Dense Representation of the Functional Protein Space with Wasserstein Autoencoders

Nicolas Bélanger NICOLAS.BELANGER@UMONTREAL.CA
Université de Montréal

Basile Dura BASILE.DURA@UMONTREAL.CA
Université de Montréal

Julien Horwood JULIEN.HORWOOD@UMONTREAL.CA
Université de Montréal

Abstract

Proteins play a major role in the way living organisms develop, and their sequencing is a very active field of research. However, much like most sequences of letters do not form a word, an overwhelmingly large proportion of proteins do not have a particular function. In this work, we aim to find a dense low-dimensional representation of the otherwise sparse space of functional proteins by leveraging the Wasserstein Auto-Encoder framework.

1. Introduction

Proteins play a major role in the way living organisms develop. Therefore, their sequencing is a very active field of research, as biologists aim at determining the function of as many protein chains as possible.

However, the process of finding and testing for functional proteins is extremely tedious since the number of possible sequences blows up exponentially with their length. Moreover, the space of *functional* proteins is sparse: much like most sequences of letters does not make an actual word, an overwhelmingly large proportion of these proteins do not have a particular function.

In this work, we aim to find a dense low-dimensional representation of the otherwise sparse space of functional proteins by leveraging the Wasserstein Auto-Encoder framework. This framework differs from previous autoencoder methods by its optimization objective, derived from the optimal transport problem. In particular, this objective decouples into a reconstruction cost and a regularization cost which looks to push the marginal encoding distribution towards a specified prior. By regularizing over the marginal distribution, the framework hopes to achieve smoother embeddings over the latent space. We can then use such a framework to better capture the latent distribution of our data. The motivation for the application of this project is that capturing the underlying distribution of functional proteins would enable biologists to look for new protein sequences in a more efficient manner, using the model as a guiding tool proposing good candidates for exploration. Such a density based analysis may also be helpful in studying the effects of sequence mutations on function preservation, which is another very active area of research.

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

2. Related Work

We relied on previous work done by researchers at Harvard's Mark Laboratory (Sinai et al., 2017) wherein the authors devise a way to leverage the variational auto-encoder framework to obtain a meaningful representation of the functional protein space. In particular, we work with the dataset used in this paper.

In essence, our work aims at applying the framework of the Wasserstein Auto-encoder described in the eponymous 2017 paper (Tolstikhin et al., 2017) to the task of protein sequence exploration, by creating a dense embedding of functional protein sequences.

3. Wasserstein Auto-Encoders

The initial phase of this project consists in understanding the Wasserstein auto-encoder (WAE) model itself. Thus we present in this section the theoretical aspects of the Wasserstein auto-encoder framework and algorithm. Let $\mathcal{X}$ be the data space (e.g. protein sequence space) and let

P_X be the true probability distribution of elements $x \in \mathcal{X}$. The WAE algorithm aims at building a generative model G having a probability distribution $P_G \approx P_X$. One of the main ideas of the WAE framework is to rely on *optimal transport theory* to measure the distance between the probability distributions P_X and P_G. Given a measurable cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, the optimal transport problem $W_c(P_X, P_G)$ is defined as finding a joint distribution P which solves the optimization problem

$$\inf_{P \in \mathcal{P}(X,\tilde{X}) | X \sim P_X, \tilde{X} \sim P_G} \mathbb{E}_{(X,\tilde{X}) \sim P} \left[c(X, \tilde{X}) \right]. \quad (1)$$

This formulation directly comes from Kantorovich's formulation of the optimal transport problem, when any joint probability distribution $P(X \sim P_X, \tilde{X} \sim P_G)$ is being viewed as a measure in $\mathcal{X}$.

3.1. Objective reformulation

Relying on an auto-encoder architecture for which we define

- $\mathcal{Z}$ as the latent space,
- P_Z a prior distribution over this latent space
- $Q(Z|X)$ an encoding distribution $\mathcal{X} \to \mathcal{Z}$, and
- $G : \mathcal{Z} \to \mathcal{X}$ a decoder mapping,

one can show that, assuming the decoder G is deterministic, the optimal transport problem (1) can be reformulated as

Theorem 1. *Under the conditions described above,*

$$W_c(P_X, P_G) = \inf_{Q(Z|X)} \mathbb{E}_{P_X} \left[\mathbb{E}_{Q(Z|X)} [c(X, G(Z))] \right]$$
$$\text{such that} \quad Q_Z = P_Z$$

This corresponds to the theorem 1 in Tolstikhin et al. (2017), first introduced in Bousquet et al. (2017). The main advantage of the reformulation is that we can optimize over the encoders instead of arbitrary joint distributions.

