PAPER • **OPEN ACCESS**

A biology-informed similarity metric for simulated patches of human cell membrane

To cite this article: Harsh Bhatia *et al* 2022 *Mach. Learn.: Sci. Technol.* **3** 035010

IOP Publishing Mach. Learn.: Sci. Technol. 3 (2022) 035010 https://doi.org/10.1088/2632-2153/ac8523

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
23 January 2022

REVISED
14 June 2022

ACCEPTED FOR PUBLICATION
28 July 2022

PUBLISHED
19 August 2022

PAPER

A biology-informed similarity metric for simulated patches of human cell membrane

Harsh Bhatia[1,*], Jayaraman J Thiagarajan[1], Rushil Anirudh[1], T S Jayram[1], Tomas Oppelstrup[2], Helgi I Ingólfsson[2], Felice C Lightstone[2] and Peer-Timo Bremer[1]

[1] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA, United States of America
[2] Physical and Life Sciences Directorate, Lawrence Livermore National Laboratory, Livermore, CA, United States of America
[*] Author to whom any correspondence should be addressed.

E-mail: hbhatia@llnl.gov

Keywords: biology-informed similarity metric, cell membrane, cancer research, deep learning, metric learning, inductive biases

Abstract

Complex scientific inquiries rely increasingly upon large and autonomous multiscale simulation campaigns, which fundamentally require *similarity metrics* to quantify 'sufficient' changes among data and/or configurations. However, subject matter experts are often unable to articulate similarity precisely or in terms of well-formulated definitions, especially when new hypotheses are to be explored, making it challenging to design a meaningful metric. Furthermore, the key to practical usefulness of such metrics to enable autonomous simulations lies in *in situ* inference, which requires generalization to possibly substantial distributional shifts in unseen, future data. Here, we address these challenges in a cancer biology application and develop a meaningful similarity metric for *'patches'*—regions of simulated human cell membrane that express interactions between certain proteins of interest and relevant lipids. In the absence of well-defined conditions for similarity, we leverage several biology-informed notions about data and the underlying simulations to impose inductive biases on our metric learning framework, resulting in a suitable similarity metric that also generalizes well to significant distributional shifts encountered during the deployment. We combine these intuitions to organize the learned embedding space in a multiscale manner, which makes the metric robust to incomplete and even contradictory intuitions. Our approach delivers a metric that not only performs well on the conditions used for its development and other relevant criteria, but also learns key spatiotemporal relationships without ever being exposed to any such information during training.

1. Introduction

Many scientific phenomena involve wide ranges of spatial and temporal scales, but computational models usually cannot cover all relevant scales with sufficient fidelity. This limitation has given rise to multiscale simulations [2, 4, 11, 14, 19, 25, 47], where coarse and hence inexpensive approximations are used to explore large and long scales, whereas more-detailed but significantly more-expensive models are used to provide details for smaller regions in space and time. A key ingredient of creating such multiscale simulations is a determination of when and where the expensive models should be used.

Existing approaches [10, 24, 35, 40] address this challenge by sampling known and predefined configuration spaces. Recently, machine learning (ML) has also been employed for this task by encoding known behavior into a reduced, latent space [4]. However, requiring precise knowledge about the specific space (or behavior) to sample from is an important limitation, especially when the goal is to explore unknown behavior. Here, we leverage ideas from *ML-based metric learning* [21, 32, 44, 52] to address this challenge and facilitate multiscale simulations in the context of cancer biology.

The overarching scientific goal is to explore the interactions of RAS proteins and RAS-RAF protein complexes with the lipid bilayer that forms the human cell membrane [17–19]. RAS-RAF activation is a crucial part of the signaling chain that controls cell growth, and up to a third of all human cancers are driven

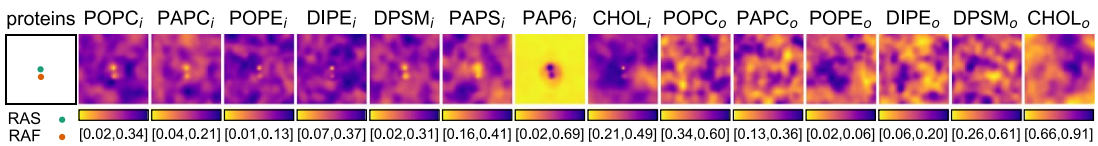

Figure 1. A 'patch' comprises zero or more proteins of two types and concentration distributions of 14 types of lipids. Facilitating target multiscale simulations requires determining similarity among patches, even though there exists no well-formulated or widely-accepted notion of similarity.

by mutations of RAS proteins that corrupt this chain [38, 43]. Consequently, understanding the signaling process in detail is of significant interest [19, 22, 45, 49]. However, even using some of the largest supercomputers, only a few potential signaling events of interest can be explored with sufficiently detailed molecular dynamics (MD) models. To maximize the opportunity for discovery, computational scientists seek to employ 'coarse-scale' models to continuously create wide-ranging membrane configurations and, from these, select a small but diverse subset to explore in detail using 'fine-scale' simulations.

Specifically, during the target multiscale simulation campaign (workflow and hardware-software functionality described by Bhatia *et al* [5]), a coarse-scale, continuum model simulation evolves density distributions of lipids and single-particle representations of RAS and RAF proteins for a large (1 μm $\times$ 1 μm) portion of a membrane. The signaling events of interest are thought to depend on the spatial arrangement of lipids in the neighborhood of RAS, the conformation of RAS, its orientation relative to the membrane, and a range of other factors. Therefore, the focus is on utilizing the fine-scale, MD simulations to explore *patches*, which represent small (30 nm $\times$ 30 nm) regions of the membrane around RAS proteins. Whereas the ongoing continuum simulation creates a continuous stream of patches, all of which are candidates for MD simulations, the high computational cost of MD allows only a small fraction (e.g. about 0.05%) of the patches to be explored. For exploration of wide-ranging RAS-RAF-membrane configurations while maintaining judicious use of computational resources, our collaborators aim to instantiate MD simulations for a diverse set of patches covering as much of the phase space explored by the continuum model as possible. Mathematically, this requirement implies capturing similarity among patches, which unfortunately is not well defined but only understood qualitatively and intuitively.

The concept of *similarity* is fundamentally important in almost all scientific fields and data-driven analyses, such as medicine [34, 50], security [27, 33], social media mining [29, 30], speech recognition [3, 26], information retrieval [16, 31], recommender systems [28, 51], and computer vision [37, 48, 53]. Although *similarity metrics* are used widely, existing approaches employ simple and well-formulated measures, such as well-defined norms (e.g. L_p-norms, Mahalanobis distance, cosine distance, and correlation coefficients) or supervisory labels. However, in many scientific applications, including the one of interest here, there exist no explicit labels, and the standard norms do not match the scientific understanding of domain scientists. Unsupervised, autoencoder-based techniques [4, 6, 20] have also been utilized to identify similarities. Nevertheless, reconstruction-based training in such approaches also implies prescient notions of properties that need to be preserved, potentially disregarding other intuitions. As such, existing ideas present little utility for our target application.

Instead, biologists—both experimental and computational—usually express various intuitions and hypotheses regarding what similar and dissimilar configurations *might* look like. In this case, experts hypothesize [19] that the spatial distribution of lipids correlates with types of protein constellations (numbers and types of RAS and RAF proteins). Therefore, of particular interest here is to understand how the different types of lipids rearrange themselves in response to the proteins (see figure 1 for an illustration). Nevertheless, neither identifying this response nor comparing such spatial patterns is straightforward, particularly through well-defined mathematical formulations, but it is rather understood only intuitively and qualitatively. The biology community often compares patches and other such configurations using simple and aggregated criteria, e.g. mean lipid concentrations [17]. However, such metrics are not descriptive enough to be useful here, e.g. because same mean concentrations can differ in their spatial distributions. On the other hand, directly comparing spatial patterns (e.g. pixel-wise) is also not suitable because it does not preserve key invariances of interest (patches are invariant to axis-aligned rotations and reflections, and so the similarity metric must be invariant too) and is therefore not biologically relevant. One might consider patches with same protein constellations (i.e. same numbers of RAS and RAF) as similar. Nevertheless, the reverse may not hold true (i.e. different protein constellations may exhibit similar spatial patterns). Broadly, such biology-informed intuitions lead to loosely defined conditions that might be under-constrained (too few conditions) or even inconsistent.

Finally, an important practical challenge is that this metric has to be developed without even knowing in advance exactly what types of patches will be generated during the target multiscale simulation, and there is no opportunity for retraining the ML model. Specifically, the multiscale simulation campaign evolves a coupled system of continuum and MD simulations, consuming almost 3.6 million GPU hours [5]—a task that cannot be repeated. Prior to the campaign, only an uncoupled continuum model may be run and only for a short period. As a result, the data available to train the metric (the '*pre-campaign data*') is expected to differ considerably from the data generated for inference during the campaign (the '*campaign data*'), although the extent and the exact type of differences are mostly speculation, since such a coupled model has never been simulated before. Therefore, generalization of the metric from training data to other relevant simulations is key to practical applicability.

Contributions

We present a method to combine several relevant intuitions and known constraints from experts to construct a *suitable and generalizable*, *biology-informed similarity metric* for patches generated from continuum simulations of RAS-RAF-membrane interactions. Whereas a broad limitation in the community is that notions of patch similarity exist only as expert intuitions rather than well-formulated conditions, our framework transforms this limitation into a strength by leveraging these biology-informed intuitions to impose *inductive biases* (a set of assumptions that allow generalizing a finite set of observations into a general model) on our learning algorithm.

We demonstrate that our metric is *scientifically relevant* and captures the underlying biology, e.g. preserves additional scientific constraints not part of the training. Most notably, our metric exhibits a strong correlation between similarity of patches and the timescales needed for one to evolve into the other. This correlation emerged naturally through metric learning and matches the fundamental assumptions of statistical mechanics approaches, despite the training data containing no notion of time evolution.

Impact

Our metric was deployed for *in situ* inference as an enabling technology for a massive multiscale simulation of RAS-RAF-membrane biology to create the first-of-its-kind scientific campaign [5] to study this phenomena. Beyond the application scope of this paper, our technique has been successfully adapted to support simulations with different constraints. Our approach is generally adaptable to a wide range of scientific problems (e.g. other protein systems) leading to better design of experiments, more-stable predictive models, and better clustering, with the potential for significant impact.

