Journal Title Here, 2024, pp. 1–10

doi: DOI HERE
Advance Access Publication Date: Day Month Year
Paper

PAPER

Aggregating Residue-Level Protein Language Model Embeddings with Optimal Transport

Navid NaderiAlizadeh[1,*] and Rohit Singh[1,2]

[1]Department of Biostatistics and Bioinformatics, Duke University, Durham, 27705, NC, USA and [2]Department of Cell Biology, Duke University, Durham, 27705, NC, USA
[*]Corresponding author. Email: navid.naderi@duke.edu

Abstract

Protein language models (PLMs) have emerged as powerful approaches for mapping protein sequences into embeddings suitable for various applications. As protein representation schemes, PLMs generate per-token (i.e., per-residue) representations, resulting in variable-sized outputs based on protein length. This variability poses a challenge for protein-level prediction tasks that require uniform-sized embeddings for consistent analysis across different proteins. Previous work has typically used average pooling to summarize token-level PLM outputs, but it is unclear whether this method effectively prioritizes the relevant information across token-level representations. We introduce a novel method utilizing optimal transport to convert variable-length PLM outputs into fixed-length representations. We conceptualize per-token PLM outputs as samples from a probabilistic distribution and employ sliced-Wasserstein distances to map these samples against a reference set, creating a Euclidean embedding in the output space. The resulting embedding is agnostic to the length of the input and represents the entire protein. We demonstrate the superiority of our method over average pooling for several downstream prediction tasks, particularly with constrained PLM sizes, enabling smaller-scale PLMs to match or exceed the performance of average-pooled larger-scale PLMs. Our aggregation scheme is especially effective for longer protein sequences by capturing essential information that might be lost through average pooling.

Key words: Protein Language Models, Representation Learning, Optimal Transport, Sliced-Wasserstein Embedding, Protein Function Prediction

Introduction

Understanding the sequence–structure–function relationship for proteins is one of the grand challenges of biology. Among the problems defined around these relationships, a prominent class of problems comprises tasks where some structural or functional property of a protein is to be predicted from its sequence. The possible set of prediction tasks in this class is very diverse, including both classification (e.g., "is the protein a kinase?") and regression (e.g., the melting temperature of the protein), as well as tasks involving auxiliary inputs (e.g., small molecule SMILES representations, for predicting drug-target interactions). Any such problem can be formulated as a prediction task over a set of proteins, where the input consists of a protein sequence and the output (a label or number) is at the level of the *entire protein*, rather than its constituent amino acids. Any model that addresses such a problem formulation will necessarily have one or more steps where information across the constituent amino acids of the protein is summarized into a protein-level estimate.

The problem of appropriately aggregating amino acid-level information has become particularly pressing with the advent of protein language models (PLMs). PLMs, which are trained on massive corpora of protein sequence data using self-supervised learning, are able to build internal representations that capture evolutionary constraints on protein sequences. Since these representations comprehensively capture constraints on protein function and structure, PLMs have proven powerful in a wide range of tasks, from structure prediction to interaction prediction and protein design. For many protein sequence-based property-prediction tasks, PLM-based approaches are now the state-of-the-art [25; 18; 39]. Given a sequence of length n, a pre-trained PLM produces an embedding of dimensionality $\mathbb{R}^{n \times d}$, where d is the per-amino acid (i.e., per-token) embedding dimensionality. The token-specific embedding captures not only biochemical information about the token (i.e., the amino acid) but also the local and global properties of the protein.

To aggregate these per-token embeddings into a protein-level representation, the most common approach has been to simply "average pool:" take the mean of each feature dimension along the length of the protein to produce an embedding in $\mathbb{R}^d$ [4; 46; 47; 39; 40]. While other pooling approaches, such as "max pooling" (taking the maximum of the set) [12] or "softmax pooling" (i.e., average pooling after exponentiation, then logarithmized) [38] are also sometimes used, average pooling is typically preferred for convenience, speed, and simplicity. However, it weighs each amino acid's representation equally. This is unrealistic—often, there are specific residues

© The Author 2024. Published by Oxford University Press. All rights reserved. For permissions, please e-mail: journals.permissions@oup.com

in the protein that are particularly important (e.g., the residues at an active site). Even when the per-token PLM representation does contain information distinguishing such residues from others, these distinctions might be lost during average pooling, which could be especially problematic for longer protein sequences. We note that such considerations are not limited to PLM-based embeddings: in many task-specific neural network architectures, e.g., PIPR for PPI prediction [8], average pooling is used to summarize variable-length intermediate representations.

In this work, we present a novel approach to aggregate variable-length protein representations. Like average pooling, max pooling, and related approaches, our method is permutation invariant: it considers the per-token embeddings as a *set* rather than a *sequence*, under the assumption that the PLM backbone fully embeds the sequential properties of the protein data in its output residue-level representations. Let $\mathbf{p}_j$ be a protein of length n_j, so that its PLM embedding can be represented as $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$, with each $\mathbf{x}_{ij} \in \mathbb{R}^d$. Our work seeks to learn a set of m reference embeddings $\{\mathbf{x}_{i0}\}_{i=1}^{m}$, with $\mathbf{x}_{i0} \in \mathbb{R}^d$, that can characterize any variable-length representation. Conceptually, this is analogous to learning a task-specific basis representation in $\mathbb{R}^d$. The intuition underpinning our work is to think of $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$ and $\{\mathbf{x}_{i0}\}_{i=1}^{m}$ as empirical probability distributions in $\mathbb{R}^d$ and to formulate $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$'s distance from the reference set $\{\mathbf{x}_{i0}\}_{i=1}^{m}$ as an optimal transport calculation.

Our work builds upon optimal transport (OT) based approaches in computer vision to characterize sets of observations. We borrow from previous advances to deploy the OT intuition effectively and in a scalable fashion. In particular, we use "slices" in the embedding space, learnable directions in $\mathbb{R}^d$ onto which the input and reference token-sets $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$ and $\{\mathbf{x}_{i0}\}_{i=1}^{m}$ are projected. These projections correspond to 1-D probability distributions. On such distributions, OT distances can be computed efficiently, and an ensemble of L slices serves to efficiently characterize the separation between input and reference sets.

The key conceptual advance of our work is unlocking task-specific learnability as a key component of PLM-based machine learning models. Broadly, these models can be thought of as pipelines of three segments: an initial *sequence* segment, a *summarization* segment, and a final *prediction* segment. For example, the sequence segment may consist of transformer layers, while the prediction segment may consist of a feed-forward network. However, if average pooling is used for summarization, no task-specific learning can happen there and will need to happen only in the sequence or the prediction segment. With our innovation, the summarization segment also becomes learnable, offering greater flexibility in matching the architecture of the neural network to the biological intuitions underlying the task.

Our work also has the potential to introduce interpretability into systems that would otherwise be opaque. The set of m reference embeddings learned by the system can serve as useful archetypes for the task at hand. For instance, in the computer vision context, it was shown that simply being able to associate the variable-length representation with one of the references can be informative about the typicality of the underlying object.

We apply our sliced-Wasserstein embedding (SWE) approach to four protein property prediction tasks: binary drug-target interaction prediction, out-of-domain drug-target affinity prediction, subcellular localization, and enzyme commission (EC) prediction. SWE broadly outperforms other pooling mechanisms, including average pooling, especially on small and moderate-sized PLMs. Notably, the onerous GPU memory requirements of the largest PLM architectures suggest that the performance boost of SWE could be critical to democratizing access to PLMs for researchers with limited GPU resources. SWE especially shines over average pooling for longer protein sequences, hence confirming our hypothesis that average pooling results in a more substantial loss of information when dealing with larger numbers of token-level representations.

