OXFORD UNIVERSITY PRESS | Bioinformatics

Deep convolutional networks for quality assessment of protein folds

Journal:	*Bioinformatics*
Manuscript ID	BIOINF-2017-1809
Category:	Original Paper
Date Submitted by the Author:	05-Oct-2017
Complete List of Authors:	Derevyanko, Georgy; Concordia University, Chemistry and Biochemistry Grudinin, Sergei; INRIA Grenoble – Rhone-Alpes Research Center, NANO-D; CNRS, Laboratoire Jean Kuntzmann Bengio, Yoshua; Université de Montréal, Department of Computer Science and Operations Research Lamoureux, Guillaume; Concordia University, Chemistry and Biochemistry
Keywords:	Machine learning, Molecular modeling, Neural networks, Protein structure prediction, Structural bioinformatics

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Original Paper

Structural bioinformatics

Deep convolutional networks for quality assessment of protein folds

Georgy Derevyanko [1,*], Sergei Grudinin [2], Yoshua Bengio [3,†] and Guillaume Lamoureux [1,*]

[1]Department of Chemistry and Biochemistry and Centre for Research in Molecular Modeling (CERMM), Concordia University, Montréal, H4B 1R6, Canada, [2]Inria, Université Grenoble Alpes, CNRS, Grenoble INP, LJK, Grenoble, 38000, France and [3]Department of Computer Science and Operations Research, Université de Montréal, Montréal, H3C 3J7, Canada

*To whom correspondence should be addressed. †CIFAR Fellow.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: The computational prediction of a protein structure from its sequence generally relies on a method to assess the quality of protein models. Most assessment methods rank candidate models using heavily engineered structural features, defined as complex functions of the atomic coordinates. However, very few methods have attempted to learn these features directly from the data.

Results: We show that deep convolutional networks can be used to predict the ranking of model structures solely on the basis of their raw three-dimensional atomic densities, without any feature tuning. We develop a deep neural network that performs on par with state-of-the-art algorithms from the literature. The network is trained on decoys from the CASP7 to CASP10 datasets and its performance is tested on the CASP11 dataset. On the CASP11 stage 2 dataset, it achieves a loss of 0.064, whereas the best performing method achieves a loss of 0.063. While the network learns to assess structural decoys globally and does not rely on any predefined features, it can be analyzed to show that it implicitly identifies regions that deviate from the native structure.

Availability: The code and the datasets are available at https://github.com/lamoureux-lab/3DCNN_MQA.

Contact: georgy.derevyanko@gmail.com or guillaume.lamoureux@concordia.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The protein folding problem remains one of the outstanding challenges in structural biology (Dill and MacCallum, 2012). It is usually defined as the task of predicting the three-dimensional (3D) structure of a protein from its amino acid sequence. Progress in the field is monitored through the Critical Assessment of protein Structure Prediction (CASP) competition (Moult *et al.*, 1995), in which protein folding methods are evaluated in terms of their accuracy at predicting structures ahead of their publication. Most methods participating in CASP include a conformational sampling step, which generates a number of plausible protein conformations, and a quality assessment step, which attempts to select the conformations closest to the unknown native structure.

In this work we explore the application of deep learning to the problem of "model quality assessment" (MQA), also called "estimation of model accuracy" (EMA) (Kryshtafovych *et al.*, 2016). Deep learning has recently garnered considerable interest in the research community (LeCun *et al.*, 2015), particularly in computer vision and natural language processing. Unlike more "shallow" machine learning approaches, deep learning improves performance by learning a hierarchical representation of the raw data at hand. It alleviates the need for feature engineering, which has traditionally constituted the bulk of the work done by researchers.

Deep learning has been applied to biological data and has yielded remarkable results for predicting the effects of genetic variations on human RNA splicing (Xiong *et al.*, 2015), for identifying DNA- and RNA-binding motifs (Alipanahi *et al.*, 2015), and for predicting the effects of non-coding DNA variants with single nucleotide precision (Zhou and Troyanskaya, 2015). These successes have one thing in common: they use raw data directly as input and do not attempt to engineer features from them.

Deep-learning-inspired methods have been used for protein structure quality assessment as well. For instance, DeepQA (Cao *et al.*, 2016) uses

9 scores from other MQA methods and 7 physico-chemical features extracted from the structure as input features to a deep restricted Boltzmann machine (Hinton *et al.*, 2006). The method has been reported to outperform ProQ2 (Ray *et al.*, 2012), which was the top-performing method in the CASP11 competition (Kryshtafovych *et al.*, 2016). ProQ3D (Uziela *et al.*, 2017) uses the same high-level input features as the earlier ProQ3 method (Uziela *et al.*, 2016) but achieves better performance by replacing the support vector machine model by a deep neural network. Since the original ProQ3 method had one of the top performances in CASP12 (Elofsson *et al.*, 2017), it can be expected that ProQ3D performs equally well.

Although both DeepQA and ProQ3D methods are based on deep neural networks, they use high-level features as input. In that sense, they use deep learning models more as traditional "shallow" classifiers than as end-to-end learning models. It is likely that they do not get all the advantages offered by the deep learning approach. By comparison, the DL-Pro algorithm (Nguyen *et al.*, 2014) uses a slighly more raw input, consisting of the eigenvectors of the Cα-to-Cα distance matrix. The model itself is an autoencoder (Hinton and Salakhutdinov, 2006) trained to classify the structures into either "near native" or "not near native".