2. A biology-informed similarity metric for patches

In this work, we pose the goal of identifying similarity among patches as a *metric learning* problem and express three relevant scientific intuitions mathematically to learn a suitable metric. We then employ a neural network to learn this metric and subsequently use it to define similarity between patches.

2.1. Fundamentals and background

2.1.1. Formal description of a patch

Shown in figure 1, a patch comprises density fields of 14 types of lipids along with the counts of the RAS and RAF proteins (Appendix section 'Description of the simulation model' introduces the underlying biological system). Specifically, we are given pairs, $(\mathbf{x}, \mathbf{n})$, where $\mathbf{x} = \mathbf{x}(i, j, c)$ is a multichannel image with the channels $0 \leqslant c < 14$ representing the concentrations of different types of lipids on a uniform spatial grid with indices $0 \leqslant i, j < 37$, and $\mathbf{n} = (n_s, n_f)$ is a tuple of the counts of RAS and RAF proteins in the patch, respectively (i.e. $n_s, n_f \geqslant 0$). We further note that RAF localizes at the membrane only in association with RAS; therefore, for each RAF, there is a RAS in the patch (i.e. $n_s \geqslant n_f$). By construction, each patch with at least one protein is centered around a RAS protein, providing a consistent reference frame across patches.

Without loss of generality and in the context of the current application, patches are broadly categorized into four *types* based on the proteins contained therein, $y(\mathbf{n}) \in [0, 1, 2, 3]$. Here, $y((0,0)) = 0$ represents a patch with no protein, $y((1,0)) = 1$ is a patch with a single RAS, $y((1,1)) = 2$ is a patch with a single RAS-RAF complex, and $y((n_s, n_f)) = 3$ represents a patch with any other combination of proteins (i.e. when $n_s > 1, n_f > 1$). As described ahead, this categorization of patches into four types will be leveraged for (partially) describing the similarity among patches.

Given $\mathbf{x}$, $\mathbf{n}$, and y, we formally denote a patch as $\mathbf{p} = (\mathbf{x}, y(\mathbf{n}))$, or simply $\mathbf{p} = (\mathbf{x}, y)$.

2.1.2. Unsuitability of existing autoencoder-based approaches

Previously, Bhatia *et al* [4] presented the only known approach that addresses the problem of identifying similarity for patches. This work employed a variational autoencoder to create a latent space and defined similarity in this latent space. Although our goals are similar at the conceptual level, the previous method is unsuitable here due to different application requirements. Specifically, in previous work, patches represent a simpler system, including only RAS proteins; thus, there is no need to identify and distinguish the responses to different proteins. As such, working only with the images is sufficient (i.e. $\mathbf{p} = \mathbf{x}$ in this work). Furthermore, patches in the previous work were represented at a substantially lower spatial resolution (5×5 as compared to 37×37 in our case). At this coarse level of detail, spatial patterns do not show enough variability for a meaningful assessment of spatial distributions—a limitation of the data in principle. Nevertheless, the previous work leverages the low resolution by capturing all-to-all pixel correlations using fully-connected (dense) layers in a neural network. Such all-to-all correlations do not distinguish between the spatial locations of pixels, leading naturally to invariance to rigid transformations of the patch. On the other hand, whereas the advantage of higher spatial resolution in patches is that it allows postulating new hypotheses about the spatial patterns and offers opportunity to study spatial correlations using convolution layers, these invariances must be accounted for explicitly.

These differences in data along with our requirement to obtain invariance to certain rigid transformations of interest necessitate a departure from autoencoder-type networks because reconstruction losses used in autoencoders rely on pixel-wise comparison that directly contradicts the invariance needed. In fact, the two parts of the autoencoder (encoder and decoder) work against each other making the optimization ill-posed. Whereas a well-performing encoder could map transformations of patches infinitesimally close to each other, the decode would fail to reconstruct original patches by distinguishing such cases. On the other hand, if the decoder learns to distinguish between transformations, it implies that the encoder failed to effectively encode the pixel-wise correlations in the patches. This conflict within the autoencoder is a fundemental limitation of such ideas, making the problem unnecessarily difficult.

Instead, our metric learning approach satisfies all necessary conditions significantly more effectively. In this work, we show that our approach delivers a $2130\times$ reduction in data (from $37 \times 37 \times 14 = 19\,166$ pixels to 9 dimensions), as compared to the previous autoencoder-based method [4], which reduces the data by only $23\times$ (from $5 \times 5 \times 14 = 350$ pixels to 15 dimensions).

2.1.3. Deep-learning based metric learning

Directly relevant to this work, deep-learning based metric learning [9, 21, 32, 44, 52] has emerged as a powerful approach. The fundamental idea is to learn a function, $f: \mathbf{X} \rightarrow \mathbf{Z}$, that maps the input space, $\mathbf{X}$, onto an embedding space, $\mathbf{Z}$, such that some known formulation of distance, $\mathrm{d}\mathbf{z}$, in this embedding space can approximate the 'semantic distance' (i.e. the similarity) in the input space. Given a suitable $\mathrm{d}\mathbf{z}$ of choice, the goal is then to learn the mapping f—a task achieved by training a neural network to optimize over a set of parameters θ and for a loss function $\mathcal{L}$, resulting in $f = f_\theta^{\mathcal{L}}$.

In this context, *triplet losses* [15, 42] have shown remarkable success, particularly in face recognition and object detection problems [12, 41]. Fundamentally, the triplet loss relies on pairs of similar ($\mathbf{x}^{\mathrm{a}}$ and $\mathbf{x}^{\mathrm{p}}$) and dissimilar ($\mathbf{x}^{\mathrm{a}}$ and $\mathbf{x}^{\mathrm{n}}$) examples of the training data; the network is trained to minimize the distance ($\mathrm{d}\mathbf{z}$) between learned representations ($\mathbf{z} \in \mathbf{Z}$) of the examples from the same class and place a margin (α) between those of different classes or categories. Formally, a triplet loss is given as

$$\mathcal{L}_\alpha(\mathbf{x}^{\mathrm{a}}, \mathbf{x}^{\mathrm{p}}, \mathbf{x}^{\mathrm{n}}) = \max(0,\ \alpha + \mathrm{d}\mathbf{z}(\mathbf{x}^{\mathrm{a}}, \mathbf{x}^{\mathrm{p}}) - \mathrm{d}\mathbf{z}(\mathbf{x}^{\mathrm{a}}, \mathbf{x}^{\mathrm{n}})). \tag{1}$$

There also exist many other popular formulations of loss functions that employ similar strategies, e.g. contrastive [13] and quadruple [7] losses; all such approaches are supervised, usually in the form of class labels. In this work, we utilize triplet losses to supervise or self-supervise the learning in part.

2.2. Capturing similarity through our metric learning framework

Given a patch $\mathbf{p} = (\mathbf{x}, y)$ as defined in section 2.1.1, we build $f_\theta^{\mathcal{L}}: \mathbf{p} \rightarrow \mathbf{z}$, where $\mathbf{z} \in \mathbf{Z} \subset \mathbb{R}^d$ represents the patch in the learned embedding space. We use the Euclidean distance metric in $\mathbb{R}^d$ to compute distances in the learned space, i.e. $\mathrm{d}\mathbf{z}(\mathbf{p}_i, \mathbf{p}_j) = \|\mathbf{z}(\mathbf{p}_i) - \mathbf{z}(\mathbf{p}_j)\|_2$. In this section, we discuss three scientific intuitions and describe how we formulate them mathematically to create three objective functions, and then combine them to create a $f_\theta^{\mathcal{L}}$ that delivers a meaningful notion of similarity captured through $\mathbf{Z}$.

2.2.1. Supervised classification of protein constellations

One key intuition on similarity that biologists have is that different protein constellations (i.e. different patch types, y) should be considered dissimilar. More generally, this intuition is an example of task-specific

information that is commonly available in many scenarios and can be used directly to supervise the learning. Since the application focuses on exploring lipids' response to proteins, a metric that learns on lipid distributions and uses protein constellations as a dependent variable is suitable to provide new insights. Therefore, given a patch $\mathbf{p} = (\mathbf{x}, y)$, we treat $\mathbf{x}$ as *input* and y as a *label*, and model this intuition as a clustering problem with a *classification loss* formulated as a triplet loss,

$$\mathcal{L}^{\text{lab}} = \mathcal{L}_{\alpha^{\text{lab}}}(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k), \tag{2}$$

where $\mathbf{p}_i = (\mathbf{x}_i, y_i)$ and $\mathbf{p}_j = (\mathbf{x}_j, y_j)$ are patches with the same type (i.e. $y_i = y_j$), $\mathbf{p}_k = (\mathbf{x}_k, y_k)$ is a patch with a different type from the first two (i.e. $y_k \neq y_i$), and α^{lab} the desired margin.

Although such a clustering approach is standard, a key challenge in our case is that this label-based loss *may* contradict other intuitions. For example, certain patches may have protein constellations (y) that cannot be distinguished based on the corresponding lipid distributions ($\mathbf{x}$) only. In fact, it is of special interest to the application to be able to identify which patches show a poor correlation between protein constellations and lipid distributions. Later in section 2.2.4, we will discuss how we synergize possible contradictions and achieve a balance between such intuitions.

2.2.2. Self-supervised invariance to axis-aligned rotations and reflections

As is common in many scientific simulations, the chosen coordinate system is arbitrary, i.e. the two coordinate axes could be rotated or inverted without changing the biology simulated. A suitable metric that learns the underlying biology must, therefore, be agnostic to the specific choice of the coordinate system. Stated differently, our learned metric must be invariant to such transformations of patches.

