. File 3.

Feature spectrum. Supp. Fig. 9a illustrates the workflow for constructing the feature spectra. Specifically, we first obtain the indices of quantized vectors in the latent representation for each image crop, and then calculate the histogram of indices in all images of each protein. As a result, we obtain a matrix of histograms in which rows correspond to protein identification (ID) and columns to the feature indices (Supp. Fig. 9b). At this point, the order of the columns (that is, the feature indices) is arbitrary. Yet, different features might be highly correlated and thus either related or even redundant (depending on how "saturated" the codebook is). To meaningfully order the feature indices, we compute the Pearson correlation coefficient between the feature index "profiles" (the columns of the matrix) for each pair of feature indices to obtain a 2048×2048 pairwise correlation matrix (see Supp. Fig. 9c). Next we perform hierarchical biclustering in which the feature indices with the most similar profiles are iteratively merged[47]. The result is that features that have similar profiles are grouped together (Supp. Fig. 9d). This ordering yields a more meaningful and interpretable view of the whole spectrum of feature indices. We detect 11 major feature clusters (ordered *i* through *xi*). Finally, for a given protein, we can produce a interpretable feature spectrum by ordering the horizontal axis of the quantized vectors histogram in the same way.

Software and hardware All deep learning architectures were implemented in TensorFlow 1.15[48] on Python 3.7. Training was performed on NVIDIA V100-32GB GPUs.

Bibliography

1. Pepperkok, R. & Ellenberg, J. High-throughput fluorescence microscopy for systems biology. *Nature reviews Molecular cell biology* **7**, 690–696 (2006).
2. Chandrasekaran, S. N., Ceulemans, H., Boyd, J. D. & Carpenter, A. E. Image-based profiling for drug discovery: due for a machine-learning upgrade? *Nature Reviews Drug Discovery* 1–15 (2020).
3. Boutros, M., Heigwer, F. & Laufer, C. Microscopy-based high-content screening. *Cell* **163**, 1314–1325 (2015).
4. Abraham, V. C., Taylor, D. L. & Haskins, J. R. High content screening applied to large-scale cell biology. *Trends in biotechnology* **22**, 15–22 (2004).
5. Scheeder, C., Heigwer, F. & Boutros, M. Machine learning and image-based profiling in drug discovery. *Current opinion in systems biology* **10**, 43–52 (2018).
6. Loo, L.-H., Wu, L. F. & Altschuler, S. J. Image-based multivariate profiling of drug responses from single cells. *Nature methods* **4**, 445–453 (2007).
7. Huh, W.-K. *et al.* Global analysis of protein localization in budding yeast. *Nature* **425**, 686–691 (2003).
8. Cai, Y. *et al.* Experimental and computational framework for a dynamic protein atlas of human cell division. *Nature* **561**, 411–415 (2018).
9. Thul, P. J. *et al.* A subcellular map of the human proteome. *Science* **356** (2017).
10. Cho, N. H. *et al.* Opencell: proteome-scale endogenous tagging enables the cartography of human cellular organization. *bioRxiv* (2021).
11. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *nature* **521**, 436–444 (2015).
12. Perlman, Z. E. *et al.* Multidimensional drug profiling by automated microscopy. *Science* **306**, 1194–1198 (2004).
13. Carpenter, A. E. *et al.* Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome biology* **7**, 1–11 (2006).
14. Yin, Z. *et al.* A screen for morphological complexity identifies regulators of switch-like transitions between discrete cell shapes. *Nature cell biology* **15**, 860–871 (2013).
15. Bray, M.-A. *et al.* Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nature protocols* **11**, 1757 (2016).
16. Eulenberg, P. *et al.* Reconstructing cell cycle and disease progression using deep learning. *Nature Communications* **8**, 463 (2017).
17. Caicedo, J. C. *et al.* Data-analysis strategies for image-based cell profiling. *Nature methods* **14**, 849–863 (2017).
18. Sailem, H., Bousgouni, V., Cooper, S. & Bakal, C. Cross-talk between rho and rac gtpases drives deterministic exploration of cellular shape space and morphological heterogeneity. *Open biology* **4**, 130132 (2014).
19. Traag, V. A., Waltman, L. & Van Eck, N. J. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports* **9**, 1–12 (2019).
20. Jones, T. R. *et al.* Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. *Proceedings of the National Academy of Sciences* **106**, 1826–1831 (2009).
21. Ouyang, W. *et al.* Analysis of the human protein atlas image classification competition. *Nature methods* **16**, 1254–1261 (2019).
22. Blasi, T. *et al.* Label-free cell cycle analysis for high-throughput imaging flow cytometry. *Nature communications* **7**, 1–9 (2016).