More in line with the "end-to-end" spirit of deep learning, methods using as input a 3D representation of the structure have been developed to score protein-ligand poses (Wallach *et al.*, 2015; Ragoza *et al.*, 2017), to predict ligand-binding protein pockets (Jiménez *et al.*, 2017), and to predict the effect of a protein mutation (Torng and Altman, 2017). The molecules of interest are treated as 3D objects represented on a grid and the predictions are obtained from that information only. While a rigorous comparison of these methods is not always possible, they appear to improve on the state of the art: both AtomNet (Wallach *et al.*, 2015) and the 3D convolutional neural network of Ragoza *et al.* (2017) perform consistently better than either Smina (Koes *et al.*, 2013) or AutoDock Vina (Trott and Olson, 2010). For small molecules, Schütt *et al.* (2017b,a) and Smith *et al.* (2017) have recently developed deep neural networks to predict the molecular energy of a variety of chemical compounds in various conformations (or even various isomeric states). These models, intended to be used as universal force fields, are trained on ab initio quantum energies and forces, and use only the nuclear charges and the interatomic distance matrix as input.

2 Materials and Methods

2.1 Datasets

We train and assess our method using the datasets of non-native protein conformations ("decoys") from the CASP competition (Moult *et al.*, 2014). We use the CASP7 to CASP10 data as training set and the CASP11 data as test set, for a total of 564 target structures in the training set and 83 target structures in the test set. Each target from the training set has 282 decoys on average. The test dataset is split into two subsets (Kryshtafovych *et al.*, 2016): "stage 1" with 20 decoys per target selected randomly from all server predictions and "stage 2" with, for each target, the 150 decoys considered best by the Davis-QAconsensus evaluation method (Kryshtafovych *et al.*, 2016). The native structures are excluded from both training and test datasets. To make the structural data more consistent we optimize the side chains of all decoy structures using SCWRL4 (Krivov *et al.*, 2009).

Training and test datasets cover a similar range of sequence lengths (see Figure S1 in Supplementary Information). To confirm that the training and test sets are significantly different, we have aligned all test sequences against all training sequences using blastp (Altschul *et al.*, 1990). Less than 11% of the targets in the test set (9 out of 83) have sequence similarity with any target in the training set (see Table S1 in Supplementary Information).

To further assess the similarity of the two datasets, we have computed their overlap in terms of Pfam families (Finn *et al.*, 2016). The families were found using HMMER (Finn *et al.*, 2015) with an E-value cutoff of 1.0 (Finn *et al.*, 2016). Accounting for targets for which no Pfam family could be determined, approximately 25% of the test set targets share a family with approximately 10% of the training set targets (see Table S2 in Supplementary Information).

We have also compared the structures in the training and test sets using the ECOD database (Cheng *et al.*, 2014). This database provides a 5-tiered classification of all structures in the PDB according to the following criteria: architecture (A-group), possible homology (X-group), homology (H-group), topology (T-group), and family (F-group). Since the ECOD classification is domain-based, multi-domain protein chains can belong to multiple A-, X-, H-, T-, or F-groups. A summary of the overlap between the training and test sets is presented in Figure 1. For each target domain in the test set (T0759 to T0858), a black tile indicates that at least one structure from the training set belongs to the same ECOD group. (See Figure S2 in Supplementary Information for another representation of the overlap data.)

Table 1. Atom types used in this work. Atoms in each group are identified using their standard PDB residue names and atom names. Asterisks (*) correspond to either 1, 2, or 3.

Type	Description	Atoms
1	Sulfur/selenium	CYS:SG, MET:SD, MSE:SE
2	Nitrogen (amide)	ASN:ND2, GLN:NE2, backbone N (including N-terminal)
3	Nitrogen (aromatic)	HIS:ND1/NE1, TRP:NE1
4	Nitrogen (guanidinium)	ARG:NE/NH*
5	Nitrogen (ammonium)	LYS:NZ
6	Oxygen (carbonyl)	ASN:OD1, GLN:OE1, backbone O (except C-terminal)
7	Oxygen (hydroxyl)	SER:OG, THR:OG1, TYR:OH
8	Oxygen (carboxyl)	ASP:OD*, GLU:OE*, C-terminal O, C-terminal OXT
9	Carbon (sp2)	ARG:CZ, ASN:CG, ASP:CG, GLN:CD, GLU:CD, backbone C
10	Carbon (aromatic)	HIS:CG/CD2/CE1, PHE:CG/CD*/CE*/CZ, TRP:CG/CD*/CE*/CZ*/CH2, TYR:CG/CD*/CE*/CZ
11	Carbon (sp3)	ALA:CB, ARG:CB/CG/CD, ASN:CB, ASP:CB, CYS:CB, GLN:CB/CG, GLU:CB/CG, HIS:CB, ILE:CB/CG*/CD1, LEU:CB/CG/CD*, LYS:CB/CG/CD/CE, MET:CB/CG/CE, MSE:CB/CG/CE, PHE:CB, PRO:CB/CG/CD, SER:CB, THR:CB/CG2, TRP:CB, TYR:CB, VAL:CB/CG*, backbone CA

2.2 Input

Each protein structure is represented by 11 density maps corresponding to the atom types defined in Table 1. These atom types are a simplification of the 20 types proposed by Huang and Zou (2006, 2008), to reduce the memory footprint of the model. The density of an atom is represented using the function

$$\rho(r) = \begin{cases} e^{-\frac{r^2}{2}} & \text{if } r \leq 2.0 \text{ Å} \\ 0 & \text{if } r > 2.0 \text{ Å} \end{cases} \quad (1)$$