Related Work

Protein Language Models

Large language models (LLMs) have become the predominant tools for modeling sequential natural language data. The success of LLMs, which mostly rely on attention-based transformer architectures [48], has inspired researchers working with biological data to use similar ideas for analyzing protein sequences. In particular, the availability of massive protein sequence datasets has given rise to large-scale protein language models (PLMs), such as ESM [36; 24], ProtBert [13], ProtGPT2 [14], SaProt [43], xTrimoPGLM [7], and DPLM [52], to name a few. These models are mostly trained using unlabeled protein sequence data in a self-supervised way, where the goal is to train the model to predict a token that has been replaced with a special *mask* token using its surrounding context, i.e., other amino acids in the sequence. Such masking-based unsupervised training leads to token-level representations that have been shown to provide state-of-the-art performance in a wide array of downstream tasks, such as protein folding [50], variant effect prediction [6], peptide generation [9], antibody design [53], and prokaryotic gene prediction [45].

The transformer architectures in LLMs, in general, and PLMs, in particular, produce residue-level representations that need to be summarized and aggregated for protein-level downstream tasks since different amino acid sequences have varying lengths. The question of aggregating a set of elements into a fixed-length representation is the key behind the research on *set representation learning*, which we discuss next.

Set Representation Learning

The goal of set representation learning is to map an unordered collection of elements into an embedding that is invariant to the permutation of the set elements and whose size is independent of the input set size [35; 51]. Deep Sets [54] and Janossy Pooling [30] are two seminal studies in this area, where the set embedding is modeled as a function of the sum or average of permutation-sensitive functions applied to all elements or all permutations of the input set. Follow-up work has leveraged ideas based on transformers [22] and optimal transport [20; 32; 31; 28], among others, for deep learning on sets, and demonstrated their efficacy in a variety of learning settings, including point cloud classification [34], graph representation [29], and multi-agent reinforcement learning [44].

In the context of PLMs, prior work has, for the most part, used *average pooling* to summarize the token-level embeddings into universal protein-level embeddings [4; 46; 47; 39; 40], and past research on other pooling methods is scarce [41; 42; 17; 12]. Averaging is a simple, unparameterized, permutation-invariant, and size-invariant function, which warrants its selection as the most natural and intuitive choice for aggregating PLM outputs. Nevertheless, it is unknown whether other, more sophisticated aggregation mechanisms could unlock additional performance gains compared to average pooling when used in conjunction with state-of-the-art PLMs. In this paper,

we give an affirmative answer to this question by proposing a parameterized aggregation operation based on ideas from optimal transport to summarize residue-level embeddings generated by pre-trained PLMs into fixed-length protein-level embeddings.

Methods

Problem Formulation

Consider a protein's primary amino acid sequence of length $n \in \mathbb{N}$, denoted by $\mathbf{p} = (p_1, \ldots, p_n) \in \mathcal{P}^n$, where $\mathbb{N}$ denotes the set of natural numbers, i.e., positive integers, $\mathcal{P}$ represents the residue alphabet. We gather the set of all possible protein sequences of arbitrary lengths into a set

$$\mathcal{X} = \bigcup_{n \in \mathbb{N}} \mathcal{P}^n. \tag{1}$$

The goal of protein-level representation learning is to find a function $\psi(\cdot; \theta_\psi) : \mathcal{X} \to \mathbb{R}^d$, parameterized by a finite-dimensional set of parameters $\theta_\psi \in \Theta_\psi$ (see Figure 1-a). Observe that the dimensionality of the embedding space, i.e., d, and the size of the model parameter space, i.e., $|\Theta|$, are *independent* of the length of the input protein sequence. In other words, the function $\psi(\cdot; \theta_\psi)$ should be able to map any given protein sequence of arbitrary length to a fixed-size representation in $\mathbb{R}^d$.

Assuming the availability of a (labeled) protein sequence dataset $\{(\mathbf{p}_j, y_j)\}_{j=1}^N$, where $\mathbf{p}_j \in \mathcal{X}, y_j \in \mathcal{Y}, \forall j \in \{1, \ldots, N\}$, with $\mathcal{Y}$ denoting the set of all possible labels, the parameters of the representation function ψ are typically derived via an empirical risk minimization (ERM) problem,

$$\min_{\theta_\psi \in \Theta_\psi} \frac{1}{N} \sum_{j=1}^{N} \ell(\psi(\mathbf{p}_j; \theta_\psi), y_j), \tag{2}$$

where $\ell : \mathbb{R}^d \times \mathcal{Y} \to \mathbb{R}$ denotes a loss function. While resembling a supervised learning setting, note that the formulation in (2) also includes an unsupervised learning scenario, where the labels y_j are trivial/non-existent and the loss function ℓ does not depend on y.

A common initial building block used to parameterize the representation function ψ is a protein language model (PLM). We denote a PLM by a parameterized function $\phi(\cdot, \theta_\phi) : \mathcal{P}^n \to (\mathbb{R}^d)^n, \forall n \in \mathbb{N}$. This implies that a given PLM maps each amino acid in an input protein sequence into an individual embedding in $\mathbb{R}^d$. The parameters of the PLM, i.e., θ_ϕ, are usually trained by a masking-based objective, where some amino acid identities in the input sequences are masked by random tokens, and the model is trained to predict the correct amino acids using the corresponding output representations.

In this paper, our goal is on the aggregation, or pooling, function that bridges the gap between PLM-generated token-level outputs and the final protein-level representation. More formally, we are interested in an informative aggregation function $\pi(\cdot; \theta_\pi) : (\mathbb{R}^d)^n \to \mathbb{R}, \forall n \in \mathbb{N}$, parameterized by $\theta_\pi \in \Theta_\pi$. Taken together, the composition of the PLM ϕ and the aggregation function π constitutes the end-to-end protein-level representation learning pipeline (see Figure 1-b); for any given protein sequence $\mathbf{p} \in \mathcal{X}$, its representation is derived as

$$\psi(\mathbf{p}; \theta_\psi) = \psi(\mathbf{p}; \theta_\phi, \theta_\pi) = \pi\Big(\phi(\mathbf{p}; \theta_\phi); \theta_\pi\Big) \in \mathbb{R}^d. \tag{3}$$

We assume that the PLM is pre-trained and its parameters, θ_ϕ, are frozen. Therefore, we are primarily interested in training the aggregation function $\pi(\cdot; \theta_\pi)$ for downstream prediction tasks.

Proposed Pooling Method

A simple and prominent example of an aggregation function is the *averaging* operation, i.e., $\pi_{\mathsf{avg}}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$, which is indeed unparameterized (i.e., $\Theta_\psi = \emptyset$; see Figure 1-c). While used extensively in prior work in the protein representation learning literature (see, e.g., [40; 39]), average pooling may not capture the entirety of the information that is present in the per-token embeddings. This calls for methods that are able to capture such information in the aggregated representations.

In this paper, we propose to use sliced-Wasserstein distances from optimal transport [11; 21], and in particular, *sliced-Wasserstein embedding* (SWE) [31; 28] to aggregate the token-level representations into a universal protein-level presentation. The main idea behind SWE is to treat the token-level embeddings as samples drawn from an underlying probability distribution, and then find the optimal transportation plan that maps that distribution to a trainable reference distribution (see Figure 1-d).

More formally, let $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$ denote the token-level embeddings of the jth protein sequence in the dataset, $\mathbf{p}_j, j \in \{1, \ldots, N\}$, of length n_j, which are produced by the pre-trained PLM; i.e., $(\mathbf{x}_{1j}, \ldots, \mathbf{x}_{n_j j}) = \phi(\mathbf{p}_j; \theta_\phi)$. We assume that the embeddings $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}$ are samples drawn from an underlying distribution $\mathfrak{D}_j$ supported on $\mathbb{R}^d$. We also consider a trainable reference set $\{\mathbf{x}_{i0}\}_{i=1}^m$ of m points in $\mathbb{R}^d$ that are drawn from a reference distribution $\mathfrak{D}_0$. Since there is no closed-form solution for calculating the optimal Monge coupling [49] between high-dimensional distributions in $\mathbb{R}^d$, we resort to *slicing* operations in order to map these distributions to several one-dimensional distributions, for which the Monge coupling has a closed-form solution. In particular, we consider trainable linear maps $\{\omega_l\}_{l=1}^L$, with $\omega_l \in \mathbb{R}^d, \forall l \in \{1, \ldots, L\}$, through which each set of token-level embeddings $\{\mathbf{x}_{ij}\}_{i=1}^{n_j}, j \in \{0, \ldots, N\}$ (including the reference embeddings with $n_0 = m$) is mapped to L *slices*, i.e., sets of one-dimensional points $\{u_{ij}^l\}_{i=1}^{n_j}, l \in \{1, \ldots, L\}$, where

$$u_{ij}^l = \omega_l^T \mathbf{x}_{ij}, \quad \forall i \in \{1, \ldots, n_j\}, \forall j \in \{0, \ldots, N\}, \forall l \in \{1, \ldots, L\}.$$