Without loss of generality, let $\rho(\mathbf{p})$ denote such transformations of interest. Given ρ, we require that $\mathrm{d}\mathbf{z}(\mathbf{p}, \rho(\mathbf{p})) = 0$ for all $\mathbf{p}$, and, by implication, $\mathrm{d}\mathbf{z}(\mathbf{p}_1, \mathbf{p}_2) = \mathrm{d}\mathbf{z}(\mathbf{p}_1, \rho(\mathbf{p}_2))$ for all $\mathbf{p}_1$ and $\mathbf{p}_2$. We model this invariance using a triplet loss to force all transformations of every patch, $\mathbf{p}_i$, closer to each other than any other pair of patches, $\mathbf{p}_i$ and $\mathbf{p}_j$ with $i \neq j$, by some margin α^{inv}. Formally, we define an *invariance loss* as

$$\mathcal{L}^{\text{inv}} = \mathcal{L}_{\alpha^{\text{inv}}}(\mathbf{p}_i, \rho(\mathbf{p}_i), \mathbf{p}_j). \tag{3}$$

Here, we are interested in five types of rigid transformations. These are reflections along the horizontal and vertical axes, and the three axis-aligned (90°, 180°, and 270°) rotations. Whereas a true biological system would be invariant to continuous rotations, the simulation framework that generates this data imposes restrictions on rotations. Specifically, the simulation uses a finite-element regular grid and generates sampled patches using a higher-order interpolant. Any continuous rotation of a patch, therefore, requires resampling the lipid densities onto a rotated grid—a task to be performed within the same simulation code, currently not available to us, and is, thus, out of scope of this work. Furthermore, patches are centered around RAS proteins (when there exists one; see section 2.1.1); therefore, translations of patches are not relevant either. In summary, the application governs what transformations are relevant and possible, and our framework itself is agnostic to the types of transformations as long as they can be generated practically.

Given a dataset, whereas in principle, two different and unrelated patches could still be exact transformations of each others, in practice, the probability of finding such a pair is virtually zero. Therefore, we assume all given patches are different and perform data augmentation in the form of online triplet mining by transforming each patch in a given training batch to its five relevant transformations. Such augmentations lead to a self-supervised approach for training the metric to identify invariance.

2.2.3. Feature proportionality for lipid spatial patterns

The spatial arrangement of lipids is key to describing similarity among patches (see figures 1 and 5). To this end, we define a feature that is inspired by radial distribution functions, which are used extensively to study such biological simulations and whose intuition is well favored by subject matter experts. This feature, a *radial profile*, is attractive because it captures similarity in lipid arrangements, helps alleviate contradictions with protein constellations, and synergizes with the required invariance.

Given the multichannel image representing lipid densities within a patch, $\mathbf{x}(i, j, c)$ and a reference pixel thereof, (i_*, j_*), a radial profile measures the maximum lipid density along a given radius r—we consider only integer values for r. Formally, we define *radial profile per channel* as $\gamma_{(i_*, j_*)}(\mathbf{x}, c, r) = \max_C(\mathbf{x}(i, j, c))$, where C is the constraint $r = \lfloor r_* + 1/2 \rfloor$ and $r_* = \sqrt{(i - i_*)^2 + (j - j_*)^2}$. We use max aggregation as it is sensitive to variations across r and, thus, is suitable to distinguish patches. Since a patch has a protein at its center (when $n_s > 0$), we compute the radial profile from the center pixel spanning the incircle of the image, i.e. $\gamma_{(18,18)}(\mathbf{x}, c, r)$ for $r < 19$ (recall that $\mathbf{x}$ is sampled on a 37×37 grid). To account for the lipids outside the incircle, we compute radial profiles from each corner, also for $r < 19$ but these corner profiles encompass only the corresponding quadrant, i.e. $\gamma_{(0,0)}$, $\gamma_{(0,36)}$, $\gamma_{(36,0)}$, and $\gamma_{(36,36)}$. To preserve invariance to transformations

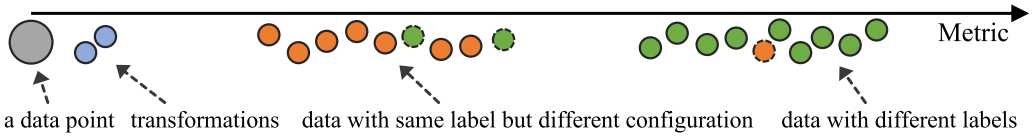

Figure 2. Our framework imposes inductive biases using biology-informed intuitions to organize neighborhoods of data points into a multiscale hierarchy that represents the different conditions. Given a data point, its relevant transformations are placed closest to it, whereas data that can be easily distinguished based on labels is placed the farthest. The intermediate space is filled with data with the same label but proportional to similarity in a chosen feature space. The 'out-of-order' points (green in the orange cloud and orange in the green cloud) represent contradictions (within the intuitions) that are synergized by our metric learning framework through explicit weighting of the three loss functions.

of interest, the corner profiles are averaged. Both the center profile and the mean of corner profiles are $[19 \times 1]$ vectors; they are concatenated to give a $[38 \times 1]$ vector for each channel. See Appendix section 'Computation of radial profiles' for further details and illustrations on radial profiles.

In this work, we used only eight lipid channels that biologists deemed to be more important than others; specifically, the eight lipids on the inner leaflet, which exhibit a stronger response to the proteins, see figures 1 and 5 (also described in Appendix section 'Description of the simulation model'). Our final radial profile feature for a patch, $\Upsilon(\mathbf{p})$ is, therefore, a $[38 \times 8]$ vector. We next define a *feature proportionality loss* that forces the metric to keep similar Υ together in the learned embedding. Specifically, we use

$$\mathcal{L}^{\text{rad}}(\mathbf{p}_i, \mathbf{p}_j) = |\mathrm{d}\mathbf{z}(\mathbf{p}_i, \mathbf{p}_j) - \lambda^{\text{rad}}\, \mathrm{d}\Upsilon(\mathbf{p}_i, \mathbf{p}_j)|, \tag{4}$$

where, $\mathrm{d}\Upsilon(\mathbf{p}_i, \mathbf{p}_j) = \|\Upsilon(\mathbf{p}_i) - \Upsilon(\mathbf{p}_j)\|_2$ and λ^{rad} controls the span of a metric learned using this loss with respect to the valid ranges of $\mathrm{d}\Upsilon$.

2.2.4. Combining the three intuitions

Each of the three conditions discussed above may be straightforward to satisfy in isolation. For example, $\mathbf{z} = f(\mathbf{x}, y) = \sum_{i,j,c} \mathbf{x}(i,j,c)$ is a simple and linear map that is invariant to transformations of interest. Supervised learning also offers several straightforward ways to define similarity based on labels (protein constellations, in our case). Finally, one can also account for only the distances in the feature space of radial profiles with ease. However, potential contradictions between these conditions pose significant challenges. Consider the scenario where two patches with different protein constellations exhibit remarkably similar radial profiles, yet other pairs of patches within same protein constellation classes exhibit a larger variability in spatial patterns. Such scenarios can arise for a variety of application-related reasons, including but not limited to a less-than-perfect parameterization of the simulation model, inherent latency in lipids' response to a new protein, patches that may show transient behavior, etc.

Nevertheless, despite potential contradictions, these conditions offer valuable heuristics that we use to impose inductive biases on the similarity metric; these inductive biases are useful to obtain generalization across datasets and simulations. Given such contradictory conditions, we develop a single and consistent framework that absorbs such contradictions by aiming to organize the learned embedding space with respect to growing neighborhoods of data points, as illustrated in figure 2.

Given a reference patch, we require all its transformations to be 'really close' to it (as compared to non-transformations) and all patches with different labels to be 'much farther' (as compared to the patches with the same label). The set of patches that are not transformations and have the same label are distributed in between, based on the similarity in the radial profiles. We realize this multiscale organization using a small margin for invariance, α^{inv}, a larger margin for labels, α^{lab}, and a proportionality factor that maps the range of distances between radial profiles to the desired range in the latent space, i.e. $\lambda^{\text{rad}} \propto 1/(\max(\mathrm{d}\Upsilon) - \min(\mathrm{d}\Upsilon))$. To combine the three requirements, we use a weighted sum of the corresponding loss functions, i.e.

$$\mathcal{L} = w^{\text{inv}}\mathcal{L}^{\text{inv}} + w^{\text{rad}}\mathcal{L}^{\text{rad}} + w^{\text{lab}}\mathcal{L}^{\text{lab}}. \tag{5}$$

Our framework allows balancing the conflicts described above through controlling the hyperparameters and the weights of the three loss functions. For instance, relatively large values of α^{lab} and w^{lab} will aim to obtain separability with respect to labels at the cost of separability in the feature space. Therefore, the targets of discovery are identifying suitable hyperparameters that support good generalizability: the weights (w^{inv}, w^{rad}, and w^{lab}), the margins (α^{inv} and α^{lab}), the ranking proportionality factor (λ^{rad}), together with the dimension (d), of the resulting embedding space, $\mathbf{Z} \in \mathbb{R}^d$.

3. Results

We use our framework to develop a similarity metric for patches using the '*pre-campaign*' dataset. See Appendix section 'Deep learning model development' for details of the model development and formalism and architecture of the final model. Here, we demonstrate that the similarity metric developed using our framework described above is meaningful and generalizes well. In the absence of any ground truth, first we develop three quantitative evaluation criteria to assess the quality of a similarity metric and then show how these criteria can be used for selecting a model. Finally, we demonstrate that our metric does indeed capture the underlying biology by evaluating it on additional scientific criteria not part of the model development and/or training as well as its performance on unforeseen data during the multiscale simulation campaign.

3.1. Metric evaluation

We use three evaluation criteria to assess the quality of metrics learned using our method and to select a suitable model for the target multiscale simulation. These criteria correspond to the three conditions presented in section 2.2. We first describe these criteria using our final, selected model that delivers a meaningful 9D embedding space and then present our model selection procedure and other models. This evaluation was done on a set of 30 000 randomly selected patches from the validation data, providing almost 450 million pairs of points that we compute distances for.

3.1.1. Separability of protein constellations

Figure 3 visualizes a 2D t-SNE [46] of $\mathbf{z}$. As expected, the no-protein patches ($y = 0$) form an isolated cluster, since the lipids do not have a protein to respond to and are, therefore, easy to distinguish from the patches with proteins. The 1-RAS and 1-RAS-RAF patches ($y = 1$ and $y = 2$) are also relatively well separated due to inherent differences in the corresponding lipid responses. Nevertheless, although all other patches ($y = 3$) appear to have a mostly-well-defined cluster, we observe notable overlap with clusters for $y = 1$ and $y = 2$—these are the patches where the lipid configurations (images) do not exhibit distinctive responses to protein compositions (classes), likely because adding a new protein (e.g. a RAS) to a patch that already has another (e.g. a RAS-RAF) does not alter lipid compositions very drastically. Such patches are of substantial interest to the application, and as a result, it is important to not over-penalize the similarity metric model for such misclassifications, but rather strive for a balance with similarity in multichannel images.