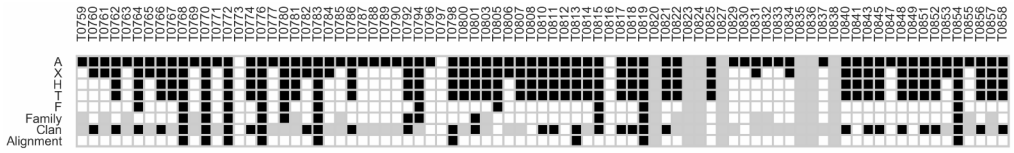

Figure 1. Overlap of the training set on each target domain of the test set (from T0759 to T0858). The first 5 rows of tiles correspond to the ECOD classification of protein domains (A-, X-, H-, T-, and F-groups). A black tile in any of these rows indicates that at least one structure from the training set belongs to the same ECOD group as the target. A white tile indicates that no structure belongs to the same group. Targets for which no ECOD classification is available are left empty (grey). A black tile in the "Family" row indicates that at least one structure from the training set belongs to the same Pfam family as the target. (Grey indicates that no Pfam family information is available for the target.) The "Clan" row shows similar information for Pfam clans. A black tile in the "Alignment" row indicates that at least one sequence in the training set aligns to the target sequence with an E-value smaller than 10^{-4}. (Grey indicates that the protein structure is absent from the PDB)

The atomic density is projected to the grid corresponding to its atom type. Each grid has a resolution of 1 Å and has $120 \times 120 \times 120$ cells. Figure 2 illustrates the atomic densities for a simple α-helical peptide (PDB code 5eh6). This structure contains only 7 of the 11 atom types, and only those density maps are shown. The 4 other maps have zero density everywhere.

2.3 Model

In this work we score protein structures using 3D convolutional neural networks (CNNs). CNNs were first proposed for image recognition by LeCun *et al.* (1989) and first applied to biological data by Bengio *et al.* (1990). Convolutional neural networks have gained wider recognition after the ImageNet 2012 competition (Krizhevsky *et al.*, 2012). The architecture of the model is shown in Figure 3. It is comprised of four blocks of alternating convolutional, batch normalization, and ReLU layers (terminated by a maximum pooling layer), followed by three fully-connected layers with ReLU nonlinearities. The final output of the network is a single number, interpreted as the score of the input structure. (See Table S3 in Supplementary Information for more details.)

Each 3D convolutional layer takes N input density maps f^{in} and transforms them using M filters F according to the following formula:

$$f_i^{\text{out}}(\mathbf{r}) = \sum_{j=1}^{N} \int F_i(\mathbf{r} - \mathbf{r}') \cdot f_j^{\text{in}}(\mathbf{r}')\, d\mathbf{r}', \quad \forall i \in [1, M] \tag{2}$$

In practice, these convolutions are approximated by sums on a 3D grid. The ReLU nonlinearity is computed as follows:

$$f_i^{\text{out}}(\mathbf{r}) = \begin{cases} f_i^{\text{in}}(\mathbf{r}) & \text{if } f_i^{\text{in}}(\mathbf{r}) \geq 0 \\ 0 & \text{if } f_i^{\text{in}}(\mathbf{r}) < 0 \end{cases}, \quad \forall i \in [1, M] \tag{3}$$

The idea of batch normalization was introduced by Ioffe and Szegedy (2015) to reduce the shift in the distribution of subnetwork outputs during training. This layer normalizes each input value according to the mean and variance within the subset of examples used to estimate the gradient (the "batch"):

$$\hat{f}_k^{\text{in}}(\mathbf{r}) = \frac{f_k^{\text{in}}(\mathbf{r}) - \mu_{\text{B}}(\mathbf{r})}{\sqrt{\sigma_{\text{B}}^2(\mathbf{r}) + \epsilon}}, \quad \forall k \in [1, N_{\text{B}}] \tag{4}$$

where $\mu_{\text{B}}(\mathbf{r})$ is the mean of all $f^{\text{in}}(\mathbf{r})$ maps from the batch (calculated at each position $\mathbf{r}$) and $\sigma_{\text{B}}^2(\mathbf{r})$ is the variance. N_{B} is the number of examples in the batch. The constant $\epsilon = 10^{-5}$ is added to avoid division by zero. The output of the layer is computed by scaling the normalized inputs:

$$f_k^{\text{out}}(\mathbf{r}) = \gamma \hat{f}_k^{\text{in}}(\mathbf{r}) + \beta, \quad \forall k \in [1, N_B] \tag{5}$$

Parameters γ and β are learned along with other parameters of the network during the training.

The maximum pooling layer ("MaxPool") is used to build a coarse-grained representation of the input. The output of this layer is the maximum over the cubes of size $d \times d \times d$ that cover the input domain with a stride l in each direction. This operation makes the output size approximately l times smaller than the input in each direction. All four "MaxPool" layers of the model (Figure 3) use $d = 3$ and $l = 2$.

During the coarse-graining procedure, the size of the individual data grids eventually shrinks to a single cell. The flattening layer reshapes the array of $1 \times 1 \times 1$ density maps into a single vector. Afterwards, we compute several transformations using fully-connected layers. Each of these layers transform a vector $\mathbf{x}_{\text{in}}$ as follows:

$$\mathbf{x}_{\text{out}} = W \cdot \mathbf{x}_{\text{in}} + \mathbf{b} \tag{6}$$

where W is a rectangular matrix and $\mathbf{b}$ is a vector, learned during the training. Each output vector is then transformed by a ReLU layer.