Consider the lth slice of a set of token-level embeddings, i.e., $\{u_{ij}^l\}_{i=1}^{n_j}$ and the corresponding slice of the reference set, i.e., $\{u_{i0}^l\}_{i=1}^m$. These sets correspond to empirical 1-D distributions $\hat{\mathfrak{D}}_j^l(u) = \frac{1}{n_j}\sum_{i=1}^{n_j} \delta(u - u_{ij}^l)$ and $\hat{\mathfrak{D}}_0^l(u) = \frac{1}{m}\sum_{i=1}^m \delta(u - u_{i0}^l)$, respectively. The Monge coupling between $\hat{\mathfrak{D}}_j^l(u)$ and $\hat{\mathfrak{D}}_0^l(u)$ has a closed-form solution, and in particular is an m-dimensional vector $\mathbf{z}_j^l = [z_{1j}^l, \ldots, z_{mj}^l]^T$ which relies solely on sorting and interpolation operations. In particular, depending on whether the length of the input protein sequence and the size of the reference set are identical, there are two cases:

- **Case 1** ($n_j = m$): In this scenario, the Monge map is simply the difference between the sorted sequences of points in the input slice and the reference slice. More precisely, let $\rho[\cdot]$ denote the permutation indices obtained by sorting $\{u_{ij}^l\}_{i=1}^{n_j}$. Moreover, let $\rho_0^{-1}[\cdot]$ denote the ordering that permutes the sorted set back to the original ordering based on sorting of elements in the reference set $\{u_{i0}^l\}_{i=1}^m$. Then, the Monge map elements are given by

$$z_{ij}^l = u_{i'j}^l - u_{i0}^l, \ \forall i \in \{1, \ldots, m\}, \tag{4}$$

  where $i' = \rho[\rho_0^{-1}[i]]$.

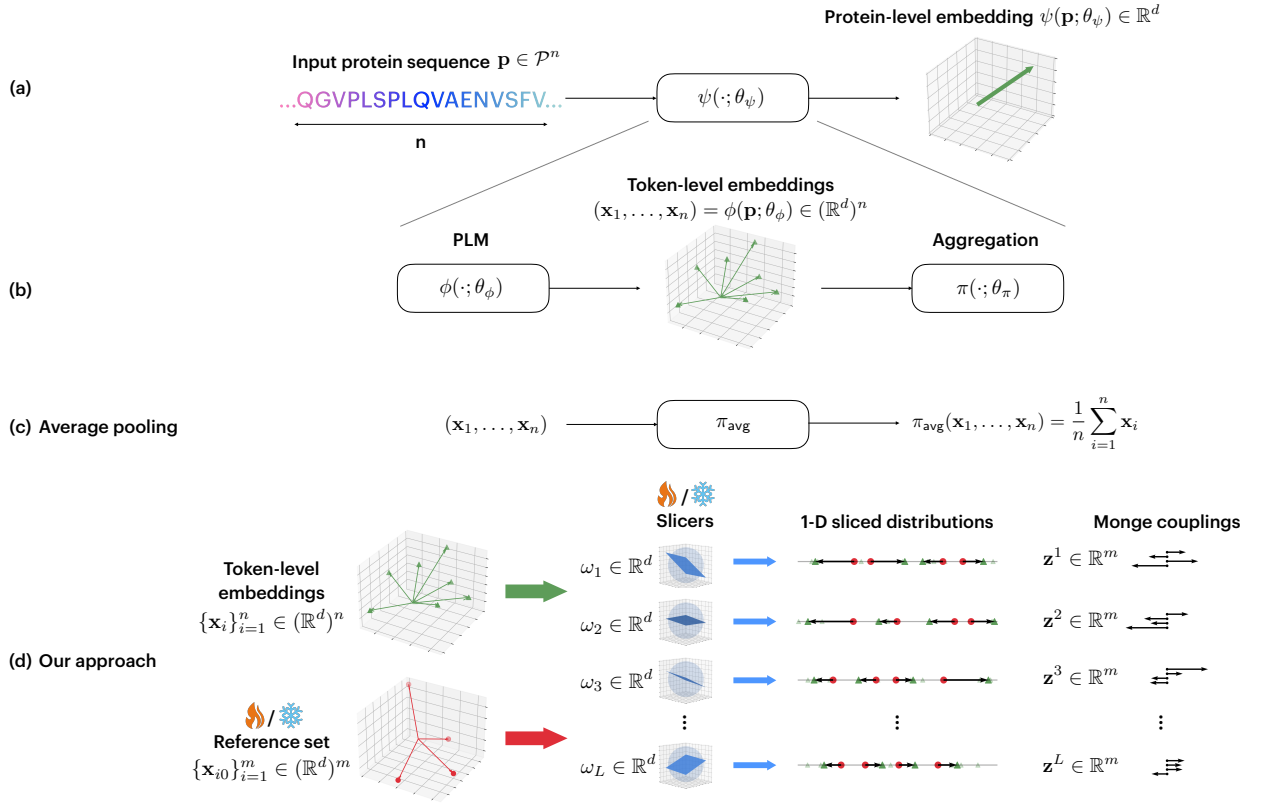


Fig. 1. Overview of the proposed method and comparison with average pooling. (a) We consider a parameterized protein representation learning function $\psi(\cdot;\theta_\psi)$, which takes as input a protein's amino acid sequence of arbitrary length and produces a fixed-length protein-level embedding at its output. (b) Breaking down the representation learning pipeline, we first pass the amino acid sequence through a pre-trained protein language model (PLM) $\phi(\cdot,\theta_\phi)$, thereby generating a set of token-level embeddings $(\mathbf{x}_1,\ldots,\mathbf{x}_n)$, each residing in $\mathbb{R}^d$. An aggregation function $\pi(\cdot,\theta_\pi)$ subsequently summarizes these embeddings into a protein-level embedding, whose size does not depend on the sequence length n. (c) Average pooling, which is most commonly used in the literature, simply takes the mean of the residue-level embeddings to derive the protein-level embedding. (d) Our proposed sliced-Wasserstein embedding aggregation module, which is based on comparing the probability distribution underlying the token-level embeddings, and a trainable reference probability distribution. Since such comparison is non-trivial in a high-dimensional space, we pass the token-level embeddings and a set of m reference elements through a set of L trainable linear slicing operations $\{\omega_l\}_{l=1}^L$, which map the embeddings and reference elements into L pairs of 1-dimensional distributions. The Monge couplings between these 1-D distributions are then calculated based on sorting and interpolation operations, and subsequently combined to generate the final fixed-length protein-level embedding. Note that in the absence of enough downstream training data, the parameters of the reference set and slicers can be left frozen, and only the final m-dimensional combination of the Monge couplings need to be learned; we refer to this variant of our proposed pooling method as "SWE_Simple."

- **Case 2** ($n_j \neq m$): The embedding procedure for this case follows similar steps as in Case 1, with the addition of an interpolation operation. Specifically, we derive the interpolated inverse cumulative distribution function (CDF) of the sliced token-level values, which we denote by F_{jl}^{-1}. This involves sorting $\{u_{ij}^l\}_{i=1}^{n_j}$, calculating the cumulative sum, and calculating the inverse through interpolation. The Monge coupling elements can then be derived as

$$z_{ij}^l = F_{jl}^{-1}\left(\frac{\rho_0^{-1}[i]}{m}\right) - u_{i0}^l, \ \forall i \in \{1,\ldots,m\}, \qquad (5)$$

where ρ_0^{-1} is defined as in Case 1. Observe that (5) reduces to (4) if $n_j = m$.