To quantify the model's classification capability, we use the *area under the precision-recall curve* (AUC) in the embedding space, $\mathbf{Z}$ (see Appendix section 'Precision-recall study for classification of class labels'). AUC ranges between 0 and 1 with higher values representing larger separability between expected clusters. For each validation data point, we compute precision and recall for increasing numbers of neighbors, leading to the precision-recall curve. For the chosen model, the AUC is 0.61. Note from table 1 that even when considering only the classification objective (Ablation: $\mathcal{L}^{\text{lab}}$), the AUC achieved for this data is only 0.76, highlighting the challenges in the class separability for patches. Nevertheless, looking at the trend of precision, recall, and accuracy with respect to neighborhood sizes for the chosen model (see figure 11(right)), we note relatively good performance (accuracy >70%) for smaller neighborhoods, which are of substantially more interest for diversity sampling than long-range neighborhoods (see section 3.3 for how the metric is used).

3.1.2. Invariance and separability of transformations

We define an *overlap* metric to quantify a model's ability to capture invariance to transformations and separate them from any pairs of patches that are not transformations of each other. Overlap reports the proportion of points that cannot be distinguished any more than the transformations of some data points. Specifically, overlap counts pairs $(\mathbf{p}_j, \mathbf{p}_k)$, where $d\mathbf{z}(\mathbf{p}_j, \mathbf{p}_k) \leqslant \max(d\mathbf{z}(\mathbf{p}_i, \rho(\mathbf{p}_i)))$ for all five transformations of interest of all points, $\mathbf{p}_i$. Figure 3(middle) shows the distributions of these distances with an overlap of only 0.05% and, thus, demonstrates an excellent separability of transformations by the chosen model.

3.1.3. Preservation of proportionality with the similarity of lipid patterns

Finally, we quantify the model's capability to preserve the distances given by the feature space (radial profiles). Figure 3(right) shows the correlation between our metric and distances in the feature space for all 450 million pairs of points in the validation dataset. The plot highlights an excellent linear correlation with an expected proportionally factor 0.1 ($=\lambda^{\text{rad}}$ used for this model). This result demonstrates the model's capability to preserve the given distances (within the proportionality factor), even if at the cost of clustering quality (discussed above), thus, addressing the contradictions where needed. We further note that the model shows low error despite the large sample size (shown in gray) in the short-range distances—which are of

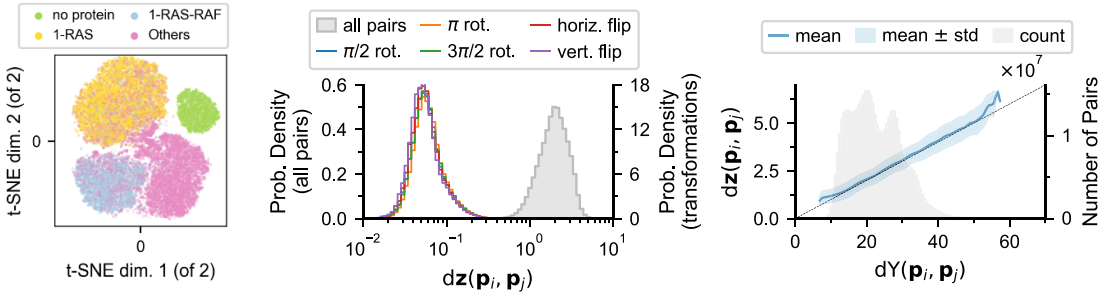

Figure 3. We evaluate our similarity metrics using three criteria: (left) high ability to separate different classes of data, using *AUC*. (middle) Low *overlap* between the distances among data points and those among transformations. (right) Low residual error (*MARE*) between the distances captured by the metric and those given by the feature space. See table 1 and figure 4 for quantitative comparison.

Table 1. Comparison with benchmarks and ablations indicate that our approach provides the best balance among the criteria and maintains a low dimensionality of our metric's embedding space. To compare against standard metrics consistently, this table uses MARE* (units of $\mathrm{d}\Upsilon$), which differs from MARE (units of $\mathrm{d}\mathbf{z}$) by a scaling factor ($\mathcal{L}^{\mathrm{rad}}$). The arrows in the table header indicate ideal values: lower values in dimensionality, overlap, and MARE* are better, whereas higher value is better for AUC.

	Metric	Metric dimensionality (↓)	Overlap % (↓)	MARE* (↓)	AUC (↑)
Standard:	ℓ_2 ($\mathbf{x}$; images)	19 166	99.99	13.54	0.34
	ℓ_2 (μ; mean conc.)	14	0	19.64	0.32
	ℓ_2 (Υ; RDFs)	304	0	0	0.36
Ablations:	$\mathcal{L}^{\mathrm{inv}}$	1	0.09	551.08	0.58
	$\mathcal{L}^{\mathrm{rad}}$	9	0.06	0.98	0.30
	$\mathcal{L}^{\mathrm{lab}}$	9	99.73	117.26	0.76
	$\mathcal{L}^{\mathrm{rad}}$ and $\mathcal{L}^{\mathrm{lab}}$	9	99.99	0.26	0.68
Our metric:	$\mathcal{L}^{\mathrm{inv}}$, $\mathcal{L}^{\mathrm{rad}}$, and $\mathcal{L}^{\mathrm{lab}}$	9	0.05	2.6	0.61

significant interest. We quantify this evaluation using the *median absolute residual error* (MARE) of the correlation, i.e. the median of $|\mathrm{d}\mathbf{z}(\mathbf{p}_i, \mathbf{p}_j) - \lambda^{\mathrm{rad}}\mathrm{d}\Upsilon(\mathbf{p}_i, \mathbf{p}_j)|$ over all pairs of points. For the chosen model, this error is 0.26 (units of $\mathrm{d}\mathbf{z}$), which is well below the overall distribution of distances in $\mathbf{Z}$ (see figures 3(middle) and (right)).

3.2. Model selection

A suitable model, $f_\theta^{\mathcal{L}}$, was selected by systematically evaluating models developed for ranges of relevant hyperparameters (w^{lab}, w^{inv}, w^{rad}, α^{lab}, α^{inv}, λ^{rad}), the dimensionality, d, of the embedding space, different network architectures (captured within the model parameters: θ), and ablation studies of the three conditions of interest (captured by the loss functions: $\mathcal{L}^{\mathrm{lab}}$, $\mathcal{L}^{\mathrm{inv}}$, and $\mathcal{L}^{\mathrm{rad}}$). See Appendix section 'Deep learning model development' and figure 10 for the architecture of the final model and values of the corresponding hyperparameters.

First, we compare our selected metric to standard, non-ML metrics used commonly in the community as well as several ablations of our approach (see table 1). The 'standard metrics' use ℓ_2 norms in the space of image pixels, $\mathbf{x}$ ($37 \times 37 \times 14$ dimensions), the mean concentration vector, μ (a 14D vector that represents mean per channel), and the radial profiles, Υ (38×8 dimensions). These metrics are chosen to reflect common approaches of comparing patches. As expected, these simpler methods and models fail to perform well on at least one of the desired criteria, whereas our metric provides the best balance among the criteria while maintaining a low dimensionality.

Next, figure 4 summarizes our evaluation of over 100 models using a parallel coordinates plot and highlights the chosen model (black line), which provided the best balance between the quality of the model (evaluation criteria) and the dimensionality of the embedding space. These models represent different hyperparameters, different architectures, as well as ablations (e.g. MARE $= -1$, i.e. an invalid value, indicates models without the radial profile condition). Zooming further on narrow ranges of the three criteria (figure 4(right)), we notice that several models offer good and comparable quality, providing empirical indication that our framework is generally robust to these specific choices. We note one 10D model (purple) that marginally outperforms the chosen model (black) that we chose to ignore in favor of a smaller dimensionality, which is also of import to the application for downstream analysis.

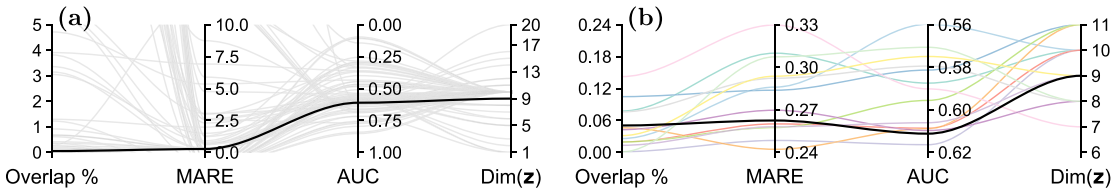

Figure 4. We evaluated over 100 models using three quantitative criteria and the dimension of the metric's embedding space. Each model is shown as a curve along the four axes of this parallel coordinates plot (lower is better on all axes). The chosen model (black) was selected by evaluating all models (a) and then focusing on a subset (b) that provided good and comparable performance.

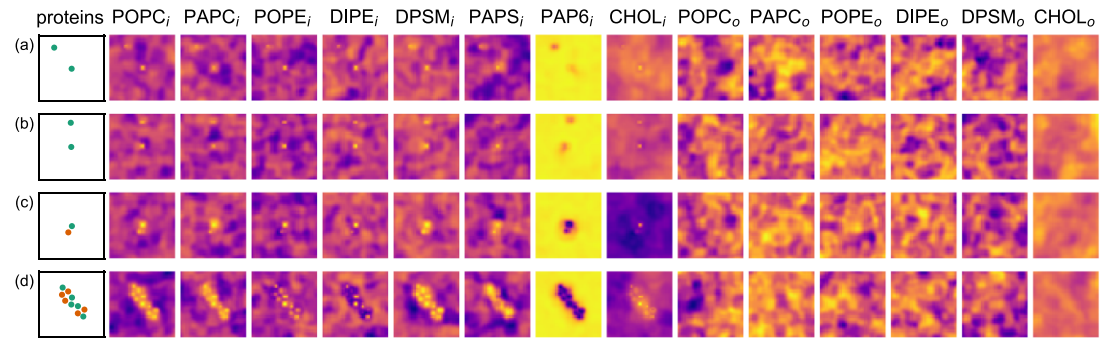

Figure 5. Visual depiction of our similarity metric for (a) a two-protein reference patch. (b) is a two-protein patch that is similar to (a); $d\mathbf{z}(a, b) = 0.262$. (c) is a two-protein patch that is dissimilar to (a); $d\mathbf{z}(a, c) = 4.052$. (d) is a ten-protein patch that is dissimilar to (a); $d\mathbf{z}(a, d) = 7.502$. Patch visualizations (rows) use the same layout as figure 1; colormap is same for each lipid (column).