Elements of proof for theorem 1. The main idea behind the proof is to show that in the case of a deterministic decoder G, we get $\mathcal{P}(P_X, P_G) = \mathcal{P}_{X,\tilde{X}}$ where:

- $\mathcal{P}(P_X, P_G)$ is the set of of joint distribution of $(X, \tilde{X})$ with marginals P_X and P_G.
- $\mathcal{P}_{X,\tilde{X}}$ (and $\mathcal{P}_{X,Z}$) denotes the set of marginals on $(X, \tilde{X})$ (and (X, Z)), induced by distributions on $(X, \tilde{X}, Z)$ in $\mathcal{P}_{X,Z,\tilde{X}}$, such that $X \sim P_X$, $\tilde{X} \sim P_G$ and $X \perp \tilde{X} | Z$.

The decoder being deterministic, we can establish that $(X \perp \tilde{X})|Z$, which leads to $\mathcal{P}(P_X, P_G) = \mathcal{P}_{X,\tilde{X}}$.

Using the tower rule in the expression, we get under these conditions:

$$\begin{aligned} W_c(P_X, P_G) &= \inf_{P \in \mathcal{P}_{X,Z,\tilde{X}}} \mathbb{E}_{(X,Z,\tilde{X}) \sim P} [c(X, \tilde{X})] \\ &= \inf_{P \in \mathcal{P}_{X,Z,\tilde{X}}} \mathbb{E}_{P_X} \mathbb{E}_{X \sim P(X|Z)} [c(X, G(Z))] \\ &= \inf_{P \in \mathcal{P}_{X,Z}} \mathbb{E}_{(X,Z) \sim P} [c(X, G(Z))] \end{aligned}$$

Noticing that $\mathcal{P}_{X,Z} = P(P_X, P_Z)$ leads to the conclusion.

□

3.2. Constraint relaxation

In order to get an unconstrained optimization problem, the constraint $Q_Z = P_Z$ is relaxed by applying a penalty λ over a given divergence function $\mathcal{D}_Z(Q_Z, P_Z)$. The WAE objective to minimize then becomes

$$\inf_{Q(Z|X)} \mathbb{E}_{P_X} \mathbb{E}_{Q(Z|X)} [c(X, G(Z))] + \lambda \mathcal{D}_Z(Q_Z, P_Z) \quad (2)$$

This allows for a trade-off between the reconstruction cost and the regularization factor (first and second term in (2), respectively).

The WAE framework is in many ways very similar to the VAE framework, especially because they both exploit an auto-encoder architecture with a latent distribution that is pushed towards a prior. The notable difference is the penalization of the marginal rather than the conditional distribution of the encoder.

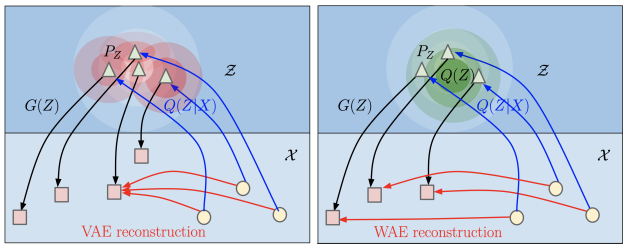
In the VAE, instead of searching for an optimal encoding distribution $Q(Z|X)$ having a marginal distribution Q(Z) close to a prior $P(Z)$ as in the WAE, we try to find $Q(Z|X)$ that is close to a prior $P(Z)$ **for each** $x \in \mathcal{X}$ of the training set. Figure 1, which is taken from the paper of Tolstikhin et al. (2017), illustrates this difference between the VAE and the WAE frameworks.

This particularity of the WAE framework tends to build denser and smoother clusters of embeddings in the latent space. Indeed, because of its point-wise regularizer, the VAE tends to build more multimodal probability mass functions than the ones found through the WAE algorithm.

3.3. The WAE Algorithm

We now describe in more detail the implementation of the WAE algorithm based on the theoretical developments presented above.

By parameterizing the encoder and decoder distributions through deep neural networks, the authors (Tolstikhin et al.,



(a) VAE reconstruction (b) WAE reconstruction

Figure 1. Illustrations from Tolstikhin et al. (2017)

2017) seem to make the implicit hypothesis that the optimal solution space of (2) can be reasonably approximated through this parametric space of neural networks.

Indeed, the theorem (1) from this paper was proved only for the case where the infimum is taken over any distribution $Q(Z|X)$ (parametric or not). Hence, there are no guarantees that there exists a parametric distribution $Q(Z|X)$ leading to the same objective value as the original optimal transport problem. Exploring this assumption further would thus be an interesting direction for future work. However, empirical evidence suggests this assumption to be reasonable, given the performance of the model.