Once the Monge couplings $\{\mathbf{z}_j^l\}_{l=1}^L$ have been derived for all slices, we concatenate them to form the embedding matrix $\mathbf{Z}_j = [\mathbf{z}_j^1,\ldots,\mathbf{z}_j^L]^T \in \mathbb{R}^{L\times m}$. To reduce the dimensionality of the embedding matrix, similarly to [31], we perform a combination across the reference set elements using a learnable projection $\mathbf{w} \in \mathbb{R}^m$ to derive the final protein-level embedding

$$\pi(\mathbf{x}_{1j},\ldots,\mathbf{x}_{n_j j};\theta_\pi) = \mathbf{Z}_j\mathbf{w} \in \mathbb{R}^L. \qquad (6)$$

We set $L = d$, so that the output embedding dimensionality equals that of average pooling. Besides, to ease the optimization process, we learn the reference elements at the *output* of the slicers, training $\{u_{i0}^l\}_{i=1,l=1}^{m,L}$ directly instead of $\{\mathbf{x}_{i0}\}_{i=1}^m$.

Runtime and Memory Considerations. Note that the trainable SWE parameters include the parameters of (i) the reference set, (ii) the slicers, and (iii) the final combination across reference elements, i.e., $\theta_\pi = \left(\{u_{i0}^l\}_{i=1,l=1}^{m,L}, \{\omega_l\}_{l=1}^L, \mathbf{w}\right)$. This implies that the total number of parameters in the proposed SWE aggregation function is $mL + dL + m = \mathcal{O}((m+d)L)$. We also consider a simpler version of SWE (referred to as "SWE_Simple"), where the reference and slicer parameters are frozen at random, and only the final projection $\mathbf{w}$ is learned end-to-end, hence leading to only m learnable parameters. For the settings we consider in our numerical evaluations, as we describe next, these are negligible overheads as compared to the size of the backbone PLMs.

Evaluation Settings

We use the state-of-the-art transformer-based ESM-2 [24] and ProGen2 [33] as the PLMs $\phi(\cdot,\theta_\phi)$ that generate token-level embeddings. These models have been pre-trained via unsupervised mask-based objectives using tens of millions

of unique protein sequences and are shown to encode evolutionary patterns in protein sequences. We evaluate our SWE aggregation method when applied to the outputs of five distinct pre-trained ESM-2 models (with 8M, 35M, 150M, 650M, and 3B parameters), as well as four distinct pre-trained ProGen2 models (small, medium, base, and large).

For the SWE aggregation method, we consider 11 options for the number of reference points, $m \in \{100, 200, \ldots, 1000\}$ and two options for freezing the reference and slicer parameters (True/False), leading to 22 different SWE configurations. Note that for the "SWE_Simple" version, the reference and slicer parameters are frozen, hence the validation only happens across the 11 options for the reference set size. In all experiments, we set the number of slicers equal to the output embedding dimensionality of the corresponding PLM, i.e., $L = d$.

We evaluate our proposed pooling method across four different downstream tasks as described below. Additional results on protein-protein interaction (PPI) prediction can be found in the Appendix.

- **Drug-Target Interaction (DTI) Prediction**: In this binary classification task, the goal is to predict whether or not a given drug interacts with a target protein. As in [39], for a given drug, we first find its Morgan fingerprint, denoted by $\mathbf{f} \in \mathbb{R}^c$. Using two linear projections $\mathbf{S} \in \mathbb{R}^{D \times c}$ and $\mathbf{V} \in \mathbb{R}^{D \times d}$, we then, respectively, map the drug and the target to a shared co-embedding space $\mathbb{R}^D$, followed by an element-wise non-linearity $\sigma(\cdot)$. We then use the cosine similarity between the drug and target embeddings to calculate their interaction probability, which we then optimize using the cross-entropy loss. In particular, for a given training dataset of (drug fingerprint, target, label) triplets $\{(\mathbf{f}_j, \mathbf{p}_j, y_j)\}_{j=1}^N$, the supervised learning problem in (2) is reformulated as

$$\min_{\theta_\pi, \mathbf{S}, \mathbf{V}} \frac{1}{N} \sum_{j=1}^{N} \ell_{\mathsf{CosBCE}}\Big(\sigma\left(\mathbf{S}\mathbf{f}_j\right), \sigma\left(\mathbf{V}\psi(\mathbf{p}_j; \theta_\phi, \theta_\pi)\right), y_j\Big), \quad (7)$$

where the optimization occurs over the parameters of the SWE aggregation function, θ_π, the drug projector, $\mathbf{S}$, and the target projector, $\mathbf{V}$, while the parameters of the PLM, θ_ϕ, are kept frozen. The loss function $\ell_{\mathsf{CosBCE}} : \mathbb{R}_+^D \times \mathbb{R}_+^D \times \{0, 1\} \to \mathbb{R}_+$ in (7) is defined as

$$\ell_{\mathsf{CosBCE}}(\mathbf{x}_1, \mathbf{x}_2, y) = -\Bigg[y \log\left(\frac{\mathbf{x}_1^T \mathbf{x}_2}{\|\mathbf{x}_1\| \|\mathbf{x}_2\|}\right) + (1-y) \log\left(1 - \frac{\mathbf{x}_1^T \mathbf{x}_2}{\|\mathbf{x}_1\| \|\mathbf{x}_2\|}\right)\Bigg].$$

We use $\mathsf{ReLU}(\cdot)$ as the non-linearity $\sigma(\cdot)$ to produce both the drug and target embeddings in the non-negative orthant of the embedding space, i.e., $\mathbb{R}_+^D$. We use two datasets used by [39], namely DAVIS [10] and Binding-DB [26], to evaluate our proposed embedding mechanism in this task.

- **Out-of-Domain Drug-Target Affinity Prediction**: In this regression task, we use the Therapeutics Data Commons (TDC) Drug-Target Interaction Domain Generalization (DTI-DG) Benchmark [16], where the goal is to predict the affinity of the interaction between drugs and protein targets patented between 2019 and 2021 by training on DTI interaction affinities patented in the preceding 5-year window (i.e., 2013–2018). We use the inner product between the drug and target embeddings to predict the interaction affinity and optimize the prediction model parameters using mean squared error (MSE). Especially, we replace the loss function in (7) with $\ell_{\mathsf{MSE}} : \mathbb{R}_+^d \times \mathbb{R}_+^d \times \mathbb{R}_+ \to \mathbb{R}_+$, defined as

$$\ell_{\mathsf{MSE}}(\mathbf{x}_1, \mathbf{x}_2, y) = \left[y - \left(\mathbf{x}_1^T \mathbf{x}_2\right)\right]^2. \quad (8)$$

- **Sub-Cellular Localization (SCL)**: This is a 10-class protein property prediction task, where the goal is to predict the localization compartment of a given protein within the cell [1; 42; 23]. Given N training samples $\{(\mathbf{p}_j, y_j)\}_{j=1}^N$, we consider a multi-class classification version of the supervised learning problem (2), i.e.,

$$\min_{\theta_\pi, \mathbf{U}} \frac{1}{N} \sum_{j=1}^{N} \ell_{\mathsf{CE}}\Big(\mathbf{U}\psi(\mathbf{p}_j; \theta_\phi, \theta_\pi), y_j\Big), \quad (9)$$

where $\mathbf{U} \in \mathbb{R}^{10 \times d}$ denotes a linear classification head, and $\ell_{\mathsf{CE}} : \mathbb{R}^{10} \times \{1, \ldots, 10\} \to \mathbb{R}_+$ is defined as

$$\ell_{\mathsf{CE}}(\mathbf{x}, y) = -\sum_{i=1}^{10} \mathbb{I}(y = i) \log\left(\frac{e^{x_i}}{\sum_{k=1}^{10} e^{x_k}}\right). \quad (10)$$

- **Enzyme Commission (EC) Prediction**: We finally consider a binary classification version of the EC number prediction task [15; 55], where we predict whether a given protein is capable of catalyzing biochemical reactions. The formulation of this problem is similar to (9)-(10), where the number of classes is reduced to two, and the classification head is replaced by a two-layer perceptron with a d-dimenstional hidden layer and a LeakyReLU non-linearity.