2.4 Training loss function

The problem of decoy quality assessment is essentially a ranking problem: we have to arrange decoys according to their similarity to the corresponding native structure as quantified, for instance, by the GDT_TS score (Zemla *et al.*, 2001). Such a ranking approach has recently been used by the MQAPRank method (Jing *et al.*, 2016), which, however, relies on a support vector machine model and uses high-level features as input.

We define the training loss function in terms of the margin ranking loss (Joachims, 2002; Gong *et al.*, 2013) for each pair of decoys. Let GDT_TS_i denote the global distance test total score of decoy i and let y_{ij} be the ordering coefficient of two decoys i and j:

$$y_{ij} = \begin{cases} 1 & \text{if } \text{GDT\_TS}_i \leq \text{GDT\_TS}_j \\ -1 & \text{if } \text{GDT\_TS}_i > \text{GDT\_TS}_j \end{cases} \tag{7}$$

The original GDT_TS score covers the range $[0, 100]$ but in this work we use a GDT_TS score normalized to the range $[0, 1]$, so that the loss stays within reasonable bounds. Let s_i denote the output of the network for decoy i. We use the following expression for the pairwise ranking loss:

$$L_{ij} = w_{ij} \max\left[0, 1 - y_{ij} \cdot (s_i - s_j)\right] \tag{8}$$

The coefficient w_{ij} represents the weight of each example and is defined so that decoys with similar scores (within 0.1) are removed from the training:

$$w_{ij} = \begin{cases} 1 & \text{if } \left|\text{GDT\_TS}_i - \text{GDT\_TS}_j\right| > 0.1 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

During the training procedure we load N_{B} decoy structures of a given target into memory (a "batch") and compute the output of the network and

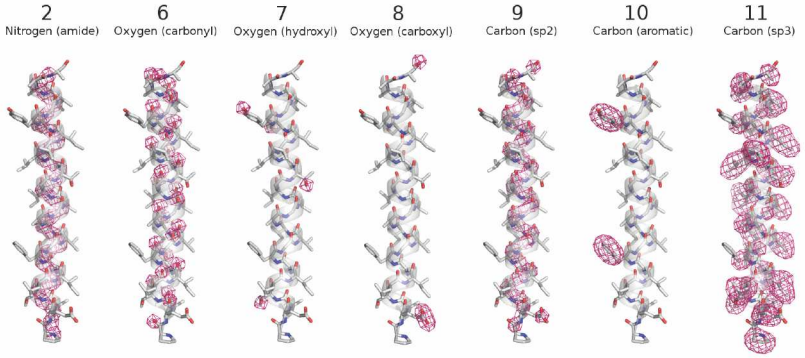

Figure 2. Representation of a protein structure (PDB code 5eh6) using atomic densities. The density maps are calculated according to Eq. 1 and rendered using Pymol (Schrödinger, LLC, 2015) with an isosurface level of 0.5.

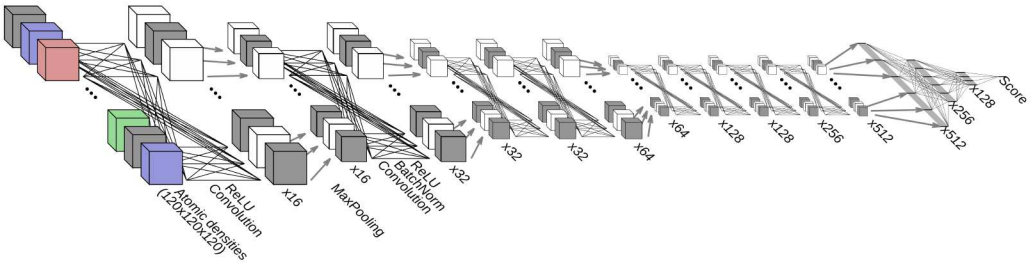

Figure 3. Schematic representation of the convolutional neural network architecture used in this work. Unless otherwise specified, line connections across boxes denote the consecutive application of a 3D convolutional layer ("Convolution"), a batch normalization layer ("BatchNorm"), and a ReLU layer. Grey arrows between boxes denote maximum pooling layers ("MaxPooling"). Labels "$\times M$" denote the number of 3D grids and the number of filters used in the corresponding convolutional layer. The grey stripes denote one-dimensional vectors and crossed lines between them stand for fully-connected layers with ReLU nonlinearities. Details of the model can be found in Supplementary Information Table S3.

the average ranking loss:

$$L = \frac{1}{N_{\mathrm{B}}^2} \sum_{i=1}^{N_{\mathrm{B}}} \sum_{j=1}^{N_{\mathrm{B}}} L_{ij} \tag{10}$$

2.5 Evaluation criteria

We evaluate the model using various correlation coefficients of the scores and using a evaluation loss function distinct from the training loss function. The evaluation loss is defined, for any given protein, as the absolute difference between the GDT_TS of the best decoy and the GDT_TS of the decoy with the lowest predicted score s:

$$\mathrm{Loss} = \left| \max_i \{\mathrm{GDT\_TS}_i\} - \mathrm{GDT\_TS}_{\mathrm{argmin}_i(s_i)} \right| \tag{11}$$

The correlation coefficients between the s score produced by the model and the GDT_TS score are computed for all decoys of a given target in the test set and are then averaged over all targets. Since the value of GDT_TS increases with the quality of a model but the value of s decreases, an ideal MQA algorithm would show the correlation coefficient of -1 and zero loss. These two evaluation criteria measure different qualities of the model. One the one hand, a perfect correlation coefficient of -1 would be achieved if the algorithm ranks all decoys in the exact order of their GDT_TS score (from best to worst). On the other hand, a zero loss would be achieved if the algorithm systematically assigns the lowest s value to the decoy with the highest GDT_TS score, irrespective of the s value it assigns to the other decoys.