3.3. Metric utility and generalizability to distributional shifts

The selected neural network was deployed on the *Summit* supercomputer as part of a large-scale workflow to facilitate the target multiscale simulation campaign[5], which ran for more than 3 months and consumed over 3.6 million GPU hours. This simulation campaign generated 6 180 000 patches (the '*campaign*' dataset) and our deep learning model was used for *in situ* inference to project patches into our metric's embedding space (i.e. $\mathbf{p} \rightarrow \mathbf{z}$). Next, we employ the dynamic-importance sampling (DynIm) framework [4] for selecting diverse patches for MD simulations. DynIm performs farthest-point sampling dynamically on the incoming streaming data (patches). At any point during the simulation, a given patch is either a previously selected patch or a candidate for current/future selection. We configure DynIm to use our metric to compute local densities of previously selected patches around each candidate patch, approximated as the mean distance of the candidate patch to its k-nearest neighbors in the set of all previously selected patches. Given a meaningful metric, the candidate patch with the largest mean distance, therefore, is the one that is most dissimilar to all previously selected patches. Continuing this process of farthest-point sampling leads to diverse subsets of interest. For coverage and convergence properties of the sampling algorithm, we refer the reader to the original publication on DynIm [4]. Here, we focus on demonstrating that our metric meaningfully captures the similarity.

We present retrospective analysis of our model's performance on the *campaign* dataset. First, we assess the quality of our metric visually by showing what the model considers as similar vs. dissimilar. Figure 5 shows a randomly chosen reference patch (a) with two proteins and its similarity ($d\mathbf{z}$) to three other patches with same and different protein counts. When compared against thousands of other two-protein patches (irrespective of the class label), (b) and (c) are found to be most similar and most dissimilar patches, respectively. The figure also shows a patch with ten proteins to illustrate that our metric returns high $d\mathbf{z}$ (low similarity) for this comparison.

3.3.1. Our metric performs well despite distributional shifts

Unlike the *pre-campaign* data (used for training), the *campaign* data (used for inference) is generated from a coupled simulation. To experts' surprise, the *campaign* data exhibits significantly greater extent of distributional shifts than expected (see Appendix section 'Distributional shifts in data: pre-campaign vs. campaign datesets'), most prominently in the concentrations of lipid PAP6 and the total number of proteins

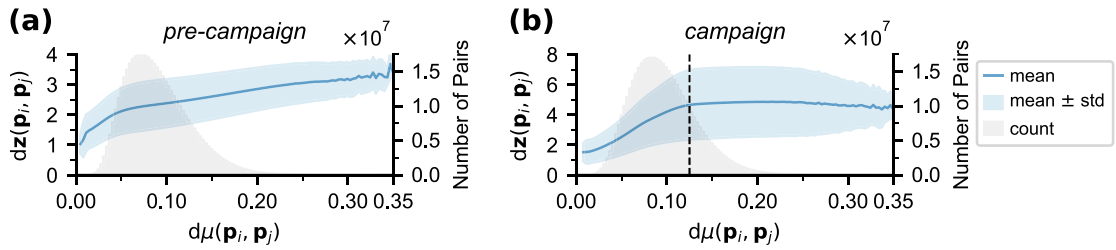

Figure 6. Our metric (d**z**) captures the differences in mean concentration vectors, $\mu(\mathbf{x})$, for both (a) *pre-campaign* and (b) *campaign* datasets. For (b), the simulation generates many new and previously unseen configurations, leading to a wider exploration in our metric's embedding space, resulting in a larger range of d**z**. Yet, the metric performs well on more than 78% of the data (left of vertical line; $d\mu \leqslant 0.125$). The Pearson correlation coefficient, R_p, for (a) is 0.957. For (b), R_p for the entire range on the horizontal axis is 0.727, whereas when considering the data for $d\mu \leqslant 0.125$, R_p is 0.996.

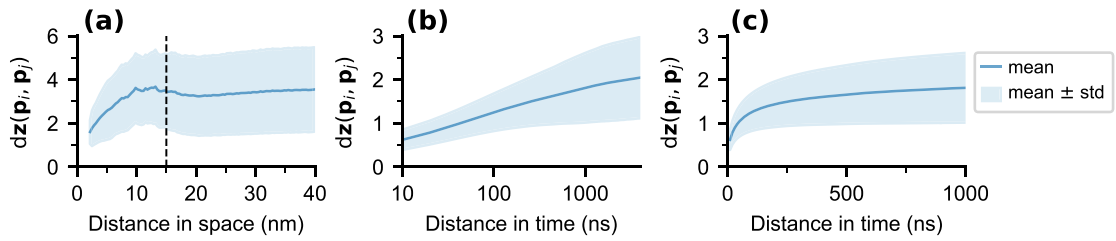

Figure 7. Despite trained agnostic to spatiotemporal locations of patches, our model captures inherent relationships with respect to both space and time. (a) As expected, patches with little to no overlap ($dx > 15$ nm apart, right of the vertical line) show no correlation with our metric. The Pearson correlation coefficient, R_p, for the two quantities is 0.919 for $dx \leqslant 15$. (b), (c) Our metric captures the high similarity existent at small temporal neighborhoods and also hints at the decorrelation time in the simulation, after which, patches are considered equally dissimilar. (b) and (c) show the same data with logarithmic and linear scale on the horizontal axis, respectively. R_p for this data is 0.81.

in the patch. This shift gets worse with progression of simulation time as more protein aggregation happens, emphasizing the need for generalizabity in our metric. As described above, we utilized our framework's flexibility to reduce the emphasis on the separability of protein constellations, which ultimately allowed us to support the *campaign* data, as a stronger emphasis on such separability would have led to poor inference. Instead, our framework absorbed such differences by resolving them in favor of the radial profiles. We support this claim by computing our evaluation criteria on a random subset of 30 000 patches taken from the *campaign* data and note that the model provides comparable quality, with overlap = 0.01%, MARE = 0.03, and AUC = 0.59 (compare with overlap = 0.05%, MARE = 0.26, and AUC = 0.61 for the *pre-campaign data*).

3.3.2. Our metric is superior to standard alternatives

We next evaluate the model against a standard alternative, mean lipid concentrations, μ—a 14D vector. Here, we show the correlation between the distances in the space of $d\mu$ with those given by the learned metric, d**z**, for the *pre-campaign* and the *campaign* data (see figure 6) and draw attention to two observations. First, the range of the horizontal axes in both plots is the same, which is the property of the simulated system. Although one expects to find fluctuations for individual lipids, the system itself is conserved (when certain lipids deplete, others replete the space) and, hence, the net ranges of the differences remain roughly the same. Second, given considerable shifts in lipid concentrations, the campaign data produces markedly different μ vectors that our metric captures by populating the extremes (previously unpopulated regions) of the learned embedding space, as indicated by the larger ranges of the similarity metric (vertical axes in the plots). Regardless, even for the *campaign* data, our metric performs well at preserving the differences in the mean concentrations for more than 78% of the data evaluated, whereas the extreme regions in the metric's embedding space (e.g. $d\mathbf{z} > 4$) appear to be less well understood.

3.3.3. Our metric understands the spatial locations of patches

Thus far, we have considered a patch as comprising only lipids and proteins. In the simulation, however, each patch has a position in space (and time). Whereas the spatial distance between patches is not a direct measure of similarity, the goal of preventing redundant simulations requires delivering high similarity for patches that

overlap spatially. Figure 7 shows our similarity metric between patches that are within same time-steps of the simulation. This result demonstrates that our metric naturally understands the spatial context of patches, even though it is trained without any explicit knowledge about the positions. Specifically, the metric correlates spatial positions with similarity for overlapping patches and shows no correlation between patches that are more than 15 nm (half the patch size) away.

3.3.4. Our metric understands decorrelation time of patches

We further demonstrate the biological relevance of our metric through the implicit connection between time and similarity measured as the *decorrelation time*. Specifically, there is an expectation that, given infinite time, any patch may evolve into any other one. Considering such evolution, a given patch is expected to remain considerably similar for short time periods, but become arbitrarily dissimilar for large enough time. This so-called decorrelation time (when patches along an evolution are not correlated anymore) is a key concept often used to guide sampling, I/O rates, and lengths of simulations. More importantly, the (shortest) time necessary for one patch to evolve into another represents a very intuitive and biologically relevant measure of similarity. However, given two arbitrary patches, directly computing this time scale is practically infeasible (an upper-bound estimate is about 500 ns, see Appendix section 'Derivation of decorrelation time').

To evaluate our metric in this context, we performed a *post-campaign* simulation that followed 300 unique RAS proteins and saved the data at 100× the temporal frequency used for training (*pre-campaign* data) and inference (*campaign* data), resulting in 300 histories that represent the temporal evolution of the respective patches. Figure 7 plots our similarity metric against the time difference between pairs of patches within the same history, i.e. around the same RAS protein. We note a strong correlation between both concepts for hundreds of ns, flattening only around 500 ns with a marked increase in standard deviation, which matches the expected decorrelation time past which there should not exist any relationship between patches. This result demonstrates that our metric, which is trained on patches at lower temporal resolution and without any explicit information on time, naturally learns the inherent correlation between patch similarity and evolution time. Most importantly, establishing this correlation, especially for shorter time scales, enables us, for the first time, to estimate the evolution time as a factor of the distance in the metric's embedding space. These insights will enable an entirely new viewpoint for the analysis of the resulting simulation by directly correlating observed properties of the membrane or the proteins with respect to their estimated difference in time.

4. Conclusion and discussion

We present a framework to compute similarities among data generated by a computational biology application crucial to cancer research. We address two key challenges. First, since complex scientific applications, such as the one presented here, work at the cutting edge of contemporary knowledge, well-formulated criteria for similarity are usually not available. In the absence of straightforward metrics, we instead utilize biology-informed criteria gathered from experts and cast them into a metric learning framework to learn a meaningful similarity metric. We show that our metric fuses these loosely defined conditions well and is robust to potential contradictions between them. Through close collaboration with subject matter experts (e.g. to identify relevant conditions and suitable features), our framework turns the lack of well-defined similarity criteria into a strength by imposing the experts' biology-informed intuitions as inductive biases that allow the metric to generalize to new simulations exhibiting significant distributional shifts in data—addressing the second challenge of deploying the model for *in situ* inference on unseen data. We demonstrate this generalization on two new datasets and show that our model learns key behavior of interest in the simulations, rather than focusing on the specific datasets themselves.