We run our numerical experiments with five different random seeds for 50 epochs. We report the mean and standard deviation of the test performance for the SWE configuration with the highest target validation performance across the 50 training epochs. The target validation performance metric for the binary DTI, DTI-DG, SCL, and EC tasks are set to AUPR (area under the precision-recall curve), PCC (Pearson correlation coefficient), accuracy, and $F_{\max}$ (F_1-score with the optimal threshold), respectively. More details on the training settings can be found in the Appendix.

While we use average pooling as the main baseline pooling method in all our experiments, for the binary DTI prediction task, we also compare our proposed pooling method with three other baseline pooling methods, including max pooling, K-max (average pooling only across top-K elements in each dimension) [12], and light attention [42]. Our implementation code can be found at `https://github.com/navid-naderi/PLM_SWE`.

Results

Binary Drug-Target Interaction Prediction. Figure 2 shows the performance of the proposed SWE-based aggregation method for the binary DTI task, evaluated over the DAVIS and Binding-DB datasets. Our proposed SWE aggregation function and its simple variant consistently outperform average pooling in all scenarios and other pooling mechanisms in the majority of the cases. Our method especially shines when i) the PLM is smaller and has lower expressive power (e.g., 8M vs. 3B ESM-2 PLMs or small vs. large ProGen2 PLMs), and ii) there is more data to train the SWE parameters (Binding-DB vs. DAVIS). While the latter point is intuitive since SWE aggregation introduces additional trainable parameters and is, therefore, prone to overfitting, the former point is very significant from an efficiency point of view. In particular, given

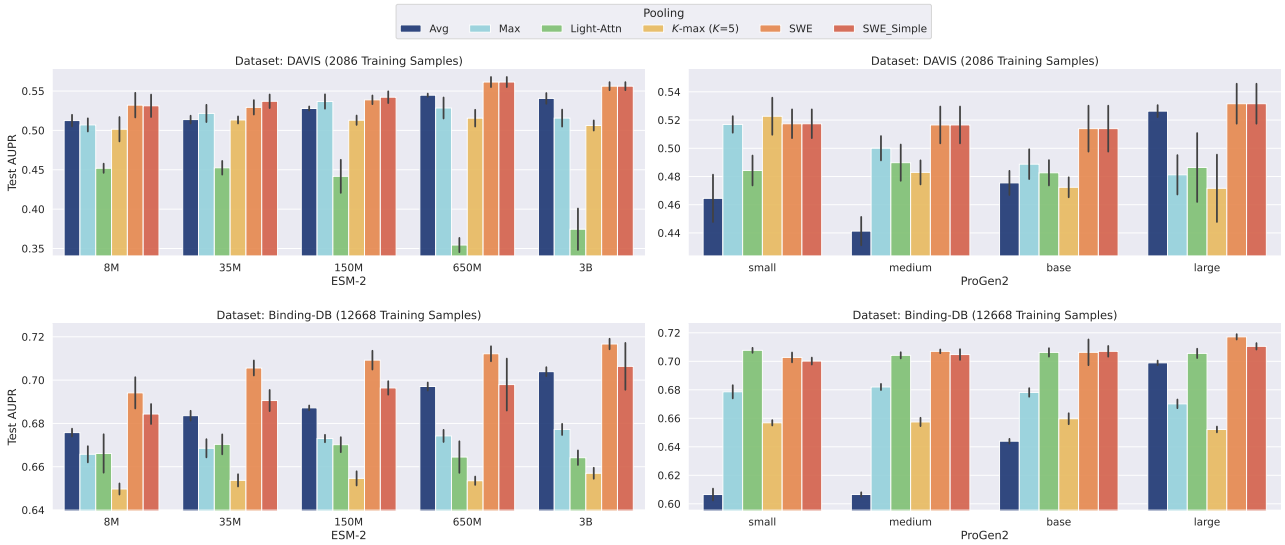


Fig. 2. Test AUPR performance of SWE and SWE_Simple as compared to baseline pooling methods for the binary drug-target interaction task on the DAVIS (top) and Binding-DB (bottom) datasets. Performance is evaluated with two families of PLMs, namely ESM-2 (left) and ProGen2 (right).

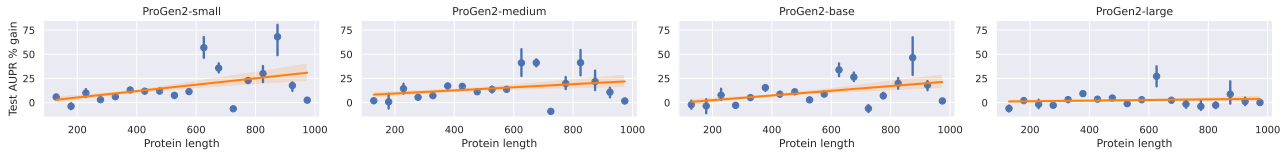


Fig. 3. Test AUPR gains of SWE over average pooling vs. the lengths of the proteins' amino acid sequences across four ProGen2 PLMs on Binding-DB.

limited computational resources, where one is constrained to using smaller PLMs, our proposed aggregation operation can, in most cases, push the performance of the smaller PLMs to similar, or even higher, levels than larger PLMs whose outputs are aggregated using averaging. Given the presence of enough training data, especially in the case of Binding-DB, our results demonstrate that there are also gains to be achieved for larger PLM backbones with billions of parameters when using SWE-based aggregation as compared to other pooling methods. However, in the absence of sufficient training data in the case of the DAVIS dataset, SWE mostly falls back to its simple version, where the reference and slicer parameters are randomly initialized and kept frozen. We provide the results on the test AUROC (area under the receiver operating characteristic curve) metric in the Appendix.

Impact of Protein Length on SWE Gains. Figure 3 illustrates how SWE's gains over average pooling in the Binding-DB DTI prediction task with ProGen2 PLMs correlate with the length of proteins' amino acid sequences. Quite interestingly, and especially with smaller PLM architectures, the gains are significantly higher for longer protein sequences. This is intuitive since drug-target interactions are typically governed by only a few amino acids, and while such information could get lost when the token-level representations generated by PLMs are simply averaged, our proposed method can extract more meaningful protein-level representations, especially for smaller PLMs and longer sequences. Similar results for the gains in test AUROC can be found in the Appendix.

Out-of-Domain Drug-Target Affinity Prediction. Figure 4-a shows the test PCC levels achieved by our proposed SWE aggregation method (and its simple version) as compared to average pooling. We observe that in this task, the SWE pooling mechanism performs better than average pooling for all the four considered ESM-2 PLMs, and interestingly, as in the DAVIS dataset, our hyperparameter grid search prefers to keep the reference and slicer parameters frozen for all PLM sizes. With a test PCC of 0.552 ± 0.006, our proposed method ranks second in the TDC DTI-DG leaderboard[1] as of this writing.

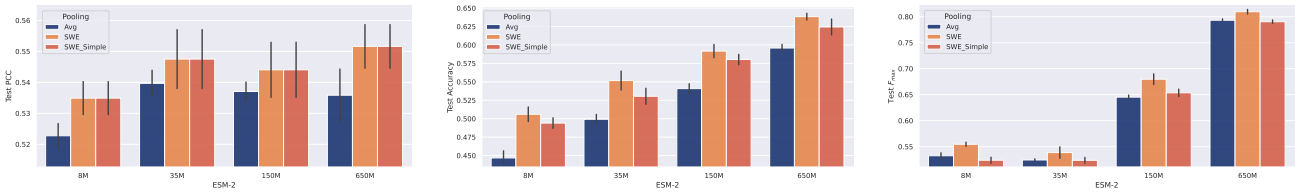
Sub-Cellular Localization. Figure 4-b demonstrates the performance comparison between SWE, SWE_Simple, and average pooling on the sub-cellular localization task. While both SWE and its simple version outperform average pooling for all four ESM-2 PLMs, we observe a performance gap between SWE and SWE_Simple, implying that training the reference and slicer parameters for this task leads to a higher accuracy than keeping them frozen.

Enzyme Commission Prediction. Figure 4-c shows the EC prediction performance of SWE and SWE_Simple compared to average pooling across four ESM-2 PLMs. While the simple version of SWE performs on par with average pooling on this task, training SWE's reference and slicer parameters results in a performance boost across all ESM-2 PLMs, providing further proof of the efficacy of the proposed pooling method.