In Tolstikhin et al. (2017), two types of divergence $\mathcal{D}_Z$ measures are experimented with: a GAN-based penalty and a maximum mean discrepancy (MMD) penalty. In the scope of this project, we decided to implement only the algorithm using the MMD-based divergence measure, notably because it allows for a direct backpropagation through automatic differentiation optimization. This measure depends on a hyper-parameter kernel function $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$, which to be rigorous is a positive-definite kernel which induces some Reproducing Kernel Hilbert Space (RKHS). Given this kernel, the MMD measure is defined as

$$\mathrm{MMD}_k(P_z, Q_z) = \left\| \int_{\mathcal{Z}} k(z, .) dP_z - \int_{\mathcal{Z}} k(z, .) dQ_z \right\|$$

where the norm is computed according to the induced RKHS.

Though we did not deeply explore the theory of RKHS, this intuitively attempts to quantify the overall distance between the latent embeddings under the encoder and the prior distribution, mapped under the pre-defined kernel over the probability space of both distributions.

In order to apply this discrepancy in practice, we use an unbiased estimator for the MMD measure so that it can be evaluated empirically. One can show that the following estimator is unbiased for $\mathrm{MMD}_k(P_Z, Q_Z)$:

$$\frac{\lambda}{n(n-1)} \sum_{\ell \neq j} k(z_\ell, z_j) + \frac{\lambda}{n(n-1)} \sum_{\ell \neq j} k(\tilde{z}_\ell, \tilde{z}_j) - \frac{2\lambda}{n^2} \sum_{\ell, j} k(z_\ell, \tilde{z}_j),$$

where z_j are sampled from the prior distribution P_Z, and the $\tilde{z}_j$ are sampled from the encoder distribution $Q(Z|X)$, for some $X = x_j$. Also, given a decoder G_θ, the transport cost between x and $G_\theta(z)$ can be empirically estimated by $\frac{1}{n} \sum_{i=1}^{n} c\big(x_i, G_\theta(\tilde{z}_i)\big)$.

Hence, we can define the main steps of the WAE algorithm with MMD penalty, which are provided in algorithm 1.

Algorithm 1 WAE with MMD-based penalty

Require:

- a regularization coefficient $\lambda > 0$ for the penalty on $\mathcal{D}_Z(Q_Z, P_Z)$
- a kernel function $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ for MMD

while (ϕ, θ) not converged **do**
sample $\{x_1, \ldots, x_n\}$ from the training set
sample $\{z_1, \ldots, z_n\}$ from the prior P_Z
sample $\tilde{z}_i$ from the current encoder $Q_\phi(Z|x_i)$ for $i = 1, \ldots, n$
update Q_ϕ and G_θ by descending with SGD:

$$\frac{1}{n} \sum_{i=1}^{n} c\big(x_i, G_\theta(\tilde{z}_i)\big) + \frac{\lambda}{n(n-1)} \sum_{\ell \neq j} k(z_\ell, z_j) + \frac{\lambda}{n(n-1)} \sum_{\ell \neq j} k(\tilde{z}_\ell, \tilde{z}_j) - \frac{2\lambda}{n^2} \sum_{\ell, j} k(z_\ell, \tilde{z}_j)$$

end while

3.4. Implementation

In order to apply the WAE model to our project, we wrote our own implementation in Pytorch, since the implementations provided by (Tolstikhin et al., 2017) was only in Tensorflow (a framework with which we are less at ease). Our implementation can be found on Github.

We also chose to implement our version of the VAE framework to allow us to make our own comparisons between the two algorithms, and try to reproduce the results obtained by (Sinai et al., 2017).

Implementation using the MMD-based divergence was relatively straightforward, however we expect implementation and tuning using the GAN divergence would be more involved. For our implementation, some tuning over the size

of the latent space and gradual decreasing of the learning rate was sufficient for the model to converge.

3.5. Example on MNIST

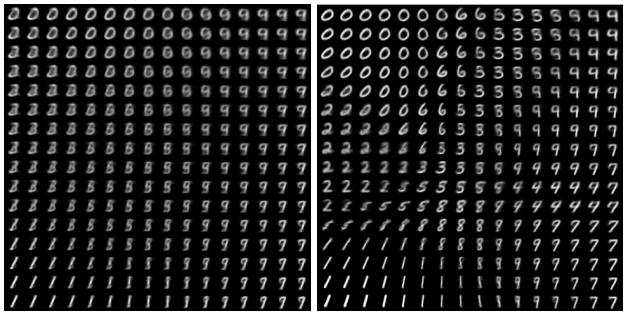


Figure 2. View of the latent space representation at the beginning and end of training on the MNIST dataset

Figure 2 shows a representation of the two-dimensional latent space. In order to get this representation, we sample regularly spaced examples from the $[-1, 1]^2$ square in the two-dimensional latent space, and present their reconstruction obtained by the decoding network.