Our framework and the resulting similarity metric has been deployed as the key driving component in the first-of-its-kind multiscale simulation campaign[5] to explore RAS-RAF-membrane interactions, with a potential for significant impact in cancer biology. Unfortunately, the massive scale of such simulations (3–4 months, consuming 3–5 million GPU hours) makes it computationally infeasible to rerun the simulations to compare different metrics, models, techniques, or benchmarks, necessitating evaluation and comparisons in a proxy or development setting (table 1 and figure 4). Through suitable evaluation metrics and validation of external criteria (e.g. decorrelation time), this manuscript also addresses the challenges of a lack of ground truth to compare against and demonstrates a meaningful metric until the biology community can test more hypotheses and develop a more widely-accepted understanding of similarity. By showing a direct relationship between our metric and the expected evolution time between patches, our work also leads to new insights and opens up new directions of research.

Our data and experiments highlight that realistic simulation are not restricted to small distributional shifts. Therefore, despite our demonstration of generalizability, an opportunity for improvement lies in updating the model *in situ* using new data from the running simulations. As such, we will explore challenges, both computational and fundamental, associated with automatic detection of a model's suitability and online learning approaches to absorb new data to update the model.

Finally, although our work is presented in a specific application context, our framework is broadly applicable to other applications, such as experimental design and other types of autonomous multiscale simulations, that face such challenges. Specifically, our framework can be easily adapted to support different types of simulations in the space of biology or elsewhere using intuitions from the application. As examples, we have since customized our framework to support a different biological system where patches are periodic in both spatial dimensions, and we are also currently applying this framework to different biological systems with GPCR proteins [39].

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

This work has been supported in part by the Joint Design of Advanced Computing Solutions for Cancer (JDACS4C) program established by the U.S. Department of Energy (DOE) and the National Cancer Institute (NCI) of the National Institutes of Health (NIH). For computing time, we thank the Advanced Scientific Computing Research Leadership Computing Challenge (ALCC) for time on *Summit* and Livermore Computing (LC) and Livermore Institutional Grand Challenge for time on *Lassen*. For computing support we thank Oak Ridge Leadership Computing Facility and LC. We thank the entire JDACS4C Pilot 2 team, particularly the Pilot 2 leads Fred Streitz and Dwight V Nissley, for their support and helpful discussion. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Release LLNL-JRNL-822957.

Appendix

Computation of radial profiles

To compute radial profiles efficiently, we use predefined spatial kernels (see figure 8) that represent distance r from the reference pixel (i_*, j_*) (shown in red) rounded to the nearest integer; each kernel is defined for $r < 19$ pixels. For a given channel c within a patch $\mathbf{x}$, a radial aggregation (max) is applied using these kernels to create a 1D radial profile, $\gamma_{(i_*,j_*)}(\mathbf{x}, c, r)$. Each profile is represented as a $[19 \times 1]$ vector. To preserve invariance to transformations, the four corner profiles are averaged and concatenated to the center profile, resulting in a $[38 \times 1]$ vector. Figure 9 shows these vectors for each channel. Finally, in this work, we use the profiles of eight most-relevant channels are appended to create Υ, which is a feature of size $[38 \times 8]$.

Deep learning model development

Data preparation

A set of 334 000 patches was generated through a small and uncoupled continuum simulation run prior to the multiscale simulation campaign. This *pre-campaign* dataset was made available to us for developing the ML model. We used this dataset to train and evaluate several models as well as comparison against benchmarks (described in table 1 and figure 4).

The multichannel images, $\mathbf{x}$, were standardized per channel (to zero mean and unit standard deviation) across the dataset to account for differences in ranges of different lipid concentrations. Class labels for protein constellations, y, and radial profiles, Υ, were precomputed and saved as auxiliary information for model training. Patch transformations were computed during the training (online triplet mining).

Training details

Our training setup comprised TensorFlow v2.1 [1] and Keras v2.2.4 [8] using one NVIDIA Volta V100 GPU each. Models were optimized using the Adam optimizer [23], and most models used a piecewise-constant learning rate decay ($[1, 0.5, 0.1, 0.01] \times 10^{-3}$ switched after 10, 20, and 30 epochs). A 90%–10% random split of the dataset was used for training and validation. Training was performed until the total loss appeared converged, which took 60–100 epochs (4–6 h of walltime).

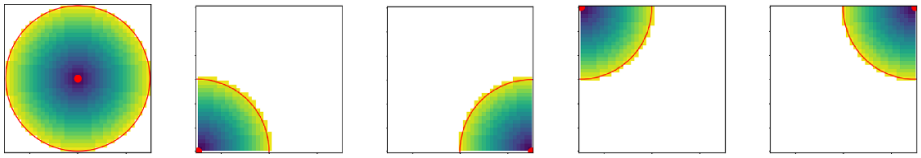

Figure 8. Spatial kernels for creating radial profiles. From left to right, kernels for center radial profile, $\gamma_{(18,18)}$ and the four corner profiles, $\gamma_{(0,0)}$, $\gamma_{(36,0)}$, $\gamma_{(0,36)}$, and $\gamma_{(36,36)}$ are shown.

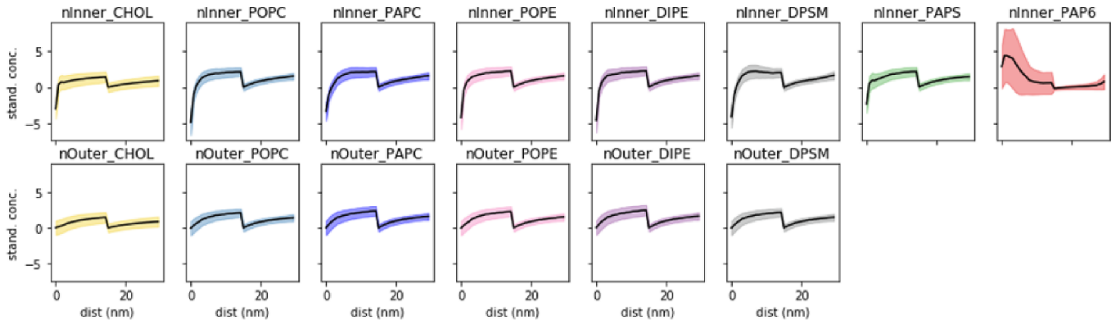

Figure 9. Mean (black) and standard deviation (color) of radial profiles for patches in the training dataset. The profiles are computed after the standardization of channels.

Network and hyperparameter search

We experimented with network architectures that comprised three to five convolution layers followed by three to four fully-connected layers. Whereas our framework appears generally robust to architecture changes, the chosen architecture gave superior results. Empirical evidence suggests that a separable convolution layer to account for correlations across lipid channels unsurprisingly improves the model performance. For searching the model's hyperparameters (w^{lab}, w^{inv}, w^{rad}, α^{lab}, α^{inv}, λ^{rad}, and d), we explored three values ([0.5, 1.0, 2.0]) for each of the weights, values [5, 8, 10, 12] for α^{lab} and values [0.1, 1] for α^{inv}. We deduced a suitable value for $\lambda^{\text{rad}} = 0.1$ through analysis of the ranges of $\mathrm{d}\Upsilon$. Finally, we experimented with varying d from 5 through 12 to look for an appropriate dimensionality of the embedding space. In total, we developed and evaluated 109 models (not counting the ablation studies summarized in table 1), searching through the hyperparameter ranges mentioned above. Instead of using a grid-based approach for all valid combinations of all parameters, we restricted our search carefully to promising possibilities. Specifically, we first identified two promising network architectures, then finalized the values of α^{lab}, α^{inv}, and λ^{rad}, and then looked at the different values for the weights and dimensionality.

Ablation studies

Highlighted in table 1, the different metrics optimize for individual criterion, e.g. the RDF-related metrics provide low overlap and low MARE* but perform poorly on AUC. On the other hand, simply combining two of the metrics does not necessary give additive benefits (e.g. $\mathcal{L}^{\text{rad}}$ and $\mathcal{L}^{\text{lab}}$). We note that our framework therefore produces a metric that is more useful than the sum of its parts and allows finding the right balance between the criteria, through balancing the hyperparameter weights. Beyond the quantitative evaluation (overlap, MARE*, and AUC), we also note the role of dimensionality of the corresponding space for computational efficiency. During the campaign, several thousand distance computations (for nearest neighbor queries) are to be made in real- time. Therefore, a large dimensionality makes the approach infeasible.

The final model

The neural network employed to learn this mapping $f_\theta^{\mathcal{L}}(\mathbf{x}, y) = \mathbf{z}$ can be formally described in terms of composable functions, each of which is represented by one or more components (known as layers or activations) of the neural network. The final model is formalized below, and an architecture diagram is given in figure 10. Notationally, the following formalism requires only $\mathbf{x}$ as input, because the labels y are passed only to compute the loss function.

$$f(\mathbf{x}) = (b \circ d) \circ \Lambda \circ (b \circ a \circ c_3) \circ (b \circ a \circ c_2) \circ (b \circ a \circ c_1)\ (\mathbf{x}).$$

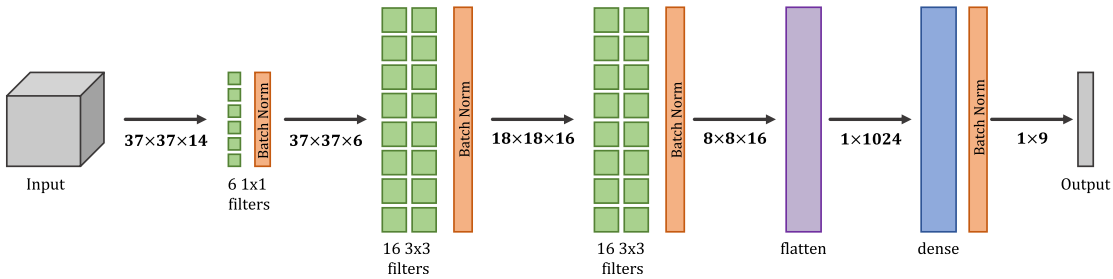

Figure 10. Architecture of the final model used to developed the proposed similarity metric. The different types of neural network layers are shown in different colors: convolutional layers in green, dense (fully-connected) layer in blue, flattening operation in purple, and batch normalization in orange. Arrows denote data passing through the neural network layers as well as to and from the input/output, which are shown in grey. Convolution layers are also followed by a rectified linear (ReLu) activation, not shown in the figure. We note that although the input to the neural network is a patch, $\mathbf{p} = (\mathbf{x}, y)$, the mapping itself works only on the images, $\mathbf{x}$. The labels, y, are used only to compute the loss functions; therefore, y is not shown in the figure. The hyperparameters associated with this chosen model are: $w^{\text{lab}} = 1.0$, $w^{\text{inv}} = 2.0$, $w^{\text{rad}} = 2.0$, $\alpha^{\text{lab}} = 8.0$, $\alpha^{\text{inv}} = 1.0$, $\lambda^{\text{rad}} = 0.1$, and $d = 9$.