2.6 Optimization and dataset sampling

The parameter optimization of the model was performed using the Adam algorithm (Kingma and Ba, 2014). The gradient of the average training loss function (Eq. 10) with respect to the model parameters is computed on the pairs of models in the batch. The batch size was set to $N_{\mathrm{B}} = 9$ models.

The training dataset is sampled by first choosing a random target from the dataset, then sampling decoys of this target. One epoch corresponds to one pass through all targets in the dataset. The decoys are sampled in a homogeneous way, by dividing all decoys of a given target into N_{B} bins according to the value of their GDT_TS score and by picking one decoy from each bin at random. Precisely, decoy i belongs to bin number

$$1 + \left\lfloor N_B \times \frac{\mathrm{GDT\_TS}_i - \min(\mathrm{GDT\_TS})}{\max(\mathrm{GDT\_TS}) - \min(\mathrm{GDT\_TS})} \right\rfloor \tag{12}$$

where $\max\{\mathrm{GDT\_TS}\}$ and $\min\{\mathrm{GDT\_TS}\}$ are computed on all decoys of the chosen target. If a bin is empty, the decoy is picked from another non-empty bin chosen at random. The order of targets and the order of decoys in the bins are shuffled at the end of each epoch.

Decoy structures are randomly rotated and translated each time they are used as input. The rotations are sampled uniformly (Shoemake, 1992) and the translation are chosen in such a way that the translated protein fits inside the 120 Å $\times$ 120 Å $\times$ 120 Å input grid (see text in Supplementary Information for details).

We select the final model based on its performance on a *validation subset* consisting of 35 targets (and their decoys) picked at random from

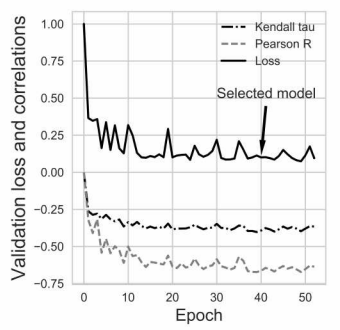

Figure 4. Evaluation loss (Eq. 11), Kendall τ, and Pearson R coefficients evaluated on the validation subset during the training procedure. One epoch corresponds to a cycle over all targets in the training subset. Models are saved every 10 epochs and the arrow shows the minimum validation loss for which a model was saved (at epoch 40).

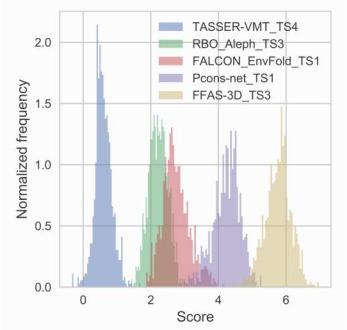

Figure 5. Distributions of the s scores of five decoys for target T0832 under random translations and rotations. A lower score represents a higher quality.

Table 2. Performance of the model from epoch 40 on the training and validation subsets.

Data	Loss (Eq. 11)	Pearson R	Spearman ρ	Kendall τ
Training subset	0.146	0.71	0.61	0.45
Validation subset	0.135	0.71	0.59	0.44

the training set and excluded from the training procedure. The remaining 529 targets are called the *training subset*. Figure 4 shows the Kendall τ and Pearson R coefficients and the evaluation loss on the validation subset over 52 epochs of training. Models are saved every 10 epochs and we pick the one that has the smallest evaluation loss (at epoch 40). Table 2 summarizes the performance metrics on the training and validation sets for the model at epoch 40. (See Figure S3 for results broken by target.)

3 Results

Ideally, the score assigned to a decoy should not depend on its position and orientation in space. To allow the model to learn this invariance, the rotational and translational degrees of freedom of all decoy structures are randomly sampled during the training. Figure 5 shows the distributions of scores for several decoy structures of the same target (T0832), calculated using the trained model for 900 rotations and translations sampled uniformly. While the score of a given structure is not strictly invariant under rotation and translation, it has a relatively narrow, unimodal distribution. (See Figure S4 in Supplementary Information for a distribution of score under rotations and translations separately.) More importantly, the difference between the average scores of two decoys is usually larger than their standard deviations. To reduce the influence of the choice of rotation and translation on the final ranking, we estimate the score of each decoy from the average of 90 scores calculated for random rotations and translations.

Table 3 shows a comparison of our model (3DCNN) with a number of state-of-the-art MQA methods: ProQ2D, ProQ3D (Uziela *et al.*, 2017), VoroMQA (Olechnovič and Venclovas, 2017), and RWplus (Zhang and Zhang, 2010). (See Figures S5 and S6 for ranking results broken by target.) ProQ2D uses a number of carefully crafted features such as atomic contacts, residue-residue contacts, surface accessibilities (as found in the structure and as predicted from the sequence), and secondary structure (observed and predicted). ProQ3D employs the same features as ProQ2D, as well as some Rosetta energy terms (Leaver-Fay *et al.*, 2011). RWplus, similar to DOPE (Shen and Sali, 2006) and DFIRE (Zhou and Zhou, 2002), uses a scoring approach based on statistical pairwise potentials.

Table 3. Performance comparison of our method (3DCNN) with other state-of-the-art MQA methods on the CASP11 dataset stages 1 and 2 (see text). The table reports the absolute, per-target average values of the correlation coefficients.