23. Pawlowski, N., Caicedo, J. C., Singh, S., Carpenter, A. E. & Storkey, A. Automating morphological profiling with generic deep convolutional networks. *BioRxiv* 085118 (2016).

24. Goyal, P. *et al.* Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988* (2021).

25. Holmberg, O. G. *et al.* Self-supervised retinal thickness prediction enables deep learning from unlabelled data to boost classification of diabetic retinopathy. *Nature Machine Intelligence* **2**, 719–726 (2020).

26. Hadsell, R. *et al.* Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics* **26**, 120–144 (2009).

27. Batson, J. & Royer, L. Noise2self: Blind denoising by self-supervision. In *International Conference on Machine Learning*, 524–533 (PMLR, 2019).

28. Kobayashi, H. *et al.* Intelligent whole-blood imaging flow cytometry for simple, rapid, and cost-effective drug-susceptibility testing of leukemia. *Lab on a Chip* **19**, 2688–2698 (2019).

29. Lu, A. X., Kraus, O. Z., Cooper, S. & Moses, A. M. Learning unsupervised feature representations for single cell microscopy images with paired cell inpainting. *PLoS computational biology* **15**, e1007348 (2019).

30. Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607 (PMLR, 2020).

31. Kolesnikov, A., Zhai, X. & Beyer, L. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1920–1929 (2019).

32. Deng, J. *et al.* Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255 (Ieee, 2009).

33. Wu, H. & Flierl, M. Vector quantization-based regularization for autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 6380–6387 (2020).

34. Van Den Oord, A., Vinyals, O. *et al.* Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 6306–6315 (2017).

35. Razavi, A., van den Oord, A. & Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*, 14866–14876 (2019).

36. Cheng, Y. & Church, G. M. Biclustering of expression data. In *Ismb*, vol. 8, 93–103 (2000).

37. Yosinski, J., Clune, J., Nguyen, A., Fuchs, T. & Lipson, H. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579* (2015).

38. Montavon, G., Samek, W. & Müller, K.-R. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* **73**, 1–15 (2018).

39. Abraham, K. J. *et al.* Nucleolar rna polymerase ii drives ribosome biogenesis. *Nature* **585**, 298–302 (2020).

40. Halevy, A., Norvig, P. & Pereira, F. The unreasonable effectiveness of data. *IEEE Intelligent Systems* **24**, 8–12 (2009).

41. Leonetti, M. D., Sekine, S., Kamiyama, D., Weissman, J. S. & Huang, B. A scalable strategy for high-throughput gfp tagging of endogenous human proteins. *Proc Natl Acad Sci U S A* **113**, E3501–8 (2016).

42. Li, C. H. & Lee, C. Minimum cross entropy thresholding. *Pattern recognition* **26**, 617–625 (1993).

43. Li, C. & Tam, P. K.-S. An iterative algorithm for minimum cross entropy thresholding. *Pattern recognition letters* **19**, 771–776 (1998).

44. Tan, M. & Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 6105–6114 (2019).

45. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778 (2016).

46. McInnes, L., Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

47. Rokach, L. & Maimon, O. Clustering methods. In *Data mining and knowledge discovery handbook*, 321–352 (Springer, 2005).

48. Abadi, M. *et al.* TensorFlow: Large-scale machine learning on heterogeneous systems (2015). URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Acknowledgements We would like to thank our colleagues at the CZ Biohub: Sandra Schmid, Mirella Bucci, Ahmet Can Solak, Bing Yang, Merlin Lange, Shruthi Vijaykumar, Luke Hyman, Marco Hein, for insightful discussions, feedback and for reviewing the manuscript. Thanks to Kibeom Kim for assistance in data analysis. Thanks to our colleague Michael Wu and James Zou from Stanford University for advice. We would like to thank Ashely Lakoduk for reviewing the manuscript. Finally, thanks to Japan Society for the Promotion of Science and its overseas research fellowships and the *Chan Zuckerberg Biohub* and its donors for funding this work.

Competing interests The authors declare that they have no competing financial interests.