4. Experiment

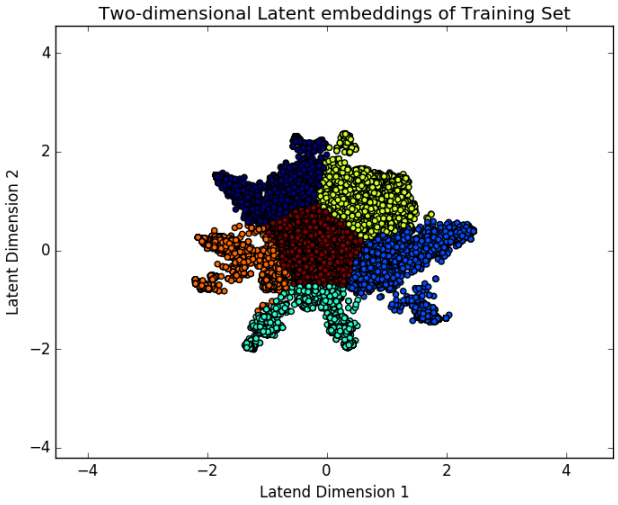
4.1. Protein Space Embedding

Biological research on protein functionality is a very costly process, notably due to the exponentially large space of protein sequences and involved testing process. The general research hypothesis is that sequences with minor mutations to functional proteins are more likely to encode similar biological functions. Thus it may be interesting to identify dense areas of functional proteins within the sequence space. We attempt to identify such high density functional protein regions by embedding known functional proteins into a latent space using WAEs. The ability to sample new sequences from these high-density areas of the latent space may then be of high value for guiding new research.

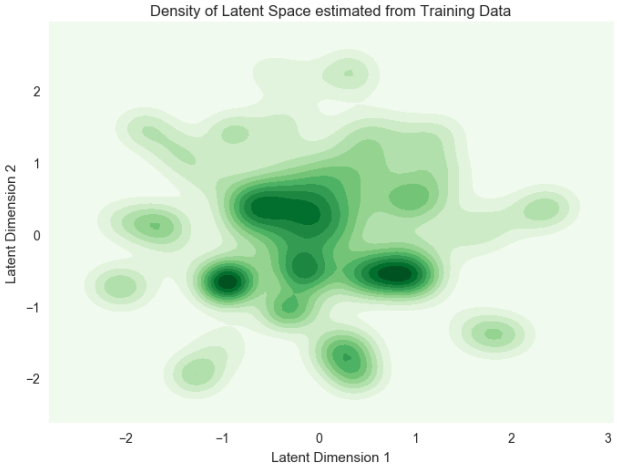
Figure 3 provides a visualization of the protein sequence training data when encoded through a two-dimensional latent space. Here we can see the clustering of the data towards the standard Gaussian distribution, which was used as a prior, and the branching shape obtained from the WAE algorithm in an attempt to separate distinct clusters of data. A kernel density estimation was then performed on this data to provide further insight into the the distribution over the space.

4.2. Performance

Once the model is trained, we can look at its performance on the protein set. There are two types of performance that



(a) Scatter plot



(b) Density Estimation

Figure 3. Visualization of the latent space for protein embedding

interest us:

- The *reconstruction performance*: we would like to know whether the latent space is informative enough to reproduce a given example.
- The *performance as a "guiding tool"*. It quantifies how well the model helps identifying proteins that are likely to be functional.

4.2.1. Reconstruction Performance

The reconstruction performance is quite simple to measure. We can first simply observe the reconstruction loss to quantify how our model is learning, or look at the accuracy of reconstruction (the number of amino acids in a sequence that were accurately produced).

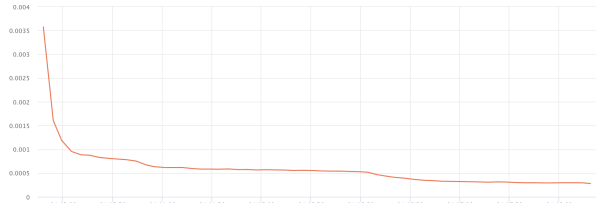


Figure 4. Testing loss of the WAE model as a function of training time

In figure 4, we see that the model has a learning behaviour consistent with expectations. We use a cross-entropy loss which is averaged over the entire flattened sequence of amino-acids.