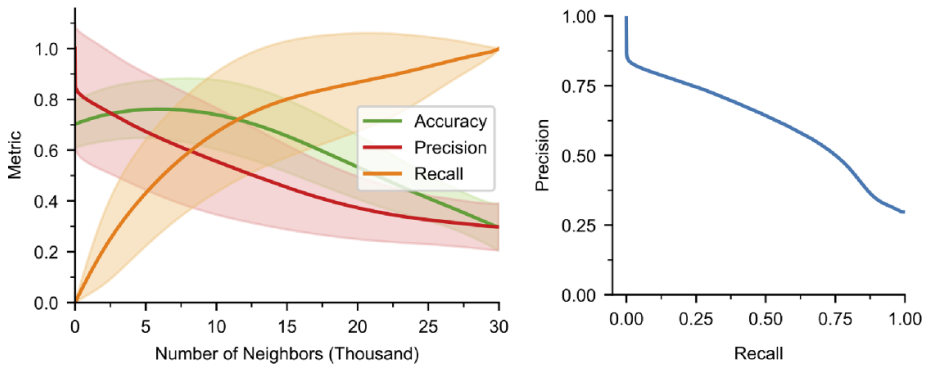

Figure 11. Accuracy, precision, and recall for our selected metric (left) with respect to the patch type and increasing neighborhoods in the metric space. We use the area under the precision-recall curve (right) to evaluate the metric.

For simplicity in formalization of these composable functions, we use a generic notations, x and X, for scalar and vector/tensor inputs, respectively, to these functions.

- Convolution 1: $c_1 = \mathbf{X} * {\mathbf{f}_1}^i$, where $\mathbf{f}^i$ for $0 \leqslant i < 6$ are learnable $[1 \times 1]$ filters.
- Convolution 2: $c_2 = \mathbf{X} * {\mathbf{f}_2}^i$, where $\mathbf{f}^i$ for $0 \leqslant i < 16$ are learnable $[3 \times 3]$ filters.
- Convolution 3: $c_3 = \mathbf{X} * {\mathbf{f}_3}^i$, where $\mathbf{f}^i$ for $0 \leqslant i < 16$ are learnable $[3 \times 3]$ filters.
- Dense layer: $d(\mathbf{X}) = \mathbf{W}\mathbf{X} + \mathbf{b}$, where $\mathbf{W}$ and $\mathbf{b}$ are learnable parameters.
- Flattening: Λ: $[8 \times 8 \times 16]$ data into $[1 \times 1024]$ (no learnable parameters).
- ReLu activation: $a(x) = \max(0, x)$ (no learnable parameters).
- Batch normalization: $b(x^{(k)}) = \frac{x^{(k)} - \mu^{(k)}}{\sqrt{\sigma^{(k)^2} + \epsilon}}$, where superscript (k) indicates normalization for each dimension individually (no learnable parameters).

Additional analysis on the final model

Precision-recall study for classification of class labels

To supplement the discussion in section 3.1 and figure 3, we present the precision-recall curve as well as accuracy of classification. To measure these quantities, we consider each point in the evaluation dataset (30 000 points), and note the types of patches found within neighborhoods as the number of neighbors are increased. Using the number of false positives and false negatives, we compute accuracy, precision, and recall using the standard formulations. Figure 11 shows the variations in these quantities with respect to the size of the neighborhoods. In this work, we are mostly interested in short-range neighborhoods and note that the accuracy remains high (greater than about 75%) for up to about 12 000 neighbors. The figure also shows precision-recall curve, whose area under the curve is used for quantitative evaluation of our metric.

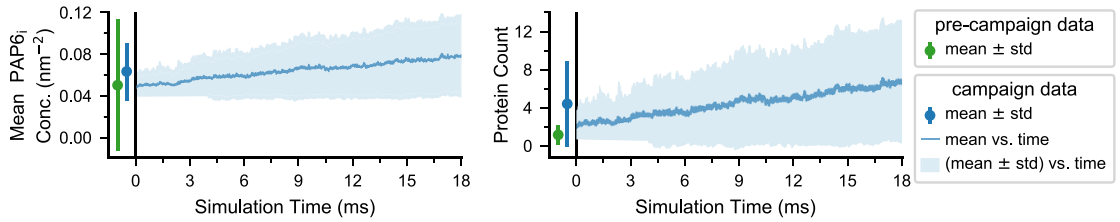

Figure 12. The distributional shifts observed between the *pre-campaign* data (used for training) and the *campaign* data (used for inference) highlights the need for generalizability of the similarity metric. The shifts become more drastic as the *campaign* simulation progresses in time.

Distributional shifts in data: pre-campaign vs. campaign datesets

The subject matter experts expect to explore significantly different types of patches during the simulation campaign than what may be modeled prior to the campaign. Although the continuum simulation during the campaign starts with the same parameters and models as available for pre-campaign, the primary reason of these anticipated differences is the evolution of a coupled multiscale system during the campaign. In particular, during the multiscale campaign, all of the several hundred thousand MD (fine scale) simulations are analyzed *in situ* and the resulting analysis is aggregated and used to improve the parameters of the continuum (coarse scale) model. As a result, the continuum model evolves and is expected to start exploring different regions of the phase space of patches. This *active feedback loop* is a key characteristic of such autonomous multiscale simulations [19].

In this work, the MD simulations indicate a higher degree of protein aggregation than previously hypothesized, which also leads to significant differences in lipid accumulation around the proteins. These shifts (highlighted in figure 12) can be consequential to any ML model since inference has to be made on types of patches that the model has not seen before. For example, inference on a patch that contains around 10 proteins, whereas training data contains patches with only up to 4 or 5 proteins.

One of the key challenges in our model development was to guard against such speculated yet not fully understood differences between the training and inference datasets, necessitating focusing on biology-informed inductive biases, rather than tailoring specifically to the data at hand.

Additional application details

Description of the simulation model

The biological system under study represents a cellular plasma membrane (PM) and the goal of our target simulations is to study the mechanism by which RAS and RAF localize on the PM, as this mechanism is hypothesized to be crucial for activating the signaling pathway, which leads to cancer.

The PM consists of two 'leaflets'—inner and outer, and our PM model is an average 8-lipid mixture [17–19]. In other words, there are eight types of lipids in this membrane model. The lipids are denoted using abbreviations as: POPC, PAPC, POPE, DIPE, DPSM, PAPS, PAP6, and CHOL. Of these, two lipids—PAPS and PAP6—exist only on the inner leaflet. Therefore, there are eight lipid species in the inner leaflet and six in the outer leaflet. The RAS protein approaches the membrane from the 'inner' side; as a result, the lipids in the inner leaflet respond strongly to the protein, as compared to those on the outer leaflet (as can be noted from figures 1 and 5). The RAF protein, also on the inner leaflet, binds to the RAS protein.

The *continuum simulation model*—the coarse scale in the target multiscale simulation—provides speed at the cost of accuracy. This continuum model uses the dynamic density functional theory [36] for representing lipid dynamics in terms of their density fields. Proteins are represented as individual particles that interact with each other and with the lipids. The continuum model simulates a 1 μm $\times$ 1 μm PM discretized as a 2400 $\times$ 2400 grid of cubic-order elements. The simulation contains 300 RAS proteins; a varying number of RAF proteins is present to model association/disassociation of RAS and RAF. A custom, scalable simulation code was developed using MPI to perform continuum model simulations. Using a total of 3600 MPI ranks, this code can simulate ~0.96 ms per day of walltime. With an I/O rate of 1 μs, a new continuum snapshot is delivered every 90 s.

Since we are interested in protein-lipid interactions, small, 30 nm $\times$ 30 nm regions, 'patches', around RAS proteins are 'cut out' from the domain—one for each RAS in the system. A patch may contain more than one protein. For control simulations, a small number of patches (30 per snapshot) are also cut out from random spatial locations such that they do not contain any proteins.

These patches are candidates for fine-scale, MD simulations, and form the input to our framework for identifying similarity. Each patch generated by the simulation contains lipid densities on a $37 \times 37 \times 14$ grid (the last dimension denoting lipids) and the positions, numbers, and types of proteins in the patch.

Derivation of decorrelation time

The coarse-scale, continuum simulation, from which patches are collected, computes the time evolution of the membrane system. Patches that evolve from times t_1 through t_2 ($t_1 \leqslant t_2$) are physically very similar if $t_2 - t_1$ is small, and become arbitrarily dissimilar as $t_2 - t_1$ goes to infinity. This presents an opportunity to verify that the metric indeed shows decreased similarity as a function of $t_2 - t_1$. Here we derive an estimate of how long it takes for a patch to get decorrelated.

In the continuum model, diffusion of lipids is a major source of entropy production and, thus, decorrelation. Given the system size and diffusivity, we can estimate the decorrelation time using the diffusion equation:

$$\frac{\partial c}{\partial t} = D\left(\frac{\partial^2 c}{\partial x^2} + \frac{\partial^2 c}{\partial y^2}\right),$$

where $c = c(x, y, t)$ is the lipid distribution (for some lipid species) and D is the corresponding diffusion coefficient. Given an $L \times L$ patch (for our data, $L = 30\,\text{nm}$) and a standard value of $D = 43.36\,\text{nm}^2\,\mu\text{s}^{-1}$ (taken as average over several relevant lipids). The slowest decaying Fourier mode of the diffusion equation above decays as $\exp\left[-D\left(\frac{2\pi}{L}\right)^2 t\right]$, where using the values above we get $D\left(\frac{2\pi}{L}\right)^2 \approx 0.53\,\mu\text{s}$.