Code and data availability Source code for the models used in this work will be available soon at: `http://github.com/royerlab/cytoself`

Correspondence Correspondence and requests for materials should be addressed to Hirofumi Kobayashi, Manuel Leonetti and Loic A. Royer ({hirofumi.kobayashi,manuel.leonetti,loic.royer}@czbiohub.org

Suppl. Files

1. proteins_for_cluster_score.csv, A list of proteins as a ground truth to compute clustering scores.
2. proteins_uniloc.csv, A list of proteins that has only one localization pattern.
3. proteins_subunits.csv, List of protein subunits for protein complexes mentioned.

Suppl. Figures

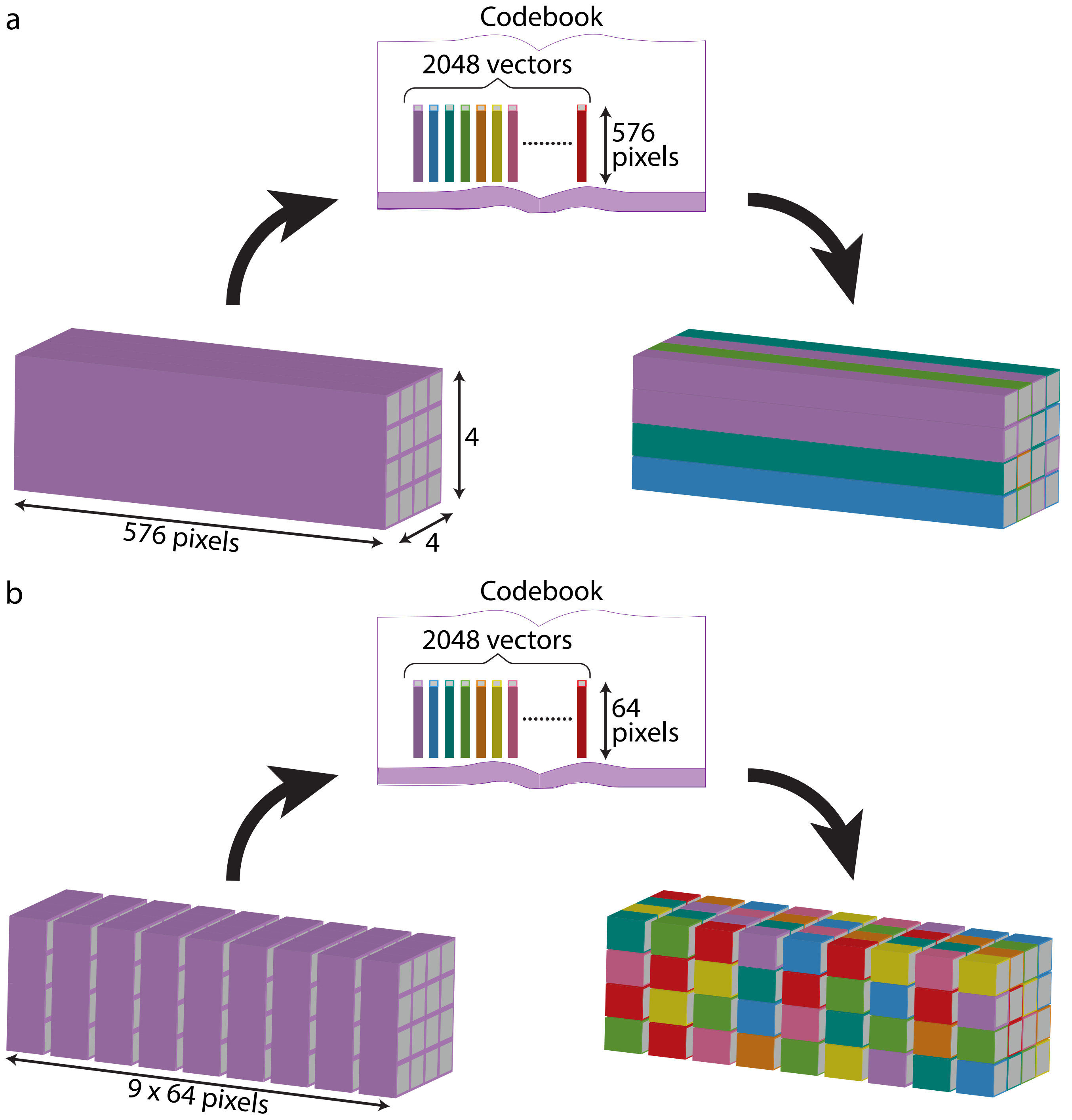

Figure 7: (Supplementary.) A schematic of split quantization. **(a)**, Without split quantization, there are only $4 \times 4 = 16$ quantized vectors in the global representation. **(b)**, With split quantization, there are $4 \times 4 \times 9 = 144$ quantized vectors in the global representation, resulting in more opportunities for codes in the codebook to be used.