MQA method	Loss (Eq. 11)	Pearson R	Spearmann ρ	Kendall τ
Stage 1				
ProQ3D	0.046	0.755	0.673	0.529
ProQ2D	0.064	0.729	0.604	0.468
3DCNN	0.064	0.535	0.425	0.325
VoroMQA	0.087	0.637	0.521	0.394
RWplus	0.122	0.512	0.402	0.303
Stage 2				
VoroMQA	0.063	0.457	0.449	0.321
3DCNN	0.064	0.421	0.409	0.288
ProQ3D	0.066	0.452	0.433	0.307
ProQ2D	0.072	0.437	0.422	0.299
RWplus	0.089	0.206	0.248	0.176

VoroMQA uses knowledge-based potentials that depend on the contact surface between pairs of heavy atoms in the protein (or the solvent). Its approach is distinct from both the machine-learning techniques exemplified by the "ProQ" methods and the statistical potential techniques exemplified by the RWplus method. The methods chosen have available codes and could be re-evaluated on our CASP11 benchmark. Targets T0797, T0798, T0825 were removed from the benchmark because they were released for multimeric prediction.

Despite relying solely on atomic coordinates, the 3DCNN model (Figure 3) achieves a performance comparable to those of the heavily engineered ProQ2D and ProQ3D models, with evaluation losses either slightly above or slightly below, depending on the test set. Moreover, like that of VoroMQA, its performance increases as structural similarity with the training set increases (see Figure S7).

3.1 Analysis

In this section we show that the 3DCNN network has learned a relevant description of the protein structure and not merely artifacts of the dataset that correlate with the desired outcome.

First, we identify the regions of a decoy structure that are responsible for an increase of its score (a *decrease* in its quality). If the network has learned interpretable features of the input, we expect these parts of the decoy to deviate from the native structure. We use the Grad-CAM analysis technique proposed by Selvaraju *et al.* (2016). The key idea of this technique is to compute the gradient of the final score with respect to the

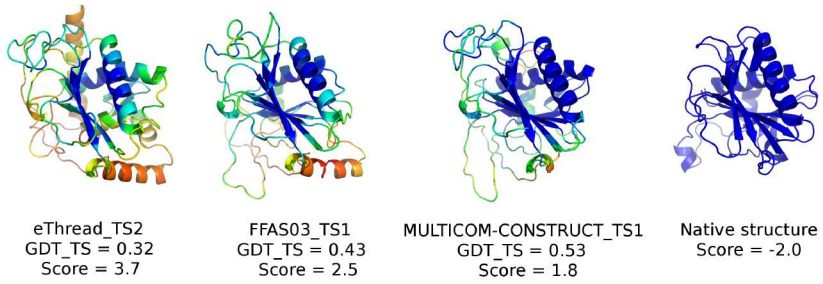

Figure 6. Output of the Grad-CAM analysis for layer 10 of the network projected on four decoys of target T0786 (PDB code 4QVU). The values are represented using the "rainbow" color scheme. The decoys are arranged in increasing quality from left to right, with the native structure on the right. Each decoy is aligned on the native structure and viewed in the same orientation.

output of a certain layer of the network, then compute the sum of this layer output weighted by the gradient. The weighted sum highlights the regions of the layer that are both strongly activated and highly influential on the final score. To generate an interpretable map, the weighted sum is then scaled up to the size of the input of the network, using tri-linear interpolation. This up-sampled map indicates which parts of the input contribute the most to the gradient of the score. In our case we choose to analyze layer 10, for which the output grid size is $25 \times 25 \times 25$. We tested the method on neighboring layers and layer 10 represents the best tradeoff between interpretability and coarseness. In line with our scoring procedure, we average the results from the Grad-CAM analysis over 90 rotations and translations of the decoy. We obtain the Grad-CAM output for each transformation and project it onto the atoms of the decoy.

Figure 6 shows a projection of the Grad-CAM results onto the atoms of four decoys of target T0786, represented as a color-coded value on the cartoon rendering of the structures. The orange/yellow regions are mainly found at the surface of the lower-quality decoys while the blue/green regions are found at the core. This indicates that the quality of the decoy would go down for any increase in atomic density at the surface but would be unaffected by an increase in density at the core (see Figure S8). It also suggests that the neural network recognizes and enforces packing. Moreover, we see that the higher secondary structure content of decoy eThread_TS2 actually decreases the score. This suggests that the 3DCNN network does not estimate the quality based only on the presence of local structural elements but that it detects large-scale features of the fold as well. Interestingly, we find that the Grad-CAM outputs are mostly zero for decoys close to the native structure, despite the fact that no gradient information was included in the training procedure (see Table S4 in Supplementary Information).

To verify that the network does not rely on artifacts in the data to rank decoys, we have assessed its performance on a second, independent dataset generated by the 3DRobot algorithm (Deng *et al.*, 2016). This algorithm yields decoys that are uniformly distributed within an RMSD range of 0 to 12 Å away from the native structure. The 3DRobot dataset consists of 300 decoys for each of 200 single-domain proteins selected from the PDB. The proteins have less than 20% sequence identity with one another and are between 80 to 250 residue long. Out of the 200 proteins, 48 are all-α proteins, 40 are all-β, and 112 are α/β. For these 60000 decoy structures, the pre-trained network yields an absolute per-target average Pearson R coefficient of 0.85. Spearman ρ coefficient and Kendall τ coefficient are 0.83 and 0.64, respectively (see Figure S9 for representative examples of score versus GDT_TS plots). This confirms that the 3DCNN model can successfully rank unrelated datasets.