Discussion

We introduce a novel approach for protein representation based on optimal transport principles. We anticipate that our method could have applications beyond PLMs to other foundation models in biology (e.g., for DNA sequence models). Our approach summarizes per-token embeddings in terms of their distance from a set of reference embeddings that are learned from the data. Compared to the conventional approach of

[1] `https://tdcommons.ai/benchmark/dti_dg_group/BindingDB_Patent`



(a) (b) (c)

Fig. 4. Performance comparison of SWE, SWE_Simple, and average pooling in the (a) drug-target affinity prediction, (b) sub-cellular localization, and (c) EC prediction tasks across four ESM-2 models.

average pooling, this learnability provides our approach with greater flexibility in capturing task-specific knowledge. Across a fairly diverse array of datasets, we observed that pooling works acceptably well for very large PLMs, where the greater complexity of the sequence representation compensates for the lack of learnability in the summarization layer. However, these large models have memory requirements that are beyond the capabilities of many GPUs. Sophisticated summarization approaches will be crucial in maximizing the power of the smaller models that can fit on common GPUs and could further unlock parameter-efficient fine-tuning opportunities for such pre-trained models [40; 37]. Our proposed method provides a relatively lightweight aggregation technique to improve the performance of PLMs. The additional complexity is especially negligible for the simple version of SWE, which only includes hundreds of learnable parameters, as compared to the millions-billions of parameters included in state-of-the-art PLMs.

Future work could focus on further exploring the interpretability of our method. Potential research directions include explicitly using active/binding sites in a dictionary learning-type setup, where the key residues in the interaction surface inform the selection of reference set elements, and incorporating auxiliary loss functions to encode greater biological intuition into the proposed aggregation operation. Such approaches would enhance the biological relevance of the representations produced by our method. Our work thus opens up new pathways for enhancing the interpretability and efficiency of PLM-based approaches in biology.

References

1. J. J. Almagro Armenteros, C. K. Sønderby, S. K. Sønderby, H. Nielsen, and O. Winther. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, 33(21):3387–3395, 2017.
2. Y. Bai, I. V. Medri, R. D. Martin, R. Shahroz, and S. Kolouri. Linear optimal partial transport embedding. In *International Conference on Machine Learning*, pages 1492–1520. PMLR, 2023.
3. Y. Bai, B. Schmitzer, M. Thorpe, and S. Kolouri. Sliced optimal partial transport. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13681–13690, 2023.
4. T. Bepler and B. Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 12(6):654–669.e3, 2021.
5. J. Bernett, D. B. Blumenthal, and M. List. Cracking the black box of deep sequence-based protein-protein interaction prediction. *bioRxiv*, 2023.
6. N. Brandes, G. Goldman, C. H. Wang, C. J. Ye, and V. Ntranos. Genome-wide prediction of disease variant effects with a deep protein language model. *Nature Genetics*, 55(9):1512–1522, 2023.
7. B. Chen, X. Cheng, P. Li, Y.-a. Geng, J. Gong, S. Li, Z. Bei, X. Tan, B. Wang, X. Zeng, et al. xTrimoPGLM: Unified 100b-scale pre-trained transformer for deciphering the language of protein. *arXiv preprint arXiv:2401.06199*, 2024.
8. M. Chen, C. J.-T. Ju, G. Zhou, X. Chen, T. Zhang, K.-W. Chang, C. Zaniolo, and W. Wang. Multifaceted protein–protein interaction prediction based on Siamese residual RCNN. *Bioinformatics*, 35(14):i305–i314, 2019.
9. T. Chen, P. Vure, R. Pulugurta, and P. Chatterjee. AMP-Diffusion: Integrating latent diffusion with protein language models for antimicrobial peptide generation. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.
10. M. I. Davis, J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, M. Hocker, D. K. Treiber, and P. P. Zarrinkar. Comprehensive analysis of kinase inhibitor selectivity. *Nature Biotechnology*, 29(11):1046–1051, 2011.
11. I. Deshpande, Y.-T. Hu, R. Sun, A. Pyrros, N. Siddiqui, S. Koyejo, Z. Zhao, D. Forsyth, and A. G. Schwing. Max-sliced Wasserstein distance and its use for GANs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10648–10656, 2019.
12. N. S. Detlefsen, S. Hauberg, and W. Boomsma. Learning meaningful representations of protein sequences. *Nature communications*, 13(1):1914, 2022.
13. A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, et al. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2021.
14. N. Ferruz, S. Schmidt, and B. Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13(1):4348, 2022.
15. V. Gligorijević, P. D. Renfrew, T. Kosciolek, J. K. Leman, D. Berenberg, T. Vatanen, C. Chandler, B. C. Taylor, I. M. Fisk, H. Vlamakis, et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):3168, 2021.
16. K. Huang, T. Fu, W. Gao, Y. Zhao, Y. Roohani, J. Leskovec, C. W. Coley, C. Xiao, J. Sun, and M. Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Proceedings of Neural Information Processing Systems, NeurIPS Datasets and Benchmarks*, 2021.
17. D. Iliadis, B. D. Baets, T. Pahikkala, and W. Waegeman. A comparison of embedding aggregation strategies in drug-target interaction prediction. *bioRxiv*, 2023.
18. K. Kaminski, J. Ludwiczak, K. Pawlicki, V. Alva, and S. Dunin-Horkawicz. pLM-BLAST: distant homology detection based on direct comparison of

sequence representations from protein language models. *Bioinformatics*, 39(10):btad579, 2023.
19. D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
20. S. Kolouri, N. NaderiAlizadeh, G. K. Rohde, and H. Hoffmann. Wasserstein embedding for graph learning. In *International Conference on Learning Representations*, 2021.
21. S. Kolouri, K. Nadjahi, U. Simsekli, R. Badeau, and G. Rohde. Generalized sliced Wasserstein distances. *Advances in Neural Information Processing Systems*, 32, 2019.
22. J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.
23. F.-Z. Li, A. P. Amini, Y. Yue, K. K. Yang, and A. X. Lu. Feature reuse and scaling: Understanding transfer learning with protein language models. In *Forty-first International Conference on Machine Learning*, 2024.
24. Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.
25. M. Littmann, M. Heinzinger, C. Dallago, K. Weissenow, and B. Rost. Protein embeddings and deep learning predict binding residues for various ligand classes. *Scientific Reports*, 11(1):23916, 2021.
26. T. Liu, Y. Lin, X. Wen, R. N. Jorissen, and M. K. Gilson. BindingDB: A web-accessible database of experimentally determined protein–ligand binding affinities. *Nucleic acids research*, 35(suppl_1):D198–D201, 2007.
27. I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.
28. Y. Lu, X. Liu, A. Soltoggio, and S. Kolouri. Slosh: Set locality sensitive hashing via sliced-Wasserstein embeddings. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2566–2576, 2024.
29. H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2019.
30. R. L. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *International Conference on Learning Representations*, 2019.
31. N. NaderiAlizadeh, J. F. Comer, R. Andrews, H. Hoffmann, and S. Kolouri. Pooling by sliced-Wasserstein embedding. *Advances in Neural Information Processing Systems*, 34:3389–3400, 2021.
32. N. NaderiAlizadeh, S. Kolouri, J. F. Comer, R. W. Andrews, and H. Hoffmann. Set representation learning with generalized sliced-Wasserstein embeddings. *arXiv preprint arXiv:2103.03892*, 2021.
33. E. Nijkamp, J. A. Ruffolo, E. N. Weinstein, N. Naik, and A. Madani. ProGen2: exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978, 2023.
34. C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.
35. S. Ravanbakhsh, J. Schneider, and B. Poczos. Deep learning with sets and point clouds. *arXiv preprint arXiv:1611.04500*, 2016.
36. A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, and R. Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS*, 2019.
37. R. Schmirler, M. Heinzinger, and B. Rost. Fine-tuning protein language models boosts predictions across diverse tasks. *Nature Communications*, 15(1):7407, 2024.
38. R. Singh, C. Im, Y. Qiu, B. Mackness, A. Gupta, T. Sorenson, S. Sledzieski, L. Erlach, M. Wendt, Y. F. Nanfack, et al. Learning the language of antibody hypervariability. *bioRxiv*, pages 2023–04, 2023.
39. R. Singh, S. Sledzieski, B. Bryson, L. Cowen, and B. Berger. Contrastive learning in protein language space predicts interactions between drugs and protein targets. *Proceedings of the National Academy of Sciences*, 120(24):e2220778120, 2023.
40. S. Sledzieski, M. Kshirsagar, M. Baek, R. Dodhia, J. Lavista Ferres, and B. Berger. Democratizing protein language models with parameter-efficient fine-tuning. *Proceedings of the National Academy of Sciences*, 121(26):e2405840121, 2024.
41. S. Sledzieski, R. Singh, L. Cowen, and B. Berger. D-SCRIPT translates genome to phenome with sequence-based, structure-aware, genome-scale predictions of protein-protein interactions. *Cell Systems*, 12(10):969–982.e6, 2021.
42. H. Stärk, C. Dallago, M. Heinzinger, and B. Rost. Light attention predicts protein location from the language of life. *Bioinformatics Advances*, 1(1):vbab035, 11 2021.
43. J. Su, C. Han, Y. Zhou, J. Shan, X. Zhou, and F. Yuan. SaProt: Protein language modeling with structure-aware vocabulary. In *The Twelfth International Conference on Learning Representations*, 2024.
44. P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018.
45. T. Tu, G. Krishna, and A. Aghazadeh. ProtiGeno: a prokaryotic short gene finder using protein language models. *arXiv preprint arXiv:2307.10343*, 2023.
46. S. Unsal, H. Atas, M. Albayrak, K. Turhan, A. C. Acar, and T. Doğan. Learning functional properties of proteins with language models. *Nature Machine Intelligence*, 4(3):227–245, 2022.
47. L. Valeriani, F. Cuturello, A. Ansuini, and A. Cazzaniga. The geometry of hidden representations of protein language models. *bioRxiv*, pages 2022–10, 2022.
48. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
49. C. Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.
50. A. Villegas-Morcillo, A. M. Gomez, and V. Sanchez. An analysis of protein language model embeddings for fold prediction. *Briefings in Bioinformatics*, 23(3):bbac142, 04 2022.
51. E. Wagstaff, F. Fuchs, M. Engelcke, I. Posner, and M. A. Osborne. On the limitations of representing functions on