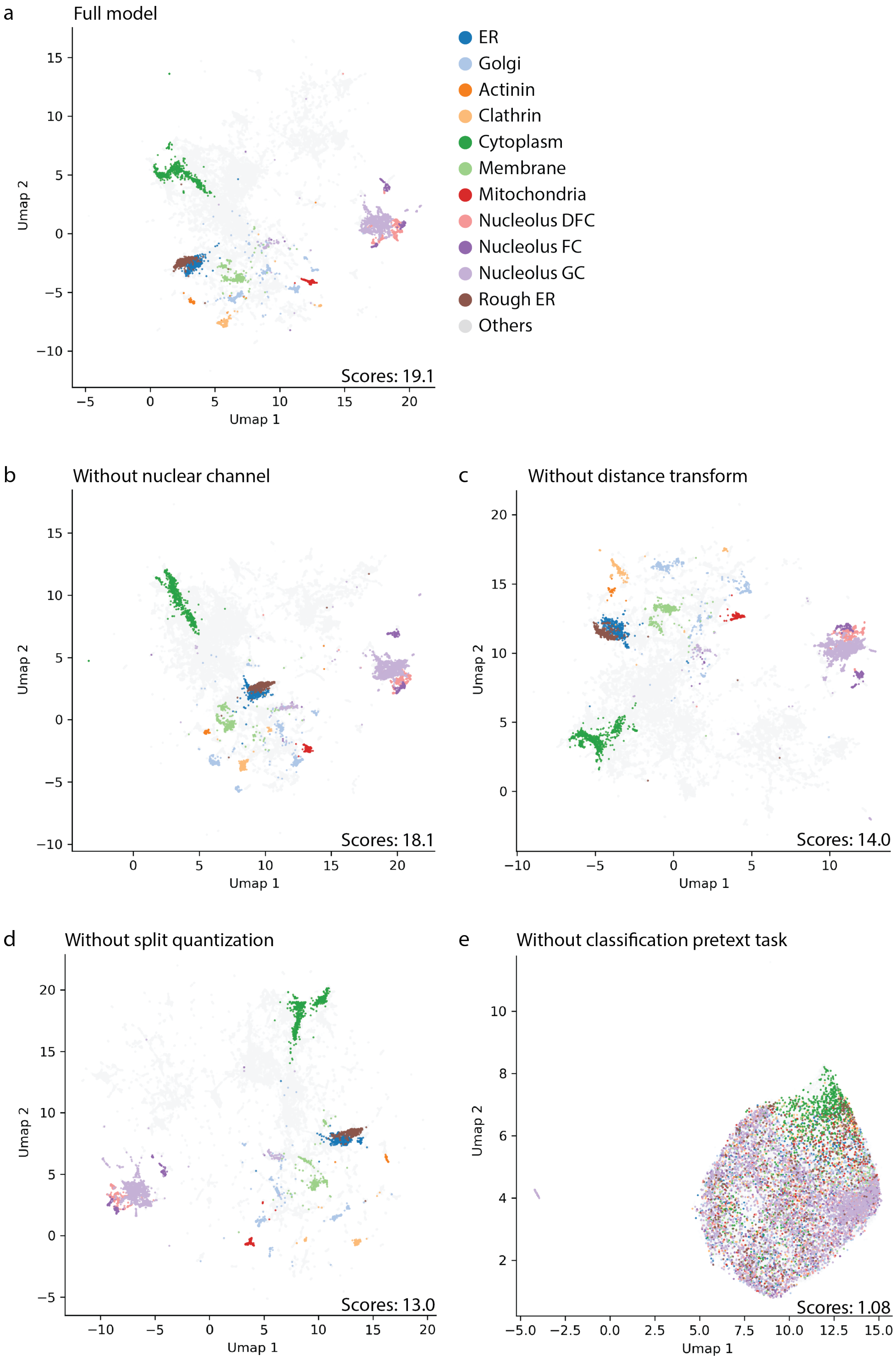

Figure 8: (Supplementary.) UMAP representations of global representation in variants of deep learning models. **(a)** Full model, **(b)**, without nuclear channel, **(c)**, without distance transform, **(d)**, without split quantization, **(e)**, without classification pretext task. Only the UMAP giving the highest clustering scores in each model variant is shown.