4 Discussion

This work shows that it is possible to construct an algorithm that learns to assess the quality of protein models from a raw representation. We have used 3D atomic densities broken down by atom types as such representation. However it is clear that any other physical quantity defined on a grid can be employed, such as the electrostatic potential calculated using Poisson-Boltzmann equation (Honig and Nicholls, 1995) or the solvent density calculated using 3D-RISM (Stumpe *et al.*, 2011). So far, no other MQA method has managed to include these crucial properties.

The loss function we used for training does not aim to predict GDT_TS of a decoy, but rather to sort decoys according to their quality. We chose this strategy so that the score can be interpreted as an energy function that has a local minimum for the native structure. In future work we plan to add terms to the loss function that penalize the first and second order derivatives of the loss at the native structure, to ensure that the score indeed reaches a local minimum there.

This work also identifies important avenues for improvement. First, the model captures the invariance of the score under translations and rotations only in an approximate way. This invariance problem can however be solved using the approach of Worrall *et al.* (2016), in which the coefficient space of the convolutional filters is restricted to circular harmonics, which encodes equivariance under rotations at each layer of the network and leads to invariance of the final output. Second, the output of the model remains difficult to interpret. While interpretation of deep neural networks remains an important research problem, the field is undergoing rapid progress. For instance, recently published work (Bau *et al.*, 2017) has shown that intepretability can be quantified using extensively labeled image datasets that contain the bounding boxes and labels for fine-grained features such as body parts or car parts. In the case of protein models, many such labels (and bounding boxes) are readily available: amino acids, secondary structure elements, hydrogen bond networks, disulfide bonds, etc. Unlike in conventional machine learning models, these features would not be used for prediction but for interpretation of the prediction.

Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) [RGPIN 355789 to G.L., RGPIN 1016552 to Y.B.]; and the Canada Research Chair and Canadian Institute for Advanced Research (CIFAR) programs [to Y.B.]. Computational resources were provided by Calcul Québec and Compute Canada.

References

Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. (2015). Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. *Nat. Biotechnol.*, **33**, 831–838.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *J. Mol. Biol.*, **215**(3), 403–410.

Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. (2017). Network Dissection: Quantifying Interpretability of Deep Visual Representations. *arXiv:1704.05796*.

Bengio, Y., Bengio, S., Pouliot, Y., and Agin, P. (1990). A neural network to detect homologies in proteins. In *Advances in Neural Information Processing Systems*, pages 423–430.

Cao, R., Bhattacharya, D., Hou, J., and Cheng, J. (2016). DeepQA: Improving the estimation of single protein model quality with deep belief networks. *BMC Bioinformatics*, **17**, 495.

Cheng, H., Schaeffer, R. D., Liao, Y., Kinch, L. N., Pei, J., Shi, S., Kim, B.-H., and Grishin, N. V. (2014). ECOD: An evolutionary classification of protein domains. *PLoS Comput. Biol.*, **10**(12), e1003926.

Deng, H., Jia, Y., and Zhang, Y. (2016). 3DRobot: automated generation of diverse and well-packed protein structure decoys. *Bioinformatics*, **32**(3), 378–387.

Dill, K. A. and MacCallum, J. L. (2012). The Protein-Folding Problem, 50 Years On. *Science*, **338**(6110), 1042–1046.

Elofsson, A., Joo, K., Keasar, C., Lee, J., Maghrabi, A. H. A., Manavalan, B., McGuffin, L., Menendez Hurtado, D., Mirabello, C., Pilstal, R., Sidi, T., Uziela, K., and Wallner, B. (2017). Methods For Estimation Of Model Accuracy In CASP12. *bioRxiv:143925*.

Finn, R. D., Clements, J., Arndt, W., Miller, B. L., Wheeler, T. J., Schreiber, F., Bateman, A., and Eddy, S. R. (2015). HMMER web server: 2015 update. *Nucleic Acids Res.*, **43**(W1), W30–W38.

Finn, R. D., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Mistry, J., Mitchell, A. L., Potter, S. C., Punta, M., Qureshi, M., Sangrador-Vegas, A., *et al.* (2016). The Pfam protein families database: towards a more sustainable future. *Nucleic Acids Res.*, **44**(D1), D279–D285.

Gong, Y., Jia, Y., Leung, T., Toshev, A., and Ioffe, S. (2013). Deep convolutional ranking for multilabel image annotation. *arXiv:1312.4894*.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, **313**(5786), 504–507.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Comput.*, **18**(7), 1527–1554.

Honig, B. and Nicholls, A. (1995). Classical electrostatics in biology and chemistry. *Science*, **268**(5214), 1144–1149.

Huang, S.-Y. and Zou, X. (2006). An iterative knowledge-based scoring function to predict protein–ligand interactions: I. Derivation of interaction potentials. *J. Comput. Chem.*, **27**(15), 1866–1875.

Huang, S.-Y. and Zou, X. (2008). An iterative knowledge-based scoring function for protein–protein recognition. *Proteins*, **72**(2), 557–579.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*.

Jiménez, J., Doerr, S., Martínez-Rosell, G., Rose, A., and De Fabritiis, G. (2017). DeepSite: Protein binding site predictor using 3D-convolutional neural networks. *Bioinformatics*.

Jing, X., Wang, K., Lu, R., and Dong, Q. (2016). Sorting protein decoys by machine-learning-to-rank. *Sci. Rep.*, **6**, 31571.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. Association for Computing Machinery.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Koes, D. R., Baumgartner, M. P., and Camacho, C. J. (2013). Lessons learned in empirical scoring with Smina from the CSAR 2011 benchmarking exercise. *J. Chem. Inf. Model.*, **53**(8), 1893–1904.