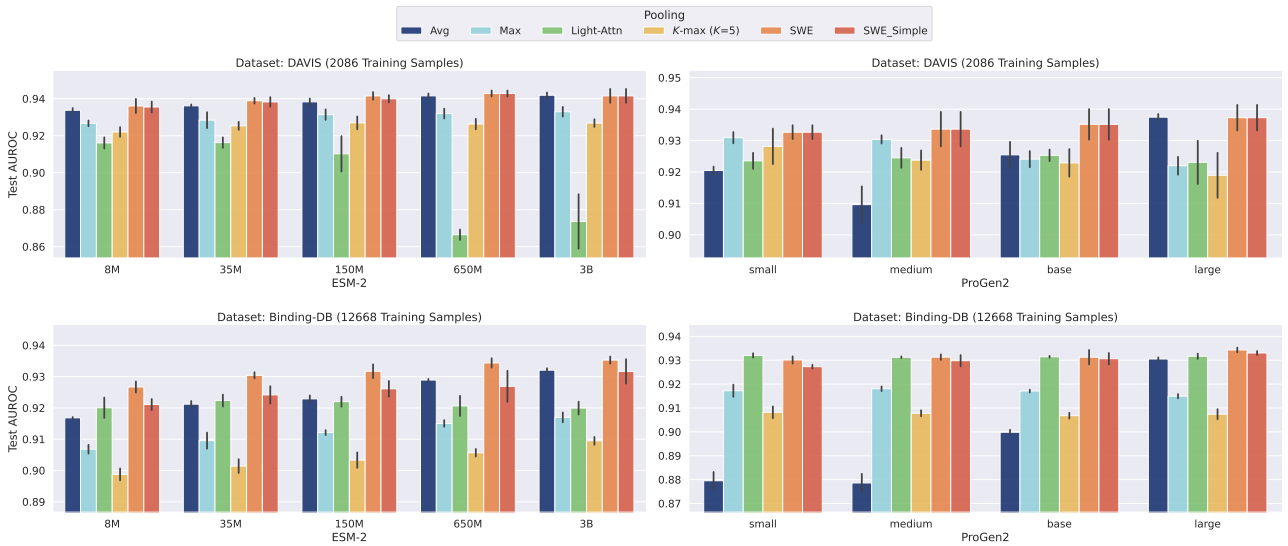


Fig. 5. Test AUROC performance of SWE and SWE_Simple as compared to baseline pooling methods for the binary drug-target interaction task on the DAVIS (top) and Binding-DB (bottom) datasets. Performance is evaluated with two families of PLMs, namely ESM-2 (left) and ProGen2 (right).

sets. In *International Conference on Machine Learning*, pages 6487–6494. PMLR, 2019.

52. X. Wang, Z. Zheng, F. Ye, D. Xue, S. Huang, and Q. Gu. Diffusion language models are versatile protein learners. In *International Conference on Machine Learning*, 2024.
53. F. Wu and S. Z. Li. A hierarchical training paradigm for antibody structure-sequence co-design. *arXiv preprint arXiv:2311.16126*, 2023.
54. M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.
55. Z. Zhang, J. Lu, V. Chenthamarakshan, A. Lozano, P. Das, and J. Tang. Structure-informed protein language model. In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*.

Appendix

Training Settings and Hyperparameters

For the DTI and SCL tasks, we train the SWE aggregation parameters using the AdamW optimizer [27] with a learning rate of 10^{-4} and cosine annealing schedule with a restart duration of 10 epochs. For the EC task, we train the SWE aggregation parameters using the Adam optimizer [19] with a learning rate of 10^{-3}, reducing it by a factor of 0.6 if the performance does not improve after 5 epochs. The batch size is set to 32 for the DTI and SCL tasks and 8 for the EC task.

Binary DTI Prediction AUROC Results

Figure 5 shows the AUROC comparison between the proposed pooling method and the baselines for the DTI prediction task on DAVIS and Binding-DB for ESM-2 and ProGen2 models. The overall trends and gains of SWE with respect to baseline pooling operations are consistent with our AUPR results in Figure 2.

Moreover, Figure 6 shows the AUROC gains of SWE over average pooling across different protein lengths. Similar to the AUPR gains in Figure 3, SWE's gains over average pooling increase with the sequence length and decrease with the PLM size.

Protein-Protein Interaction (PPI) Prediction

In this task, we focus on predicting whether or not two given proteins will interact with each other. Specifically, we embed each protein's amino acid sequence using the same PLM and aggregation pipeline in parallel, and then we leverage the cosine similarity between the protein-level representations to estimate their interaction probability. For a given PPI training dataset consisting of N (protein, protein, label) triplets $\{(\overline{\mathbf{p}_j}, \underline{\mathbf{p}_j}, y_j)\}_{j=1}^N$, this task seeks to solve the following supervised learning problem:

$$\min_{\theta_\pi \in \Theta_\pi, \mathbf{V} \in \mathbb{R}^{D\times d}} \frac{1}{N} \sum_{j=1}^{N} \ell_{\mathsf{CosBCE}} \Big(\sigma\left(\mathbf{V}\psi(\overline{\mathbf{p}_j}; \theta_\phi, \theta_\pi)\right), \sigma\left(\mathbf{V}\psi(\underline{\mathbf{p}_j}; \theta_\phi, \theta_\pi)\right), y_j \Big).$$

We use the "gold standard" dataset provided in [5] to evaluate our proposed SWE aggregation mechanism in this task, which comprises balanced PPI data without any leakage between training, validation, and testing samples. For this specific task, we consider four options for the number of slices, $L \in \{128, 256, 512, 1024\}$, as well as four options for the number of reference points, $m \in \{128, 256, 512, 1024\}$, leading to 16 different SWE configurations. While the default batch size is 32, for experiments with $L = 1024$ slices, we use a reduced batch size of 24 due to computational limitations. We report the mean and standard deviation of the test/validation performance for the SWE configuration (i.e., (L, m) pair) with the highest validation AUPR across the 50 training epochs.