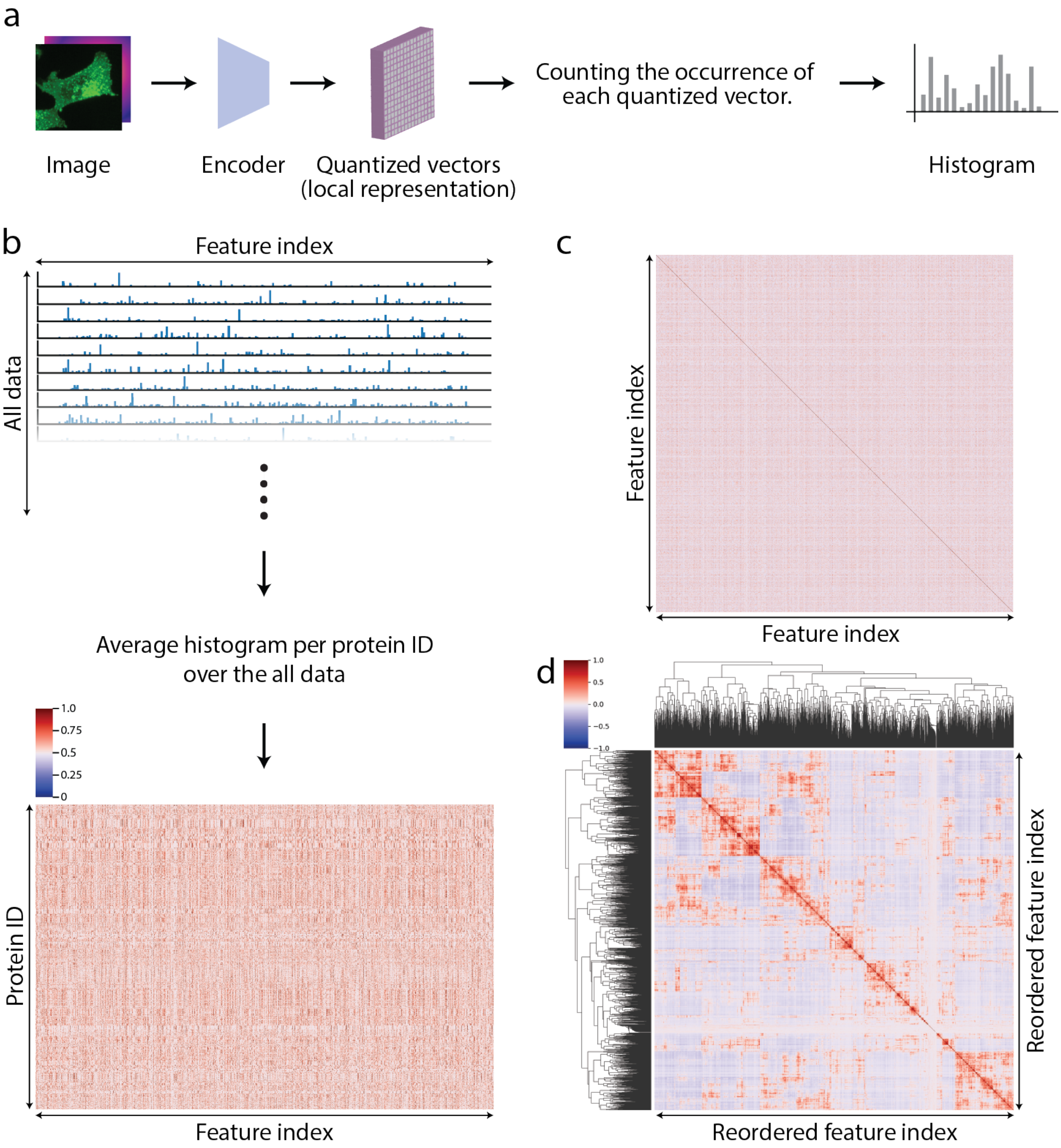

Figure 9: (Supplementary.) Process of constructing feature spectra. **(a)** First, the quantized vectors in the local representation were extracted and converted to a histogram by counting the occurrence of each quantized vector. **(b)** Next, taking the average of the histograms per protein ID over the all data and create a 2D histogram. **(c)** Pearson's correlation between any two representation indices were calculated and plotted as a 2D matrix. **(d)** Finally, hierarchical clustering was performed on the correlation map so that similar features are clustered together, revealing the structure inside the local representation. The whole process corresponds to the Spectrum Conversion in Fig. 1a.