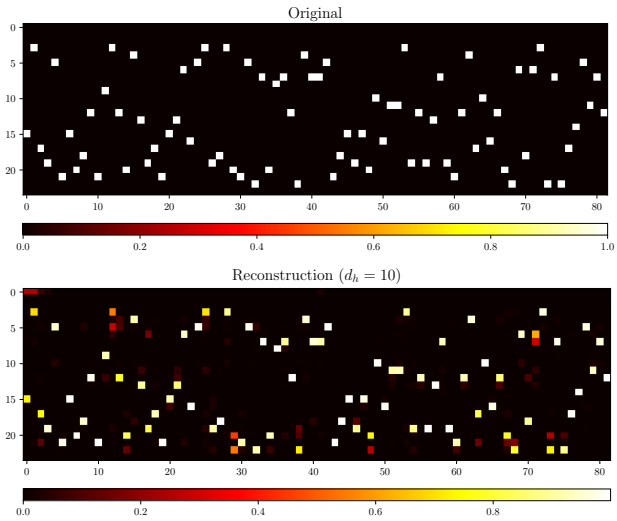


Figure 5. Reconstruction of the sequence by the WAE

In figure 5, we show the one-hot encoding of a given sequence, along with its reconstruction as output by our network. The one-hot encoding is performed over the amino-acid space of length 24, at every position in the protein sequence. Thus the figure represents a probability distribution over the predicted amino-acids of the reconstruction, at each position in the sequence.

With a simple dense-layer architecture, we obtain a 68% accuracy on average, where we take the argmax over the amino-acid axis probabilities to predict the sequence reconstruction.

4.2.2. Performance as a guiding tool

In order to test the performance of our model for protein exploration, we follow the approach by Sinai et al. (2017) and compare the log-probability[1] of known sequences (output by the network) with their fitness. The *fitness* is an experimentally measured quantity that assesses the "functionality" of a sequence. The intuitive idea of this measure is that we want high probability to correlate well with high fitness.

The log-probability of a sequence $x = (x_i)_{i=1}^n$ is given by:

$$l(x) = \sum_{i=1}^{n} \log p(\tilde{x}_i = x_i \mid x) \tag{3}$$

Where $p(\tilde{x}_i = x_i \mid x)$ is the probability for the i$^{\text{th}}$ amino-acid composing $\tilde{x}$ to be x_i, as given by the last layer of the network.

Table 1. Spearman Correlation between the *fitness* and the log-probability

Model	Correlation
VAE	0.271
WAE	0.311

Table 1 shows the results obtained using the same architecture with both the VAE and WAE frameworks. Although we did not manage to achieve the same correlation as Sinai et al. (who achieved correlations as high as 60%), note that the Wasserstein autoencoder outperforms the VAE framework.

5. Conclusion and Future Directions

In this work, we have explored the usefulness of Wasserstein Auto-Encoders as generative models. In particular, we have demonstrated their functionality by interpolating between the digit embeddings of MNIST. Furthermore, we inspired ourselves of the work of Sinai et al. (Sinai et al., 2017) on learning dense representations of the protein space by applying WAEs to this problem.

We identify here some directions that would be worth exploring in future work:

- Although using parametric encoders yields good experimental results, there are no theoretical guarantees that we are in fact minimizing the distance described in equation (1). Working on quantifying the discrepancy could prove extremely useful.
- In the future, a stronger collaboration between computer scientists and biologists would benefit the field of computational biology. During our project, we have had some trouble reaching a deep understanding of some definitions and concepts regularly used by biologists. Endeavors such as the Marks Laboratory at Harvard (Sinai et al., 2017) help translate the great advances in machine learning into powerful discoveries in biology.

[1] Since the last layer of our model is a softmax along the amino-acid axis, it can be interpreted as a probability.

References

Bousquet, Olivier, Gelly, Sylvain, Tolstikhin, Ilya, Simon-Gabriel, Carl-Johann, and Schoelkopf, Bernhard. From optimal transport to generative modeling: the VEGAN cookbook. *arXiv preprint arXiv:1705.07642*, 2017.

Sinai, Sam, Kelsic, Eric, Church, George M, and Nowak, Martin A. Variational auto-encoding of protein sequences. *arXiv preprint arXiv:1712.03346*, 2017.

Tolstikhin, Ilya, Bousquet, Olivier, Gelly, Sylvain, and Schoelkopf, Bernhard. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.