Table 1 compares the validation and test performance of our proposed SWE aggregation method with average pooling across the considered ESM-2 PLMs. As the table shows, SWE generally outperforms average pooling in terms of F1-score and recall, while performing on par with average pooling in terms of accuracy, and underperforming for the other metrics, including precision and specificity. These results are consistent with the ones reported by [40], where more expressive models and

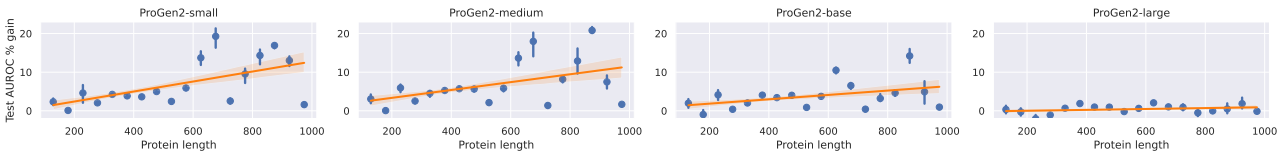


Fig. 6. Test AUROC gains of SWE over average pooling vs. the lengths of the proteins' amino acid sequences across four ProGen2 PLMs on Binding-DB.

	ESM-2 PLM	Aggregation	Accuracy	F1	MCC	AUPR	Precision	Recall	Specificity
Validation	8M	SWE	0.607 ± 0.003	0.626 ± 0.008	0.216 ± 0.006	0.652 ± 0.004	0.619 ± 0.006	0.671 ± 0.023	0.701 ± 0.013
		Avg	0.606 ± 0.002	0.614 ± 0.004	0.216 ± 0.004	0.653 ± 0.002	0.637 ± 0.004	0.643 ± 0.012	0.731 ± 0.008
	35M	SWE	0.617 ± 0.002	0.632 ± 0.009	0.235 ± 0.004	0.669 ± 0.003	0.627 ± 0.002	0.693 ± 0.024	0.690 ± 0.009
		Avg	0.617 ± 0.001	0.628 ± 0.008	0.237 ± 0.002	0.667 ± 0.001	0.648 ± 0.001	0.666 ± 0.024	0.735 ± 0.005
	150M	SWE	0.623 ± 0.002	0.633 ± 0.006	0.248 ± 0.003	0.677 ± 0.003	0.633 ± 0.005	0.663 ± 0.021	0.681 ± 0.011
		Avg	0.624 ± 0.002	0.635 ± 0.006	0.251 ± 0.004	0.679 ± 0.003	**0.661 ± 0.004**	0.676 ± 0.023	**0.754 ± 0.008**
	650M	SWE	**0.629 ± 0.002**	**0.668 ± 0.001**	**0.259 ± 0.004**	**0.685 ± 0.002**	0.630 ± 0.008	**0.810 ± 0.004**	0.643 ± 0.023
		Avg	0.626 ± 0.001	0.650 ± 0.004	0.251 ± 0.002	0.684 ± 0.002	0.651 ± 0.003	0.732 ± 0.030	0.732 ± 0.008
Test	8M	SWE	0.637 ± 0.004	0.638 ± 0.013	0.274 ± 0.008	0.688 ± 0.005	0.636 ± 0.007	0.642 ± 0.030	0.632 ± 0.027
		Avg	0.639 ± 0.000	0.637 ± 0.005	0.279 ± 0.001	0.690 ± 0.000	0.641 ± 0.004	0.633 ± 0.013	**0.645 ± 0.013**
	35M	SWE	0.646 ± 0.003	0.661 ± 0.010	0.294 ± 0.004	0.696 ± 0.005	0.635 ± 0.011	0.690 ± 0.035	0.602 ± 0.040
		Avg	0.648 ± 0.000	0.651 ± 0.010	0.297 ± 0.001	0.699 ± 0.000	0.646 ± 0.008	0.658 ± 0.029	0.639 ± 0.028
	150M	SWE	0.649 ± 0.002	0.664 ± 0.007	0.299 ± 0.005	0.703 ± 0.002	0.636 ± 0.005	0.696 ± 0.020	0.601 ± 0.020
		Avg	0.651 ± 0.001	0.654 ± 0.009	0.303 ± 0.003	0.708 ± 0.002	**0.649 ± 0.006**	0.659 ± 0.025	0.643 ± 0.023
	650M	SWE	0.650 ± 0.005	**0.683 ± 0.007**	0.307 ± 0.007	0.705 ± 0.005	0.625 ± 0.012	**0.754 ± 0.033**	0.545 ± 0.042
		Avg	**0.657 ± 0.002**	0.670 ± 0.006	**0.315 ± 0.003**	**0.717 ± 0.001**	0.646 ± 0.007	0.696 ± 0.022	0.618 ± 0.024

Table 1. PPI validation and test results on the "gold standard" dataset in [5] for the SWE and average pooling methods across four different ESM-2 PLM backbones. Following [40], for a comprehensive evaluation, we report the performance using seven different metrics, including accuracy, F1-score, Matthews correlation coefficient (MCC), AUPR, precision, recall, and specificity, with mean and standard deviation across five different random seeds. Underlined numbers indicate the best aggregation performer in each metric for each phase (validation/test) and each PLM. Bold numbers indicate the best performer in each metric for each phase (validation/test).

heavier fine-tuning lead to superior F1 and recall levels, while lowering the other metrics. Further research, including using optimal partial transport- (OPT-)based embedding methods [2; 3], is required to strike the right balance among the different metrics through which PPI prediction quality is measured.

Interpretability of the Learned SWE Representations

One of the desirable properties of the proposed embeddings is that the sliced-Wasserstein distance of the distributions underlying the token-level embeddings of two given protein sequences can be approximated by the average distance of their Monge couplings to the reference across different slices [31]. In particular, for two proteins $\overline{\mathbf{p}}$ and $\underline{\mathbf{p}}$ with Monge coupling matrices $\overline{\mathbf{Z}} = [\overline{\mathbf{z}}^1, \dots, \overline{\mathbf{z}}^L]^T \in \mathbb{R}^{L \times m}$ and $\underline{\mathbf{Z}} = [\underline{\mathbf{z}}^1, \dots, \underline{\mathbf{z}}^L]^T \in \mathbb{R}^{L \times m}$, respectively, we can approximate their pairwise sliced-Wasserstein distance as

$$\mathcal{SW}_2(\overline{\mathbf{p}}, \underline{\mathbf{p}}) \approx \left(\frac{1}{L} \sum_{l=1}^{L} \left\| \overline{\mathbf{z}}^l - \underline{\mathbf{z}}^l \right\|_2^2 \right)^{\frac{1}{2}}, \tag{11}$$

where, with a slight abuse of notation, we use $\mathcal{SW}_2(\overline{\mathbf{p}}, \underline{\mathbf{p}})$ to denote the sliced-Wasserstein distance between token-level representations of $\overline{\mathbf{p}}$ and $\underline{\mathbf{p}}$ at the output of the PLM backbone.

The approximation in (11) allows us to visualize the pairwise distance of interacting and non-interacting proteins in the embedding space. Figure 7 shows the distributions of (approximate) SW distances between proteins in the training dataset separated by whether or not they interact, where the embeddings are generated by the 650M-parameter ESM-2 backbone and a trained SWE aggregation module with $m = 1024$ reference points and $L = 1024$ slices. As the figure demonstrates, there is a separation between the two histograms, with interacting proteins landing closer to each other in the embedding space from a sliced-Wasserstein distance point of view as compared to non-interacting pairs of proteins.

Fig. 7. Histograms of sliced-Wasserstein distance between embeddings of interacting and non-interacting proteins at the output space of a pre-trained ESM-2 PLM with 650M parameters. The SW distance is calculated using a trained SWE pooling module with $m = 1024$ reference points and $L = 1024$ slices.