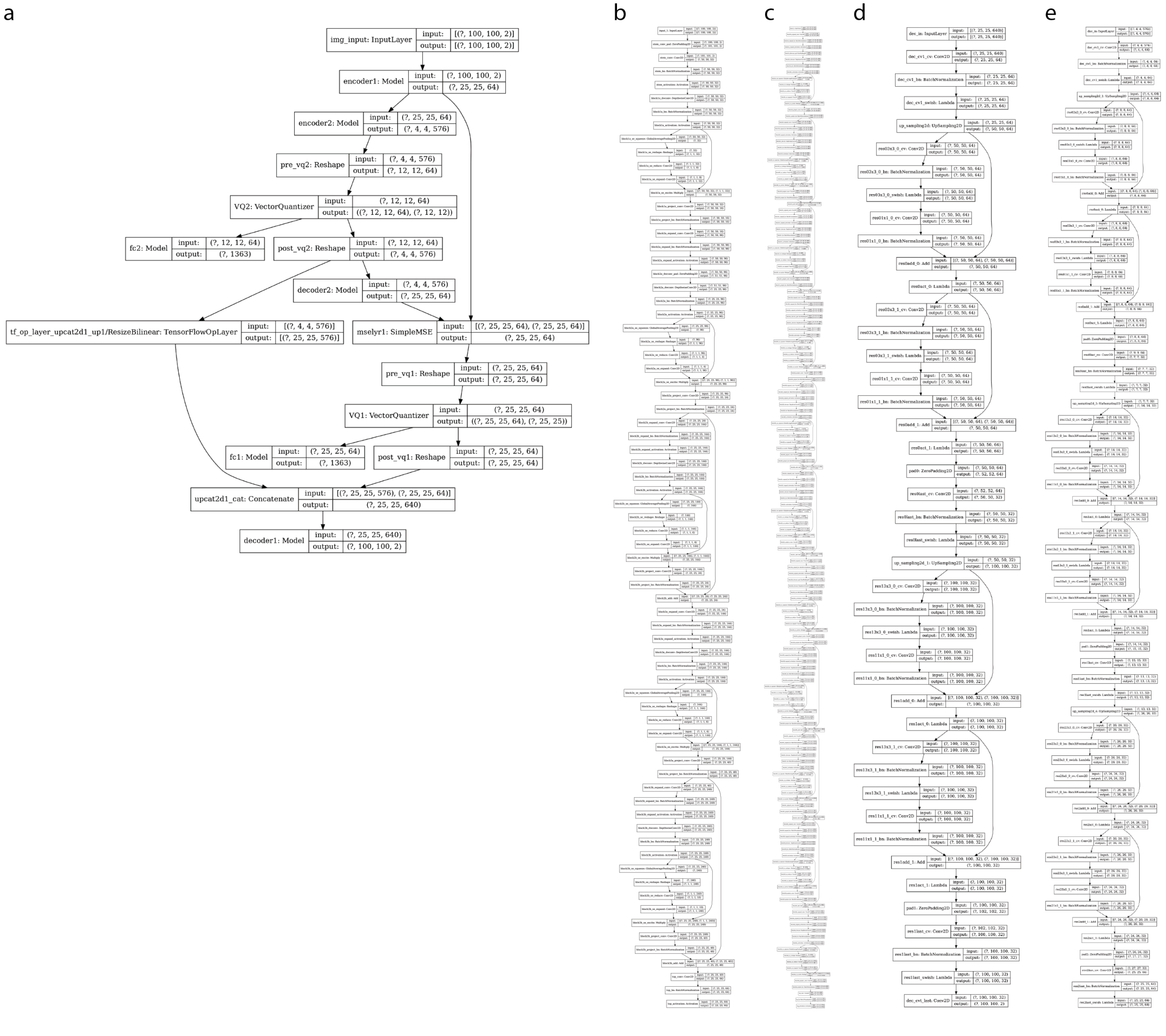
a
b
c
d
e
img_input: InputLayer input: [(?, 100, 100, 2)] output: [(?, 100, 100, 2)]
encoder1: Model input: (?, 100, 100, 2) output: (?, 25, 25, 64)
encoder2: Model input: (?, 25, 25, 64) output: (?, 4, 4, 576)
pre_vq2: Reshape input: (?, 4, 4, 576) output: (?, 12, 12, 64)
VQ2: VectorQuantizer input: (?, 12, 12, 64) output: ((?, 12, 12, 64), (?, 12, 12))
fc2: Model input: (?, 12, 12, 64) output: (?, 1363)
post_vq2: Reshape input: (?, 12, 12, 64) output: (?, 4, 4, 576)
decoder2: Model input: (?, 4, 4, 576) output: (?, 25, 25, 64)
tf_op_layer_upcat2d1_up1/ResizeBilinear: TensorFlowOpLayer input: [(?, 4, 4, 576)] output: [(?, 25, 25, 576)]
mselyr1: SimpleMSE input: [(?, 25, 25, 64), (?, 25, 25, 64)] output: (?, 25, 25, 64)
pre_vq1: Reshape input: (?, 25, 25, 64) output: (?, 25, 25, 64)
VQ1: VectorQuantizer input: (?, 25, 25, 64) output: ((?, 25, 25, 64), (?, 25, 25))
fc1: Model input: (?, 25, 25, 64) output: (?, 1363)
post_vq1: Reshape input: (?, 25, 25, 64) output: (?, 25, 25, 64)
upcat2d1_cat: Concatenate input: [(?, 25, 25, 576), (?, 25, 25, 64)] output: (?, 25, 25, 640)
decoder1: Model input: (?, 25, 25, 640) output: (?, 100, 100, 2)