The motion of the protein and inherent randomness (noise) in the lipid evolution equations introduce further entropy and thus shortens the decorrelation time compared to the above estimate. We can see that figure 7 shows a decorrelation time roughly similar to this diffusive estimate.

ORCID iDs

Harsh Bhatia https://orcid.org/0000-0001-8712-7773
Rushil Anirudh https://orcid.org/0000-0002-4186-3502
Tomas Oppelstrup https://orcid.org/0000-0002-6366-7190
Helgi I Ingólfsson https://orcid.org/0000-0002-7613-9143
Peer-Timo Bremer https://orcid.org/0000-0003-4107-3831

References

[1] Abadi M *et al* 2016 TensorFlow: a system for large-scale machine learning *12th USENIX Symp. on Operating Systems Design and Implementation (OSDI)* pp 265–83
[2] Ayton G S and Voth G A 2010 Multiscale simulation of protein mediated membrane remodeling *Semin. Cell Dev. Biol.* **21** 357–62
[3] Bai Z, Zhang X-L and Chen J 2020 Speaker verification by partial AUC optimization with mahalanobis distance metric learning *IEEE/ACM Trans. Audio Speech Lang. Process.* **28** 1533–48
[4] Bhatia H *et al* 2021 Machine learning based dynamic-importance sampling for adaptive multiscale simulations *Nat. Mach. Intell.* **3** 401–9
[5] Bhatia H *et al* 2021 Generalizable coordination of large multiscale workflows: challenges and learnings at scale *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Analysis, SC '21* (New York: ACM)
[6] Bhowmik D, Gao S, Young M T and Ramanathan A 2018 Deep clustering of protein folding simulations *BMC Bioinform.* **19** 484
[7] Chen W, Chen X, Zhang J and Huang K 2017 Beyond triplet loss: a deep quadruplet network for person re-identification *2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (Los Alamitos, CA: IEEE Computer Society) pp 1320–9
[8] Chollet F *et al* 2015 Keras (available at: https://github.com/fchollet/keras)
[9] Chopra S, Hadsell R and LeCun Y 2005 Learning a similarity metric discriminatively, with application to face verification *2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'05)* vol 1 pp 539–46
[10] Dupuis P, Spiliopoulos K and Wang H 2012 Importance sampling for multiscale diffusions *Multiscale Model. Simul.* **10** 1–27
[11] Enkavi G, Javanainen M, Kulig W, Róg T and Vattulainen I 2019 Multiscale simulations of biological membranes: the challenge to understand biological phenomena in a living substance *Chem. Rev.* **119** 5607–774
[12] Ge W, Huang W, Dong D and Scott M R 2018 Deep metric learning with hierarchical triplet loss *Computer Vision—ECCV 2018* ed V Ferrari, M Hebert, C Sminchisescu and Y Weiss (Cham: Springer) pp 272–88
[13] Hadsell R, Chopra S and LeCun Y 2006 Dimensionality reduction by learning an invariant mapping *2006 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'06)* vol 2 pp 1735–42
[14] Hoekstra A, Chopard B and Coveney P 2014 Multiscale modelling and simulation: a position paper *Phil. Trans. R. Soc.* A **372** 20130377
[15] Hoffer E and Ailon N 2015 Deep metric learning using triplet network *Similarity-Based Pattern Recognition* ed A Feragen, M Pelillo and M Loog (Cham: Springer) pp 84–92
[16] Hu H, Wang K, Lv C, Wu J and Yang Z 2019 Semi-supervised metric learning-based anchor graph hashing for large-scale image retrieval *IEEE Trans. Image Process.* **28** 739–54
[17] Ingólfsson H I *et al* 2020 Capturing biologically complex tissue-specific membranes at different levels of compositional complexity *J. Phys. Chem.* B **124** 7819–29

[18] Ingólfsson H I, Carpenter T S, Bhatia H, Bremer P-T, Marrink S J and Lightstone F C 2017 Computational lipidomics of the neuronal plasma membrane *Biophys. J.* **113** 2271–80
[19] Ingólfsson H I *et al* 2022 Machine learning-driven multiscale modeling reveals lipid-dependent dynamics of RAS signaling proteins *Proc. Natl Acad. Sci.* **119** e2113297119
[20] Jacobs S A *et al* 2021 Enabling rapid COVID-19 small molecule drug design through scalable deep learning of generative models *Int. J. High Perform. Comput. Appl.* **35** 469–82
[21] Kaya M and Bilge H Ş 2019 Deep metric learning: a survey *Symmetry* **11** 1066
[22] Kessler D *et al* 2019 Drugging an undruggable pocket on KRAS *Proc. Natl Acad. Sci.* **116** 15823–9
[23] Kingma D P and Ba J 2017 Adam: a method for stochastic optimization (arXiv:1412.6980)
[24] Knap J, Barton N R, Hornung R D, Arsenlis A, Becker R and Jefferson D R 2008 Adaptive sampling in hierarchical simulation *Int. J. Numer. Methods Eng.* **76** 572–600
[25] Krzhizhanovskaya V, Groen D, Bozak B and Hoekstra A 2015 Multiscale modelling and simulation workshop: 12 years of inspiration *Proc. Comput. Sci.* **51** 1082–7
[26] Li R, Jiang J-Y, Li J L, Hsieh C-C and Wang W 2020 Automatic speaker recognition with limited data *Proc. 13th Int. Conf. on Web Search and Data Mining (WSDM'20)* (New York: Association for Computing Machinery) pp 340–8
[27] Li T, Kou G and Peng Y 2020 Improving malicious urls detection via feature engineering: linear and nonlinear space transformation methods *Inf. Syst.* **91** 101494
[28] Li X and Tang Y 2020 A social recommendation based on metric learning and network embedding *2020 IEEE 5th Int. Conf. on Cloud Computing and Big Data Analytics (ICCCBDA)* pp 55–60
[29] Liu Y, Gu Z, Ko T H and Liu J 2018 Multi-modal media retrieval via distance metric learning for potential customer discovery *2018 IEEE/WIC/ACM Int. Conf. on Web Intelligence (WI)* pp 310–7
[30] Liu Y, Pi D and Cui L 2017 Metric learning combining with boosting for user distance measure in multiple social networks *IEEE Access* **5** 19342–51
[31] López-Sánchez D, Arrieta A G and Corchado J M 2019 Visual content-based web page categorization with deep transfer learning and metric learning *Neurocomputing* **338** 418–31
[32] Lu J, Hu J and Zhou J 2017 Deep metric learning for visual understanding: an overview of recent advances *IEEE Signal Process. Mag.* **34** 76–84
[33] Luo Y, Hu H, Wen Y and Tao D 2020 Transforming device fingerprinting for wireless security via online multitask metric learning *IEEE Internet Things J.* **7** 208–19
[34] Ma Z, Zhou S, Wu X, Zhang H, Yan W, Sun S and Zhou J 2019 Nasopharyngeal carcinoma segmentation based on enhanced convolutional neural networks using multi-modal metric learning *Phys. Med. Biol.* **64** 025005
[35] Manohar K, Kaiser E, Brunton S L and Kutz J N 2019 Optimized sampling for multiscale dynamics *Multiscale Model. Simul.* **17** 117–36
[36] Marconi U M and Tarazona P 1999 Dynamic density functional theory of fluids *J. Chem. Phys.* **110** 8032–44
[37] Nguyen B and De Baets B 2019 Kernel distance metric learning using pairwise constraints for person re-identification *IEEE Trans. Image Process.* **28** 589–600
[38] Prior I A, Hood F E and Hartley J L 2020 The frequency of RAS mutations in cancer *Cancer Res.* **80** 2969–74
[39] Rosenbaum D M, Rasmussen S G F and Kobilka B K 2009 The structure and function of G-protein-coupled receptors *Nature* **459** 356–63
[40] Rouet-Leduc B *et al* 2014 Spatial adaptive sampling in multiscale simulation *Comput. Phys. Commun.* **185** 1857–64
[41] Schroff F, Kalenichenko D and Philbin J 2015 FaceNet: a unified embedding for face recognition and clustering *2015 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* pp 815–23
[42] Schultz M and Joachims T 2004 Learning a distance metric from relative comparisons *Advances in Neural Information Processing Systems* vol 16 (MIT Press) pp 41–48
[43] Simanshu D K, Nissley D V and McCormick F 2017 RAS proteins and their regulators in human disease *Cell* **170** 17–33
[44] Suárez-Díaz J L, García S and Herrera F 2020 A tutorial on distance metric learning: mathematical foundations, algorithms, experimental analysis, prospects and challenges (with appendices on mathematical background and detailed algorithms explanation) (arXiv:1812.05944)
[45] Travers T, López C A, Van Q N, Neale C, Tonelli M, Stephen A G and Gnanakaran S 2018 Molecular recognition of RAS/RAF complex at the membrane: role of RAF cysteine-rich domain *Sci. Rep.* **8** 8461
[46] van der Maaten L and Hinton G 2008 Visualizing data using t-SNE *J. Mach. Learn. Res.* **9** 2579–605
[47] Voth G A 2017 A multiscale description of biomolecular active matter: the chemistry underlying many life processes *Acc. Chem. Res.* **50** 594–8
[48] Wang C, Peng G and De Baets B 2020 Deep feature fusion through adaptive discriminative metric learning for scene recognition *Inf. Fusion* **63** 1–12
[49] Waters A M and Der C J 2018 KRAS: the critical driver and therapeutic target for pancreatic cancer *Cold Spring Harb. Perspect. Med.* **8** a031435
[50] Wei G, Qiu M, Zhang K, Li M, Wei D, Li Y, Liu P, Cao H, Xing M and Yang F 2020 A multi-feature image retrieval scheme for pulmonary nodule diagnosis *Medicine* **99** e18724
[51] Wu H, Zhou Q, Nie R and Cao J 2020 Effective metric learning with co-occurrence embedding for collaborative recommendations *Neural Netw.* **124** 308–18
[52] Xing E, Ng A Y, Jordan M I and Russell S J 2003 Distance metric learning with application to clustering with side-information *Advances in Neural Information Processing Systems* vol 15, ed S Becker, S Thrun and K Obermayer (MIT Press) pp 521–8
[53] Zhao C, Wang X, Zuo W, Shen F, Shao L and Miao D 2020 Similarity learning with joint transfer constraints for person re-identification *Pattern Recognit.* **97** 107014