Krivov, G. G., Shapovalov, M. V., and Dunbrack, R. L. (2009). Improved prediction of protein side-chain conformations with SCWRL4. *Proteins*, **77**(4), 778–795.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kryshtafovych, A., Barbato, A., Monastyrskyy, B., Fidelis, K., Schwede, T., and Tramontano, A. (2016). Methods of model accuracy estimation can help selecting the best models from decoy sets: Assessment of model accuracy estimations in CASP11. *Proteins*, **84**, 349–369.

Leaver-Fay, A., Tyka, M., Lewis, S. M., Lange, O. F., Thompson, J., Jacak, R., Kaufman, K. W., Renfrew, P. D., Smith, C. A., Sheffler, W., Davis, I. W., Cooper, S., Treuille, A., Mandell, D. J., Richter, F., Ban, Y.-E. A., Fleishman, S. J., Corn, J. E., Kim, D. E., Lyskov, S., Berrondo, M., Mentzer, S., Popović, Z., Havranek, J. J., Karanicolas, J., Das, R., Meiler, J., Kortemme, T., Gray, J. J., Kuhlman, B., Baker, D., and Bradley, P. (2011). Rosetta3: An Object-Oriented Software Suite for the Simulation and Design of Macromolecules. *Methods Enzymol.*, **487**, 545–574.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, **1**(4), 541–551.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, **521**(7553), 436–444.

Moult, J., Pedersen, J. T., Judson, R., and Fidelis, K. (1995). A large-scale experiment to assess protein structure prediction methods. *Proteins*, **23**(3), ii–v.

Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. (2014). Critical assessment of methods of protein structure prediction (CASP)—round x. *Proteins*, **82**(S2), 1–6.

Nguyen, S. P., Shang, Y., and Xu, D. (2014). DL-PRO: A Novel Deep Learning Method for Protein Model Quality Assessment. *2014 International Joint Conference on Neural Networks (IJCNN)*.

Olechnovič, K. and Venclovas, Č. (2017). VoroMQA: Assessment of protein structure quality using interatomic contact areas. *Proteins*, **85**(6), 1131–1145.

Ragoza, M., Hochuli, J., Idrobo, E., Sunseri, J., and Koes, D. R. (2017). Protein–Ligand Scoring with Convolutional Neural Networks. *J. Chem. Inf. Model*, **57**(4), 942–957.

Ray, A., Lindahl, E., and Wallner, B. (2012). Improved model quality assessment using ProQ2. *BMC Bioinformatics*, **13**, 224.

Schrödinger, LLC (2015). The PyMOL Molecular Graphics System, Version 1.8.

Schütt, K. T., Kindermans, P.-J., Sauceda, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. (2017a). MolecuLeNet: A continuous-filter convolutional neural network for modeling quantum interactions. *arXiv:1706.08566*.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. (2017b). Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.*, **8**, 13890.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2016). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. *arXiv:1610.02391*.

Shen, M.-y. and Sali, A. (2006). Statistical potential for assessment and prediction of protein structures. *Protein Sci.*, **15**(11), 2507–2524.

Shoemake, K. (1992). Uniform random rotations. In *Graphics Gems III*, pages 124–132. Academic Press Professional, Inc.

Smith, J. S., Isayev, O., and Roitberg, A. E. (2017). ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.*, **8**, 3192–3203.

Stumpe, M. C., Blinov, N., Wishart, D., Kovalenko, A., and Pande, V. S. (2011). Calculation of Local Water Densities in Biological Systems: A Comparison of Molecular Dynamics Simulations and the 3D-RISM-KH Molecular Theory of Solvation. *J. Phys. Chem. B*, **115**(2), 319–328.

Torng, W. and Altman, R. B. (2017). 3D deep convolutional neural networks for amino acid environment similarity analysis. *BMC Bioinformatics*, **18**, 302.

Trott, O. and Olson, A. J. (2010). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *J. Comput. Chem.*, **31**(2), 455–461.

Uziela, K., Shu, N., Wallner, B., and Elofsson, A. (2016). ProQ3: Improved model quality assessments using Rosetta energy terms. *Sci. Rep.*, **6**, 33509.

Uziela, K., Menéndez Hurtado, D., Shu, N., Wallner, B., and Elofsson, A. (2017). ProQ3D: improved model quality assessments using deep learning. *Bioinformatics*, **33**(10), 1578–1580.

Wallach, I., Dzamba, M., and Heifets, A. (2015). AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. *arXiv:1510.02855*.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. (2016). Harmonic Networks: Deep Translation and Rotation Equivariance. *arXiv:1612.04642*.

Xiong, H. Y., Alipanahi, B., Lee, L. J., Bretschneider, H., Merico, D., Yuen, R. K., Hua, Y., Gueroussov, S., Najafabadi, H. S., Hughes, T. R., Morris, Q., Barash, Y., Krainer, A. R., Jojic, N., Scherer, S. W., Blencowe, B. J., and Frey, B. J. (2015). The human splicing code reveals new insights into the genetic determinants of disease. *Science*, **347**(6218), 1254806.

Zemla, A., Venclovas, Č., Moult, J., and Fidelis, K. (2001). Processing and evaluation of predictions in CASP4. *Proteins*, **45**(S5), 13–21.

Zhang, J. and Zhang, Y. (2010). A novel side-chain orientation dependent potential derived from random-walk reference state for protein fold selection and structure prediction. *PloS One*, **5**(10), e15386.

Zhou, H. and Zhou, Y. (2002). Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein Sci.*, **11**(11), 2714–2726.

Zhou, J. and Troyanskaya, O. G. (2015). Predicting effects of noncoding variants with deep learning-based sequence model. *Nat. Methods*, **12**(10), 931–934.