Figure 10: (Supplementary.) Detailed structure of SVQ-VAE model. **(a)** the whole model structure, **(b)** the structure of encoder1, **(c)** the structure of encoder2, **(d)** the structure of decoder1, **(e)** the structure of decoder2.

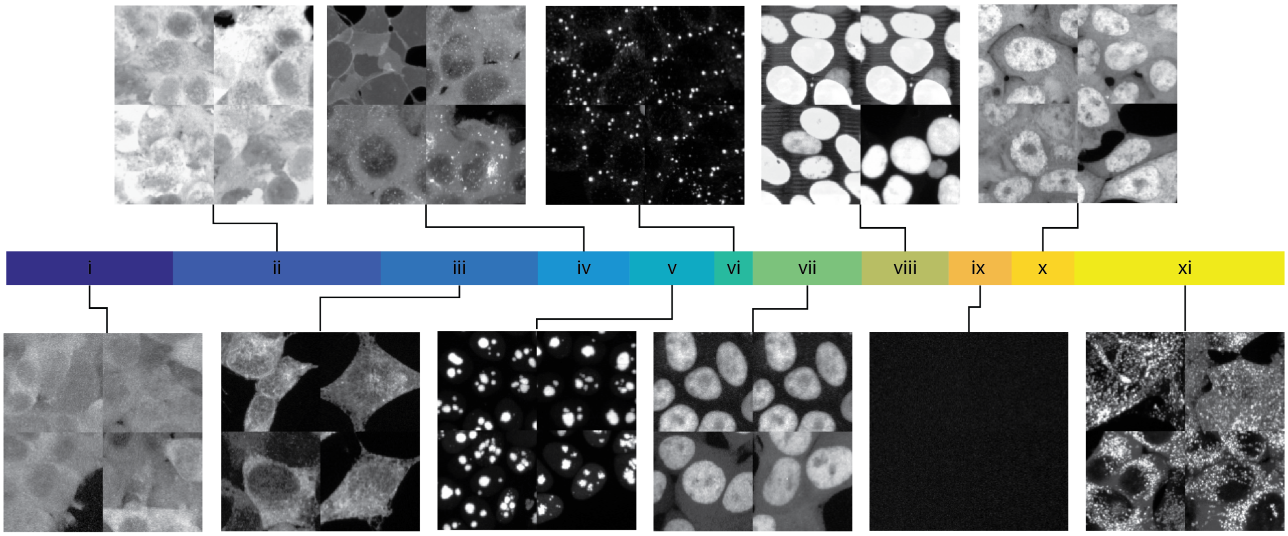

Figure 11: (Supplementary.) UMAP representations of global representation for differently ablated model variants. **(a)** Full model, **(b)** without nuclear channel, **(c)** without distance transform, **(d)** without split quantization, **(e)** without classification pretext task. Only the UMAP giving the highest clustering scores in each model variant is shown.